Publisher: IISI - International Institute for Socio-Informatics
ISSN 1861-4280

international reports on socio-informatics

volume 19 issue 3
2022

Zur Siegener Perspektive auf Sozio-Informatik
Ein Dokument der Selbstverständigung

Autor:
Peter Brödner

The 'international reports on socio-informatics' are an online report series of the International Institute for Socio-Informatics, Bonn, Germany. They aim to contribute to current research discourses in the fields of 'Human-Computer-Interaction' and 'Computers and Society'. The 'international reports on socio-informatics' appear at least two times per year and are exclusively published on the website of the IISI.

Impressum

IISI - International Institute for Socio-Informatics
Stiftsgasse 25
53111 Bonn
Germany

fon: +49 228 6910-43
mail: iisi@iisi.de
web: http://www.iisi.de

Zur Siegener Perspektive auf Sozio-Informatik - Ein Dokument der Selbstverständigung

Peter Brödner
Universität Siegen, Germany
peter.broedner@t-online.de

1 Vorbemerkungen

Dieses Dokument ist aufgrund eines Gesprächs und einer Verabredung mit Volker Wulf im Zusammenhang mit der in Aussicht stehenden Gründung eines Instituts für Sozio-Informatik an der Universität Siegen angestoßen worden. Ein hierzu vorgelegter Entwurf diente dazu, eine als Diskurs angelegte Klausur mit externen Experten im Sommer 2021 durchzuführen, um Klarheit über Ziele, Kernaktivitäten und die wissenschaftliche Ausrichtung des Instituts zu gewinnen. Das Dokument fasst wesentliche Einsichten aus dem Diskurs zusammen.

Gegenstand und Charakter der im deutschsprachigen Raum (abweichend vom angelsächsischen Sprachraum) »Informatik« genannten Fachdisziplin sind trotz ihrer inzwischen 50-jährigen Geschichte noch immer ungeklärt und umstritten: Eine »Strukturwissenschaft« wie die Mathematik für die einen, eine »Ingenieurwissenschaft« mit starken sozialwissen-schaftlichen Bezügen für die anderen, die Lehre von der »rationalen, insbesondere ma-schinellen Informationsverarbeitung« etwa für die Académie Française oder die GI. Es erscheint daher geboten, sich bei dieser Gelegenheit erneut Kernfragen des Fachs zu stel-len.

Ein derartige Selbstreflexion steht in einer Reihe mit früheren Anstrengungen um eine »Theorie der Informatik« mit ähnlichem Anspruch, wie etwa Beiträge zum Sammelband »Sichtweisen der Informatik« (Coy et al. 1992), ein ausführlicher Eintrag über »Philosophy of Computer Science« in der Stanford Encyclopedia of Philosophy, der Beitrag von Floyd & Klaeren (1999) zum Fernstudien-Projekt »Informatik & Gesellschaft« oder Bemühungen von Denning (2003) um einen curricularen Kanon des Fachs: »Principles of Computing«.

Im Unterschied zur theoretischen Informatik, die logisch-mathematische Grundlagen des Fachs zu Fragen der »Aufzählbarkeit, Entscheidbarkeit und Berechenbarkeit« (Hermes 1972) bereit stellt, geht es einer theoretischen Perspektive auf Informatik um Fragen nach dem Gegenstand des Fachs und wesentlichen Methoden seiner Analyse, Beschreibung, Gestaltung und Bewertung. Theorie (abgeleitet von griech. theoro = ich sehe) ist stets perspektivisch, sie beschreibt, was von einem Standpunkt aus gesehen wird; in reflektierender Haltung dient sie der rationalen Klärung des Selbstverständnisses als Wissenschaft.

Als Teil der Informatik muss sich auch die Sozio-Informatik in diesem wissenschaftlichen Diskurs positionieren und vor diesem Hintergrund ihren besonderen Fokus, ihr Gegenstandsverständnis und ihre Vorgehensweisen artikulieren.

2 Am Anfang steht das Menschenbild

Vor allem anderen ist in solcher Perspektive zu bestimmen, von wo aus, aus welcher Position und Richtung auf (Sozio-)Informatik und und ihren Gegenstand geschaut wird. Da es ohne menschliches Handeln gar keine Computertechnik gäbe, muss folglich am Anfang die Frage nach der Natur menschlichen Handelns, nach dem Menschenbild stehen: Wie wir unser Verhältnis zur äußeren Natur, zu uns selbst und zu unseren Artefakten als Produkten unseres Tätigseins sehen. Und bereits hier scheiden sich die Geister.

Diesen Ausführungen liegt ein Menschenbild zugrunde, das tief in der humanistischen Tradition wurzelt und die Evolution von Natur und Kultur als Entwicklungsprozesse aufgrund von Selbstorganisation begreift, einer Bewegungsform der Materie fern vom Gleichgewicht, durch die per Energiezufuhr unter bestimmten Umgebungsbedingungen dauerhaft höher organisierte Ordnungsstrukturen von selbst hervorgebracht werden (die ihrerseits Bedingung der Möglichkeit weiterer Selbstorganisation sind). Dabei wird die kulturelle Evolution als die Fortsetzung der natürlichen mit anderen Mitteln gesehen.

Der homo sapiens als Produkt natürlicher Evolution markiert den Übergang vom Sein lebendiger Organismen zu bewusstem Sein, der ihn als soziales Naturwesen nicht nur zu absichts- und bedeutungsvollem Handeln befähigt, sondern auch dazu, sich zugleich der Handlungen selbst gewahr zu sein. Folglich ist er zu reflexiver Steuerung seines Handelns fähig. Entsprechend kennzeichnet ihn gemeinschaftliche Daseinsvorsorge und v.a. die durch seine bewusste gemeinschaftliche Tätigkeit geschaffene Welt der Kultur, deren Evo-lution sich, getrieben durch jeweils dominante Einsichten und Interessen, niederschlägt in Formen und Mustern der Herstellung und des Gebrauchs von

- Werkzeugen als Mitteln zweckmäßiger Nutzung von Naturkräften und -effekten (Kausa-lität: vermittelt Wirkungen),
- Zeichen als Mitteln sozialer Interaktion, Kommunikation und Reflexion (Sprache: vermittelt Bedeutungen).

Zurecht wird daher der homo sapiens auch als homo faber und »semiotisches Tier« (Hausdorff 1897) bezeichnet (der allerdings als animal laborans dem Stoffwechsel unterworfen bleibt; Arendt 1956). Im Zusammenhang mit Computertechnik interessiert im folgenden v.a. sein Umgang mit Zeichen. Mittels Zeichen (Peirce 1983) können in der Sphäre bewussten Seins Dinge oder Vorgänge der Lebenswelt bezeichnet, als etwas begriffen und in ihrem Zu-sammenhang gedacht und beschrieben werden:

- zunächst zwecks Kooperation lautlich gesprochen (Organisation des Stoffwechsels als gemeinschaftliche Daseinsvorsorge),
- dann zur Verwaltung von großem gesellschaftlichen Mehrprodukt auch schriftlich no-tiert mittels Schrift- und Zahlzeichen (›digital‹; Flusskulturen),

- zuletzt auch formalisiert und in Form von Algorithmen und Datenstrukturen binär co-diert (mit Methoden der ›Computing Science‹); Begreif- und Denkbares wird damit par-tiell berechenbar.

Der Zeichengebrauch (→ Semiotik; Peirce 1983) der Sprache konstituiert die intentional-kontingente soziale Welt der Bedeutungen im Unterschied zu kausalen physischen Wirkungen der Natur und erschließt damit Eigenschaften der Dinge (Taylor 2017). Dieser Sicht zufolge sind auch zwei grundverschiedene Arten von Maschinen entstanden (vgl. 2).

Dieses Menschenbild bildet die Grundlage der im folgenden entfalteten praxistheoretischen Zugangs (zusammenfassend: Reckwitz 2003) zu Gestaltung und Gebrauch von Computer-artefakten. Es unterscheidet sich fundamental von derzeit weit verbreiteten, wenn nicht noch immer dominanten Sichtweisen der rationalistischen Tradition, die das Handeln von Menschen mit dem Verhalten von Maschinen vergleichen und diesem letztlich gleichset-zen, so etwa

- das schon von Beginn an mit der Realisierung programmierbarer Rechenmaschinen verknüpfte »computational model of the mind« (»Elektronengehirn«, künstliche »neu-ronale Netze« oder »symbolische KI«);
- das der neoklassischen Ökonomik zugrunde liegende Modell des homo oeconomicus, das den vereinzelten Menschen als Nutzen maximierenden Automaten (gemäß seiner ihm eigenen Präferenzordnung) versteht;
- das dem »KI«-Diskurs wie auch der verhaltensorientierten Ökonomik zugrunde liegen-de Modell von voreingenommenen, gewohnheitsgesteuerten und irrational handelnden Menschen (EU High Level Expert Group), deren Verhalten es durch »rationale Informa-tionsverarbeitung« (Académie Française), »Propaganda« (Bernays 1928). datengetriebe-nes »Nudging« (Thaler & Sunstein 2012) oder »Künstliche Intelligenz« zu verbessern gilt.

Dem wird nachfolgend ein dem in der humanistischen Tradition verwurzelten Menschenbild entsprechendes Verständnis von Technik im Allgemeinen und von Computertechnik im Besonderen entgegen gestellt.

3 Technik: Herstellung und Gebrauch gebrauchstauglicher Artefakte

Gewöhnlich wird Technik als bloße Ansammlung von zweckmäßig gestalteten Artefakten missverstanden. Hingegen wird mit téchne im ursprüngliche Wortsinn eine List bezeichnet, die List einsichtiger Menschen, Wirkungen der äußeren Natur für eigene Zwecke nutzbar zu machen. Aristoteles sieht darin einen eigenen Teil unserer praktischen Vernunft, die Fähigkeit, etwas Nützliches herstellen zu können, beruhend auf Erfahrung, Übung und Einsicht in Naturverhältnisse.

Genauere Analyse zeigt, dass das auf Artefakte fixierte Technikverständnis in die Irre führt: Technische Artefakte und ihre Funktionen fallen nicht vom Himmel, sondern müssen, um wirksam und gebrauchstauglich zu sein, aufgrund von Einsichten in Prozesse der Natur oder in Vorgänge sozialer Praxis für bestimmte Zwecke mühsam konzeptuell ent-

worfen und materiell hergestellt werden. Als solche sind sie freilich bloß tote, nutzlose Gegenstände, solange ihre Funktionen nicht für bestimmte Aufgaben der Praxis zweckgemäß eingesetzt, mithin dafür angeeignet und praktisch wirksam verwendet werden. Das alles geschieht stets im Spannungsfeld einerseits des technisch Machbaren, der Gestaltbar-keit von Natur bzw. sozialer Praxis, und andererseits des sozial Wünschenswerten, ab-hängig von jeweils herrschenden Interessen (Brödner 1997).

Entsprechend wird nach allgemeinem professionellem Verständnis Technik definiert als die Gesamtheit von Maßnahmen zur Herstellung und zum Gebrauch künstlicher Mittel für gesellschaftliche Zwecke. Ihr werden damit nicht nur die Artefakte und Sachsysteme selbst zugerechnet, sondern auch deren sozial konstruierte und kulturell vermittelte Herstellung und Anwendung (Ropohl 1991, VDI 1991). Als geronnene Erfahrung verkörpern sie explizites, begrifflich-propositionales Wissen über Natur bzw. über soziale Praxis und als Arbeitsmittel stellen sie Handlungsanforderungen an ihren Gebrauch, durch den die Artefakte erst ihren Sinn erhalten und in ihrer Qualität zu beurteilen sind. Gerade in den Prozessen der Entwicklung und Herstellung technischer Artefakte sowie ihrer Aneignung zu praktisch wirksamer Verwendung liegen die eigentlichen Probleme von Technik als der ›Anstrengung, Anstrengungen zu ersparen‹, dem Sinn technischen Handelns; eben hierin liegen auch die Wurzeln misslingenden oder gar missbräuchlichen Umgangs.

Vor diesem Hintergrund können nun die fundamentalen Unterschiede zwischen klassischen Maschinen der Energie und Stoffumwandlung (»Kraft- und Arbeitsmaschinen«, chemi-sche oder biologische Prozesse) und Computern als semiotischen (»symbolischen«) Maschi-nen aufgezeigt werden (Krämer 1988). Erstere nutzen durchweg physikalische Kenntnisse der Thermo- bzw. Elektrodynamik und Mechanik, um zweckmäßig gestaltete Funktionen der Krafterzeugung und -übertragung zu realisieren und so Naturkräfte und -effekte für selbstbestimmte Zwecke nutzbar zu machen. Im Unterschied dazu greifen Computer in Zeichenprozesse sozialer Praxis ein, sind damit »symbolische« Maschinen, die zweckge-mäß programmgesteuert Zeichen(-träger) bzw. aus logischer Sicht ›Daten‹ verarbeiten (vgl. 4). Energie- und stoffumwandelnde Maschinen übertragen Kräfte, semiotische Ma-schinen manipulieren bedeutungslose Zeichen(-träger).

Diese fundamentalen Unterschiede zwischen beiden Maschinenklassen zeigen sich in deren Wirkbereichen, Funktionsweisen und Zwecken. Der Wirkbereich von Kraft- und Arbeits-maschinen (wie auch von artifiziellen chemischen und biologischen Prozessen) liegt in der Natur, indem natürliche Kräfte zweckgemäß für Prozesse der Energie- und Stoffumwand-lung genutzt werden, während der Wirkbereich semiotischer Maschinen überwiegend in zeichenbasierter sozialer Praxis liegt und auf wohl determinierten Funktionen der Verar-beitung von Daten als Zeichenträgern in zugrunde liegenden Zeichenprozessen beruht. Mit semiotischen Maschinen wird der soziale Raum der Zeichenprozesse und Interaktion nirgends verlassen. Auch wenn Computer zur Steuerung von Naturprozessen eingesetzt werden, funktionieren sie auf Basis zeichenbasierter Modelle derselben, zuvor durch Wis-sensgenese in sozialer Praxis geschaffen.

Die Funktionsweise von Maschinen und Prozessen der Energie- und Stoffumwandlung beruht auf der Einsicht in natürliche Effekte als Ergebnis von Naturerkenntnis und ihr

Zweck ist die Nutzung von Naturkräften. Die Funktionsweise semiotischer Maschinen beruht hingegen auf algorithmisch formal gefassten Regeln und Vorschriften zur Manipulation von Zeichen(-trägern) (logisch: ›Daten‹), gewonnen durch Analyse, Modellierung und Formalisierung von Zeichenprozessen kognitiver Arbeit und sie dient der Organisation und Koordination kollektiven Handelns (bzw. der modellgestützten Steuerung physischer Prozesse). Den zwecks Maschinisierung körperlicher Arbeit geschaffenen mechanischen Funktionen der Kraftübertragung als dem Kern maschineller Energie- und Stoffumwandlung entsprechen dann bei Computern als semiotischen Maschinen die algorithmischen Funktionen für Datenverarbeitung, -speicherung und -transfer zwecks Maschinisierung kognitiver Arbeit. Ebenso wie man Mechanik verstehen muss, um kraftübertragende Maschinen zu konstruieren, muss man sich mit Logik und Algorithmik – mit berechenbaren Funktionen – auskennen, um Computer als semiotische Maschinen entwerfen zu können (Brödner 2020).
Somit ist den technischen Artefakten beider Klassen gemeinsam, dass sie als gestaltete Kulturprodukte explizites methodisches und funktionales Wissen vergegenständlichen, das sich, gewonnen aus Begriffsbildung und theoretischen Einsichten über zugrunde liegende Prozesse, der natürlichen analytischen Intelligenz ihrer Konstrukteure verdankt. Die dadurch wohl bestimmten maschinellen Funktionen sind dann wiederum durch ihre Nutzer in deren Handlungskontext zu interpretieren, um sie wirkungsvoll zu gebrauchen (die funktionale ›Sprache‹ der Artefakte). Kraft der ihnen je eigenen Funktionen vollziehen Maschinen beider Klassen dann im Einsatz jeweils durch die Eingaben kausal determinierte, zweckgemäß gesteuerte wiederholbare Bewegungen. Um sinnvolle Eingaben machen und de-ren kausale Wirkungen interpretieren zu können, müssen Handlungen ihres Gebrauchs in der funktionalen ›Sprache der Artefakte‹ zum Ausdruck gebracht werden.

4 Historischer Abriss der Entwicklung der Computertechnik

Im historischen Rückblick zeigt sich, dass die Entwicklung beider Maschinen-Gattungen nahezu zeitgleich und auch aus gleichen Ursprüngen im Zuge der industriellen Revolution ihren Anfang nimmt. Zwar gibt es auch schon zuvor vereinzelte verblüffend komplizierte Maschinen (z.B. mechanische Vierspezies-Rechenmaschinen oder Musikautomaten), ihre machtvoll anhaltende Entwicklung setzt aber erst mit den Zwängen der Kapitalver-wertung im industriellen Kapitalismus auf der Grundlage beginnender Verwissenschaftli-chung von Produktion und ausgeprägter betrieblicher (›horizontaler‹) Arbeitsteilung ein (nach den Grundsätzen von Smith und Babbage).
Im Unterschied zu landläufigen Erzählungen, die den Beginn der Entwicklung von Computertechnik in der Zeit des zweiten Weltkriegs ansiedeln, reicht diese, jedenfalls was ihre sämtlichen logisch-konzeptionellen Grundlagen (»Software«) anbelangt, bis in den Übergang vom 18. ins 19. Jahrhundert zurück: In der Folge der Französischen Revolution hat der Nationalkonvent beschlossen, physikalische Grundgrößen wie etwa die Länge auf metrische Maßeinheiten (»Urmeter«) und das vorherrschende duodezimale Zahlensystem

auf das Dezimalsystem umzustellen. Vor dem Hintergrund bereits vorhandener großer naturwissenschaftlicher, mathematischer und technischer Wissensbestände stellt das die Gesellschaft vor die gigantische Aufgabe, insgesamt die umfangreich existierenden mathematischen Tafelwerke (Logarithmen- und trigonometrische Tafeln für physikalische und ingenieurtechnische Berechnungen, nautische Almanache für die Navigation, Geschütztafeln für Flugbahn-Berechnungen von Geschossen etc.) neu zu berechnen.
Der damit betraute Mathematiker Gaspard de Prony meistert diese Aufgabe in nur neun Jahren (1792-1801) auf der Grundlage extremer Teilung geistiger Rechenarbeit (wie sie bei körperlicher Arbeit bereits erprobt und dort Grundlage von Maschinenentwicklung ist). Dazu entwickelt er für die Berechnungsverfahren Formulare, worin die genaue Abfolge der einzelnen arithmetischen Operationen vorgeschrieben und die jeweiligen Operations-Ergebnisse festgehalten werden; jeder der vielen Operateure (englisch: »Computer«) führt nach Maßgabe des dort verzeichneten Berechnungsfortschritts die jeweils nächste Operati-on aus. Mittels der Durchführung der Berechnungsgänge in drei parallel arbeitenden Gruppen gelingt zudem eine gewisse Qualitätssicherung der Ergebnisse (Babbage 1833). Somit stellen die Formulare bereits die Keimform einer formalen Beschreibung eines Algo-rithmus dar (ähnlich wie ein Programm-Ablaufplan). Noch bis zum 2. Weltkrieg ist dies das Standardverfahren zur arbeitsteiligen Ausführung umfang-reicher technisch-wissenschaftlicher Berechnungen. Das Verfahren inspiriert auch Babbage zur Konzeption seiner programmierbaren mechanischen Rechenmaschine (Randell 1982).
Die nachstehende entwicklungsgeschichtliche Übersicht verzeichnet die weiteren wesentlichen logisch-mathematischen Entwicklungsschritte, die allesamt notwendige Voraussetzungen für die maschinelle Durchführung komplexer Berechnungsverfahren bzw. Algorithmen bilden. Sie liegen hauptsächlich bereits im 19. Jhdt. mit Ausnahme der letzten, die im Zusammenhang mit der Grundlagenkrise der Mathematik entstehen, zur Jahrhundert-wende ausgelöst durch die Russellsche Antinomie als einer Paradoxie der Mengenlehre und Fragen nach deren axiomatischer Fundierung. Als dominante Strategie zur Überwin-dung der Krise zeigt sich das ›Hilbert-Programm‹ einer axiomatisch begründeten durch-gängigen Formalisierung mathematischer Beweisführung, das durch strikte Trennung ei-ner rein formal mit logischen Formeln operierenden Ebene von einer ›meta-mathematischen‹ Ebene axiomatischer Begründung und intuitiv-inhaltlichen Schließens ermöglicht werden sollte (Hilbert 1922). Später aufkommende Zweifel an dessen vollstän-digen Durchführbarkeit führen in der Folge zu den überraschenden Einsichten, dass es u.a. unmöglich ist, einen Algorithmus anzugeben, der alle Sätze eines formalen Systems abzu-leiten und deren Widerspruchsfreiheit zu zeigen imstande ist (Gödel 1931) ebenso wie ei-nen Algorithmus anzugeben, der von jeder Formel eines formalen Systems entscheiden kann, ob sie ein wahrer Satz des Systems ist (Turing 1936). Damit werden letztlich theore-tische Grundsatzfragen von Berechenbarkeit und Entscheidbarkeit geklärt und der Begriff des ›Algorithmus‹ präzise definiert (Kleene 1952, Hermes 1972).
Als notwendige Bedingungen der maschinellen Durchführung von Berechnungsverfahren im Rahmen zeichenbasierter kognitiver Arbeit sind damit sämtliche logisch-

mathematischen Grundlagen der Formalisierung bereits vor deren physischer Realisierung entwickelt (vgl. nachstehende Übersicht). Dabei erweist sich die Maschinen-Hardware als der eigentlich limitierende Faktor der Entwicklung: Nachdem Babbage an den Tücken me-chanischer Repräsentation von Zahlen und arithmetischer Operationen gescheitert ist, dauert es bis zur Verfügbarkeit steuerbarer elektromechanischer (Relais) oder elektroni-scher Schalter (Röhren, Transistoren) bzw. später integrierter binärer Schaltnetze, um die bereits zu hinreichender Reife entwickelten logisch-mathematischen Verfahren tatsächlich maschinell ausführen zu können. Bis heute wird die Entwicklung der Computertechnik am allerwenigsten von logisch-konzeptionellen Neuerungen, sondern v.a. von den Steige-rungen der Leistungsfähigkeit der Schaltsysteme (wie auch von Hilfsmitteln zur Formali-sierung von Zeichenprozessen sozialer Praxis) bestimmt.

Historische Übersicht zu den theoretischen Grundlagen der Computertechnik

1792-1801	Gaspard de Prony entwickelt und nutzt ein formularbasiertes Verfahren zur extrem arbeitsteiligen Neuberechnung mathematischer Tafeln im Dezimalsystem (als Ba-sis-Werkzeug für Ingenieurarbeit); in diesem frühen großen Softwareprojekt bildet das Formularschema der Rechenoperationen die Keimform eines Algorithmus (noch im 2. WK werden z.B. Tragwerke, V2-Flugbahnen u.v.a. auf diese Weise berechnet).
1805	Jacquard-Webstuhl, erste digital gesteuerte Arbeitsmaschine (mit Lochbrettern).
1812	Charles Babbage konzipiert die »Difference Engine« zur einfachen Berechnung von Polynomen: $f(x) = an\ xn + \ldots + a1\ x + a0$ (1822 prototypisch rea-lisiert).
Um 1830	Charles Babbage entwirft und programmiert die »Analytical Engine«; sie nimmt die von-Neumann-Architektur programmierbarer Universalrechner (Prozessor - Speicher - Steuerung) vorweg (scheitert aber an der Mechanik).
1847	Die de Morganschen Gesetze [$\neg(a \wedge b) = \neg a \vee \neg b$ und $\neg(a \vee b) = \neg a \wedge \neg b$] aufgreifend publiziert George Boole einen Logikkalkül (um 1888 von G. Peano als ›Boolesche Algebra‹ axiomatisiert); er bildet das logisch-funktionale Fundament für binäre Schaltsysteme (heutige Computer-Hardware).
1860-1880	C.S. Peirce entwickelt erstmals einen Prädikatenkalkül 1. Stufe, arbeitet an »logi-schen Maschinen« und entwickelt eine triadische Zeichentheorie, ohne die der allg. Computereinsatz nicht zu verstehen ist (äquivalent: G. Freges »Be-griffsschrift« 1879; s. unten »algorithmisches Zeichen«).
1931	Kurt Gödel beweist u.a. unter Verwendung rekursiver Funktionen die Unvollstän-digkeit formaler Systeme wie das der principia mathematica von B. Russell & A.N. Whitehead.

1936	Alan Turing publiziert das ideelle Modell der ›Turingmaschine‹, definiert damit formal die Begriffe Algorithmus und berechenbare Funktion (äqui-valent: λ-Kalkül von A. Church & S.C. Kleene).
1941/45	Konrad Zuse entwickelt die Z3 als ersten binären Rechner (Relaistechnik) und den Plankalkül als erste (quasi-funktionale) Programmiersprache (markiert die Geburt des modernen Computers).

5 »Algorithmische Zeichen« und Computer als semiotische Maschinen

Nach westeuropäischem Verständnis der Fachdisziplin geht es in der »Informatik« im Kern um die »rationale, insbesondere maschinelle Verarbeitung von Information« (Académie Française; GI). Diese Definition ist in mehrfacher Hinsicht problematisch. Ers-tens existieren unter der Bezeichnung »Information« mindestens drei verschiedene, mitei-nander unvereinbare Begriffe; zur Analyse des Einsatzes von Computern verwendet bleibt somit offen, welcher jeweils tatsächlich gemeint ist:

- umgangssprachlich der Inhalt einer Nachricht, Auskunft oder Belehrung,
- im Bereich sozialer Interaktion (Praxis) jeder Unterschied, der im Handeln etwas aus-macht (»any difference that makes a difference«; Bateson 1980),
- in der technischen Signalübertragung (Hartley; Shannon) der syntaktische Informati-onsgehalt von Signalen (»Entropie«): $I = \sum pj\ ld\ (1/pj)$ (mit pj als relativen Häufigkeiten bedeutungsloser Zeichen eines endlichen Alphabets in einer Nachricht).

Zweitens bleibt damit insbesondere unbestimmt, was genau der Gegenstand der »maschinellen Verarbeitung« ist: der syntaktische Informationsgehalt von Signalen oder der »Unterschied, der etwas ausmacht«? Ebenso bleibt drittens in der praktischen Verwendung unklar, wie im Umgang mit auf Computern ausgeführten, durch Programme formal beschriebenen Berechnungsverfahren bedeutsame »Unterschiede« entstehen können. Dieser Wirrwarr macht »Information« als vermeintlich wissenschaftlichen Begriff für die Analyse computerunterstützter sozialer Praxis gänzlich unbrauchbar; so gebietet sich, nach alterna-tiven begrifflichen Zugängen zu suchen.

Dazu ist unumgänglich, sich – gestützt auf die Einsichten der theoretischen Informatik (s.o. 3) wie auch der Logik und Physik binärer Schaltsysteme – erneut grundlegender Erkenntnisse über Aufbau und Funktionsweise von Computern zu vergewissern (vgl. Brödner 1997). Laut Boolescher Algebra können sämtliche logischen Operationen mithilfe der elementaren logischen Operatoren der Konjunktion (›und‹), Disjunktion (›oder‹) und Negation verwirklicht werden. Als NAND- und NOR-Gatter physisch realisiert lassen sich mit ihnen beliebige binäre Schaltsysteme aufbauen, u.a. auch Speicherzellen sowie Halb- und Volladdierer, mit denen etwa die Nachfolgefunktion realisiert und damit wiederum beliebige Dualzahlen korrekt addiert werden können. So werden arithmetische auf physisch realisierbare logische Operationen zurückgeführt. Damit lässt sich ein minimales

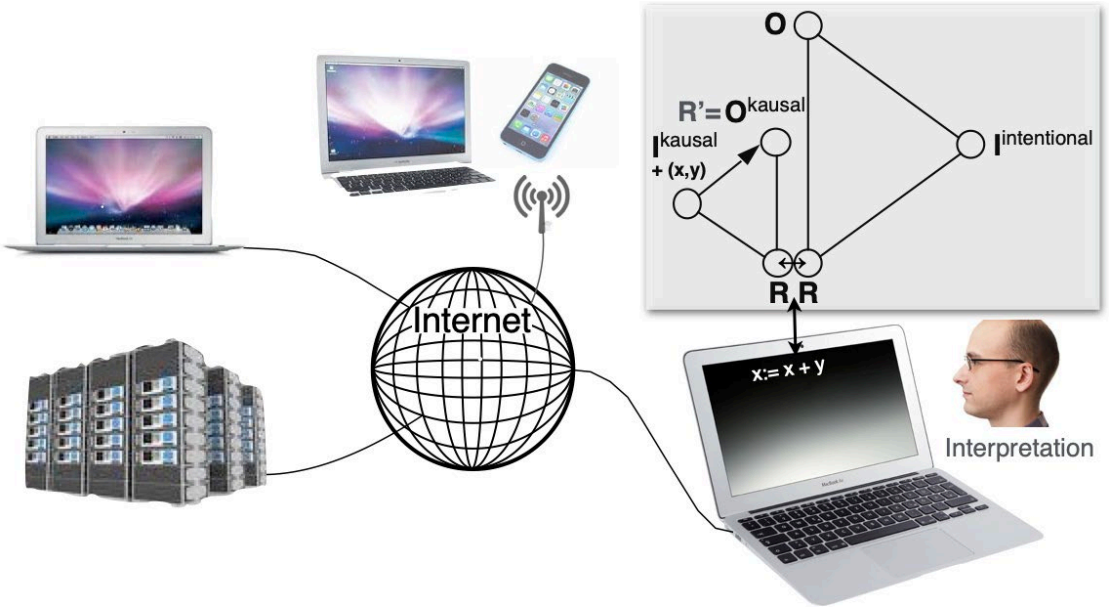
Rechnermodell angeben, das gleich mächtig wie die Turingmaschine ist (und wie diese beliebig große Daten- und Programmspeicher für natürliche Zahlen einschließlich der Null voraussetzt), als Basis für sämtliche arithmetischen Operationen mit beliebigen, letzt-lich auch negativen oder Gleitkomma-Zahlen. Die maschinelle Ausführung beliebiger Al-gorithmen benötigt diesem Modell zufolge gemäß Rekursionstheorie nur vier elementare Anweisungen für Operationen auf einer Speicherzelle: die Zuweisung der 0 (Löschung x := 0), die Nachfolgerbestimmung (x := x + 1), die binäre Komplementbildung (für negative ganze Zahlen) und das while-Programm: while … do … end.

Festzuhalten bleibt: Computer führen auf der materiellen Basis binärer Schaltsysteme im streng mathematischen Sinne berechenbare Funktionen aus – nichts sonst. Nun sind Logik und Mathematik, Hilbert zufolge, ein »Spiel mit wenigen Regeln und bedeutungslosen Zeichen auf Papier«; von den für sie geltenden Regeln abgesehen sind sie durch Abstraktion frei von Bedeutung. So operieren etwa Automaten und formale Sprachen, ebenso wie Turingmaschinen oder partiell rekursive Funktionen (als äquivalenten Modellen für Berechenbarkeit) allein mit solchen bedeutungslosen Zeichen als Elementen eines endlichen Alphabets (›Zeichenvorrat‹) und den dafür geltenden Regeln (Kleene 1952, Krämer 1988).

Folglich liegt es nahe, für die Analyse anstelle des schillernden Informationsbegriffs einen adäquaten Zeichenbegriff zur Grundlage der Analyse zu machen, der zwischen bedeutungslosem Operanden einer berechenbaren Funktion einerseits und sozialem Sinn (Bedeutung) im Rahmen einer sozialen Praxis andererseits zu vermitteln vermag; ein solcher wurde vom Logiker C.S. Peirce (deutsch: 1983) entwickelt (etwa zeitgleich mit und äquiva-lent zu G. Freges Begriffsschrift). Dabei wird unter einem Zeichen die dreistellige Relation zwischen einem beliebigen physischen Zeichen(-träger) (Repräsentamen R), dem damit bezeichneten Objekt (O) und dem Begriff (Interpretant I) verstanden, der dieser Referenz Bedeutung in einem Handlungskontext sozialer Praxis zuweist: I – (R – O). »Ein Zeichen ist etwas, das für jemanden in einer bestimmten Hinsicht oder Fähigkeit für etwas steht.« (Peirce). Mit diesem triadischen Zeichenbegriff wird die formale Welt der Mathematik, mit ihren bloß gedachten Gegenständen gesehen als ein »Spiel mit wenigen Regeln und bedeutungs-losen Zeichen auf Papier«, mit der sinnhaften Welt zeichenbasierter Interaktion sozialer Praxis verbunden. Er bildet die Klammer, die beide Welten zusammenführt.

Computer und per Programm darauf ausführbare berechenbare Funktionen werden entworfen und eingesetzt entweder zur digitalen Steuerung physischer Prozesse (als »embedded« bzw. »cyber-physical systems«) oder zur interaktiven Nutzung durch Menschen in deren sozialer Praxis (als »Informations-« bzw. »Organisationssysteme«). Die Steuerung physischer Prozesse greift in Naturprozesse ein, basiert aber auf deren aus Einsicht in die physischen Wirkungen der Prozesse gewonnenen, zeichenbasierten Beschreibung. Ein da-raus zweckgemäß entwickeltes mathematisches oder zumindest formal beschriebenes heu-ristisches Modell des jeweiligen Prozesses erlaubt dann, dafür das Programm einer digita-len Steuerung zu entwerfen, das aufgrund von relevanten Signalen aus dem Prozess ziel-führende Eingriffe in dessen Energie- oder Stoffströme

auszulösen vermag, um dessen gewünschten Verlauf automatisch zu gewährleisten. Dabei werden die algorithmischen Funktionen der Steuerung über Mess- und Stellglieder mit dem Naturprozess verknüpft.
Mit dem interaktiven Gebrauch von Computern wird hingegen modellbasiert in soziale Praktiken interveniert, wobei der Gebrauch seinerseits auf dem verständigen, sinngebenden Umgang mit Zeichen in diesen Praktiken beruht. In beiden Fällen dienen Computer der rein syntaktischen Verarbeitung bedeutungsloser Signale als Zeichenträgern mittels berechenbarer Funktionen. Dabei erlaubt der triadische Zeichenbegriff zugleich auch die genaue Analyse und Beschreibung des interaktiven Umgangs mit Computern.



Interaktion zwischen Mensch und Rechner im Kontext sozialer Praxis (Brödner 2020)

In der Interaktion mit Computern werden von Benutzern Zeichen für ›Daten‹ und damit operierende Funktionen eingegeben (add(x,y), copy(x), edtxt(…) etc.), sog. »algorithmische Zeichen« (Nake 2001), die im jeweiligen äußeren sozialen Handlungskontext bestimmte Bedeutung tragen. Mit der Eingabe werden diese außen situativ sinnvoll interpretierbaren Zeichen auf bloße Signale bzw. ›Daten‹ als deren materiellen Verkörperungen (R) redu-ziert und mittels maschinell ausführbarer Anweisungen eines Programms, die das Berech-nungsverfahren – den Algorithmus – formal beschreiben, verarbeitet (nach dem Modell der Turingmaschine, Turing 1936; vgl. Abbildung). Das mithin vollständig determinierte Resultat R' dieser Signalverarbeitung kann dann bei Erscheinen an der Systemoberfläche erneut als Zeichen im sozialen Handlungskontext interpretiert werden. Beide Zeichenpro-zesse, die interne, auf programmgesteuerte, rein syntaktische Signalverarbeitung reduzier-te wie auch die äußere sinngebende Interpretation, sind über den beiden gemeinsamen Zeichenträger R fest gekoppelt: Die interne Verarbeitung der Signale (›Daten‹) ist anstelle der äußeren intentionalen Interpretation seitens der Benutzer durch die Anweisungen des Programms als kausalem Interpretanten determiniert. Das Ergebnis R' fällt mit dem auf die-se Weise berechneten, kausal bestimmten Objekt zusammen. Eben die Kenntnis dieser Zusammenhänge ermöglicht dann außen eine sinnvolle Interpretation.

So ist Interaktion mit Computern gekennzeichnet durch kausale Determination sinn- und kontextfreier Signalverarbeitung im Innern und durch sinngebende Interpretation der an der Oberfläche als Zeichen gedeuteten Signale bzw. Daten außerhalb. Der soziale Raum der Zeichenprozesse wird dabei nicht verlassen. Der triadische Zeichenbegriff vermag mithin eine Brücke zu schlagen zwischen der physischen Welt, in der Computer algorithmisch programmgesteuert Signale (physisch) bzw. Daten (logisch) verarbeiten, mit der sozialen Welt der Signifikation, der Zuweisung von Bedeutung durch intentionale Interpretation in einem Handlungskontext sozialer Praxis. Dabei vermittelt der triadische Zeichenbegriff zwischen Signal und Sinn und ermöglicht damit – im Unterschied zu der den Kern der Sache verschleiernden Bezeichnung »maschineller Verarbeitung von Information« – eine genaue Analyse der Vorgänge beim Umgang mit vernetzten Computersystemen als einer »Infrastruktur« sozialer Interaktion (Pipek & Wulf 2009), als instrumentellem Medium der Kommunikation und Kooperation im sozialen Raum, das vielfältige Mittel zur Verarbeitung, zum Austausch und zur Aufbewahrung von Daten in sich vereint.

Epistemologisch geht es beim interaktiven Computergebrauch im Rahmen sozialer Praxis um drei grundverschiedene Typen von Wissenschaft mit je eigenen Erkenntnisweisen, die aber alle für eine genaue Analyse relevanter Vorgänge des Computereinsatzes benötigt werden (vgl. nachstehende Übersicht). Wenn folglich gebrauchstaugliche Computersysteme auf wissenschaftlich fundierter Grundlage geschaffen werden sollen, muss man sich – wegen der Einbettung der physisch nach Berechnungsverfahren Zeichen manipulierenden Systeme in soziale Praktiken zeichenbasierter Wissensarbeit – problemorientiert auf alle diese Wissensdomänen zugleich abstützen.

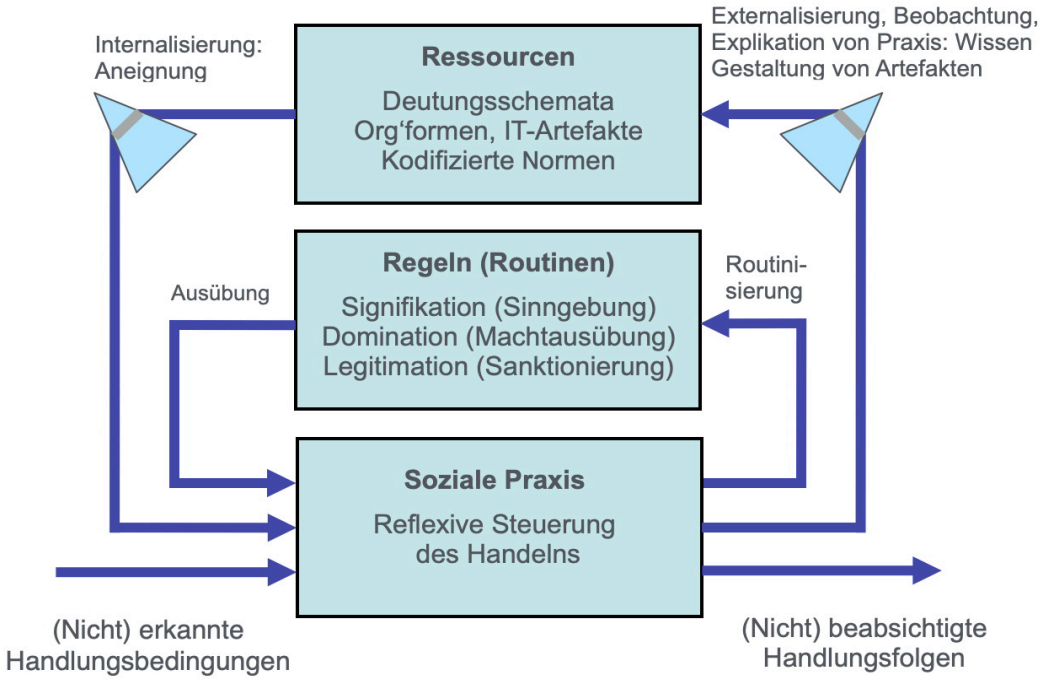
Ontologie: Sphäre des Seins	**Epistemologie**: Wissen (Theorien) über das Sein
Natur Innere Natur (eigener Körper), äußere Natur (natürliche Umgebung) Kausalität: *causa efficiens* Kräfte und Wirkungen	**Naturwissenschaft**(en) Erkenntnissicherung: Experiment Beobachtung (erkennendes Bezeichnen); Nutzung von Naturkräften mittels Werkzeugen & energie-/stoffwandelnden Maschinen Nachdenken über Natur
Soziale Praxis Gemeinschaftlich handelnde Individuen Intentionalität: *causa finalis* Sprachlich konstituierte Tatsachen, Bedeutungen (Zeichengebrauch)	**Sozialwissenschaft**(en) Erkenntnissicherung: Praxistest Beobachtung (erschließende Konstitution von Bedeutung); zeichenvermittelte soziale Interaktion & semiotische Maschinen Nachdenken über Handeln
Mathematik Mittels Zeichenkörpern & Regeln gedachte Objekte & Relationen (›Sei *x* ein…‹)	**Logik** Erkenntnissicherung: Beweis Geregeltes Spiel mit bedeutungslosen Zeichen Nachdenken über Denken

Übersicht: Ontologie und Epistemologie der Computertechnik (eigene Darstellung)

6 Computereinsatz setzt Modellierung und Formalisierung voraus

Vor dem Hintergrund dieser Einsichten in Entstehung, Besonderheiten und grundsätzliche Funktionsweise von Computertechnik ist das besondere Anliegen und Kennzeichen der Siegener Perspektive auf Sozio-Informatik, Grundlagen zur analytischen Durchdringung und Lösung realer gesellschaftlicher Probleme sozialer Praxis zu erforschen (»basic research of real world problems«). Dabei orientiert sich diese Forschung der Sache angemessen an praxistheoretischen Zugängen und Erkenntnissen (vgl. die Übersicht bei Reckwitz 2003), vorzugsweise mithilfe der Strukturationstheorie sozialer Praxis von Giddens (1988; zu deren praktischer Nutzbarkeit vgl. z.B. Orlikowski 2000).

Aus dieser strukurationstheoretischen Sicht und der bisherigen Darstellung geht u.a. hervor, dass mit dem Einsatz von Computertechnik in Zeichenprozessen sozialer Interaktion massiv in soziale Praktiken organisierter, stets zeichenbasierter und kooperativer Wissensarbeit interveniert wird. Irritationen oder enttäuschte Erwartungen im Zusammenhang mit der Ausübung eingeübter Praktiken geben zunächst Anlass, diese mittels Beobachtung und begrifflicher Explikation, v.a. der innewohnenden impliziten Regeln und Routinen der Handlungssteuerung, zu analysieren und so zunächst kodifiziertes Wissen über diese zu generieren. Auf dieser Basis können involvierte Akteure – stets zweckorientiert und interessengeleitet – die zum Einsatz nutzbarer Softwareartefakte führende Beobachtung, Modellbildung und Formalisierung vorantreiben und letztlich als im Handlungskontext gebrauchstaugliches Computersystem implementieren. Um dessen vielseitige Funktionen aber tatsächlich praktisch wirksam nutzen zu können, müssen diese erst durch die Nutzer mühsam angeeignet werden (»making sense of artifacts«), bevor sie durch Einübung routiniert verwendet werden können (vgl. nachstehende Abb.).



Praxistheorie: Rekursive Konstitution von Handeln und Struktur
(eigene Darstellung angelehnt an Giddens 1988)

Aus dieser Sicht ergibt sich als wichtige Folgerung, dass die Tätigkeiten, die im Kreislauf von Beobachtung, Modellierung, Formalisierung, Implementation berechenbarer Funktio-nen im Computersystem und deren Aneignung zu praktisch wirksamen Gebrauch vollzo-gen werden, insgesamt die organisationalen Praktiken stark verändern. Dabei ist insbe-sondere zu reflektieren, dass sowohl Modellbildung und Funktionsimplementierung im Computersystem als auch deren Aneignung für die Nutzung jeweils kreative, zweckorien-tierte und interessengeleitete Tätigkeiten sind, deren Verlauf und Ergebnis nicht vorherzu-sehen sind. Somit ist dieser Kreislauf in hohem Maße selbstbezüglich, indem er maschinel-le Funktionen für eine Praxis zu schaffen anstrebt, die am Ende eine andere ist als die, für die sie konzipiert wurden. Soziale Praktiken der Wissensarbeit als Gegenstand der Model-lierung geraten durch den Vorgang des Modellierens in Bewegung. Damit erweist sich der Entwicklungsprozess der Software zugleich auch als Prozess der Organisationsentwick-lung: Software ist ›Orgware‹, ein Medium des Organisierens.

Aufgrund dieser Einsichten stellen sich bezüglich der Nutzung von Computertechnik zur Lösung realer gesellschaftlicher Probleme die Kernfragen: Wie gelangt man von der Analyse von Naturprozessen bzw. der Beobachtung schwer zu durchschauender sozialer Praxis mit ihren Mustern und Routinen zu zweckerfüllenden berechenbaren Funktionen, die allein auf binären Schaltsystemen ausführbar sind? Wie sind diese zu gestalten und wie lassen sie sich aneignen, um Aufgaben der Praxis effektiver und effizienter zu bewältigen? Gestaltung und Einsatz interaktiv genutzter Computer erfordern eine schrittweise, stets nur partiell mögliche Reduktion durch abstrahierende Modellierung und Formalisierung bestimmter Vorgänge oder Aspekte sozialer Praktiken mit meist implizitem Können und eingeübten Routinen (bei Steuerungen ist die Modellierung der zu steuernden Naturprozesse gefordert). Dazu müssen der Praxis zugrunde liegende Zeichenprozesse zunächst in bestimmter Perspektive modelliert und beschrieben werden durch deren partielle Explikation in Gestalt begrifflich-propositionalen Wissens über deren Strukturen und Abläufe. Die Modellbildung – Kern des Software Engineering – erfordert Können, Intuition und natürliche Intelligenz und durchläuft im Prinzip folgende Schritte der Reduktion, Abstraktion und Formalisierung (Andelfinger 1997, Krämer 1988):

- Semiotisierung: Präzise Beschreibung relevanter Aspekte einer sozialen Praxis mittels Zeichen liefert ein perspektivisch reduziertes Abbild von Wirklichkeit als Ergebnis ge-meinsamer Reflexion und Kommunikation der Akteure (Sprachanalyse, ›Ontologie‹). Ergebnis: das Anwendungsmodell.
- Formalisierung: Abstraktion von situations- und kontextgebundenen Bedeutungen und Reduktion auf bedeutungslose Standardzeichen und -operationen. Ergebnis: ein formales Modell.
- Algorithmisierung: Überführung von Gegenständen und Abläufen des formalen Mo-dells in auto-operational ausführbare Prozeduren in Form von Daten und berechenba-ren Funktionen (Algorithmen). Ergebnis: das Berechnungsmodell (in Form eines Pro-gramms).

Dabei sind stets zwei grundsätzlich nicht überwindbare Grenzen zu bedenken: erstens die mathematisch-logisch bedingte Grenze der Berechenbarkeit und zweitens die Grenze der

Explikation impliziten Könnens, die sich freilich aufgrund der Dynamik von Explikation in kodifiziertes Wissen und dessen Aneignung zu erweitertem Können ständig verschiebt. Somit können auch Vorgänge sozialer Praktiken kooperativer Wissensarbeit (»kognitive Funktionen«) stets nur partiell sprachlich gefasst, formalisiert, in ein Berechnungsmodell überführt und per Programm maschinell ausgeführt werden.

Die physische Ausführung des Berechnungsmodells – des Algorithmus – durch ein Schalt-system stellt einen »degenerierten«, auf eine dyadische Relation reduzierten Zeichenpro-zess ohne ›Fenster zur Welt‹ dar, dem der Bezug zu einem erlebten, leiblich erfahrenen oder gedachten Objekt, eben die ›Be-zeichnung‹, fehlt. Er ist nur eine »Quasi-Semiose«, die mit Signalen (auf der physischen Ebene) bzw. Daten (auf der logischen und algorithmi-schen Ebene) als auf Syntax reduzierten »Quasi-Zeichen« operiert (Nöth 2002). Deren Zu-standsänderungen werden durch den Algorithmus gesteuert ohne Ansehen ihrer Bedeu-tung. Im Computersystem implementiert entstehen damit »auto-operationale Formen« (Floyd 2002) als Ausdruck abstrakter, formalisierter und automatisch ausgeführter Opera-tionen (vgl. oben 3). Deren Sinn muss durch Aneignung seitens der Benutzer für den prak-tisch wirksamen Gebrauch erst noch erschlossen werden. So werden Computersysteme durch Interpretation ihrer Funktionen im Handlungskontext einer sozialen Praxis wieder in einen – freilich eben dadurch veränderten – Praxiszusammenhang gestellt. Auf diese Weise erweisen sich die Gestaltung und Aneignung von Computersystemen als massive, allerdings hochgradig selbstbezügliche Intervention in soziale Praktiken.

Vor diesem Hintergrund ist die klare Unterscheidung zwischen Wirklichkeit (als einem beobachtbarem Wirkungs- bzw. Bedeutungszusammenhang) und Modell (als deren durch Abstraktion gewonnenem Abbild oder Beschreibung) wesentlich. Dies umso mehr, als oftmals Modelle gedankenlos für die Wirklichkeit selbst gehalten werden und zu folgen-reichen Irrtümern und Selbsttäuschungen führen:

	Abstraktion	
Wirklichkeit	——————>	**Modell**
Landschaft		Landkarte
Mahlzeit		Menü
Gegenstand		Bezeichnung
Sachverhalt		Beschreibung
Maschinelle Berechnung		Turingmaschine
Technik		Technologie
(Funktionen, Verfahren)		(Lehre über Technik)

7 Softwaretechnik reduziert artikulatorische Distanz ohne sie zu beseitigen

Mit Softwaretechnik werden Vorgehensweisen, Hilfsmittel und Methoden geschaffen, um Probleme der Reduktion, die klaffende epistemische und artikulatorische Lücke zwischen begrifflich-propositionalen Beschreibungen sozialer Praktiken zeichenbasierter Wissens-

arbeit einerseits und der Konstruktion zweckmäßiger Berechnungsverfahren (Algorithmen) zur gezielten Verbesserung der Praxis andererseits, leichter zu überbrücken. Sie sollen kognitiver Tätigkeiten der Modellierung unterstützen, die zur Überwindung dieser großen artikulatorischen Distanz zwischen der Ebene sprachlicher Beschreibung von Strukturen und Abläufen einer gewünschten Praxis und der Programm-Ebene der formalen Darstellung des entsprechenden Berechnungsmodells notwendig sind.

Dabei wird in der Computertechnik häufig ein Aufbau- und Konstruktionsprinzip genutzt, das aus eindeutig bestimmten und vielfach erprobten einfachen Berechnungsfunktionen komplexere und reichhaltigere Funktionen zu bilden erlaubt (z.B. bei der rekursiven Funktionsdefinition, für Suchen, Sortieren, Bildverarbeitung oder Datentransfer u.v.a. mehr). Im Zuge der weiteren Entwicklung werden dabei speziell für jeweils bestimmte Aufgaben und Funktionen geschaffene Softwaremodule derart in aufeinander aufbauenden Schichten angeordnet, dass Module oder Funktionen einer höheren Schicht nur solche einer tieferen Schicht aufrufen und deren zurückgelieferte Ergebnisse nutzen können. Auf diese Weise auf die Schichten verteilt lassen sich auch sehr komplexe Berechnungsverfahren und Programmabläufe einigermaßen transparent und nachvollziehbar organisieren, von direkt hardwarebezogener Mikroprogrammierung über Compiler und Betriebssysteme zu aufgabenangemessenen Anwendungs- und Präsentationsfunktionen – ggf. auch über mehrere Rechner verteilt (»Distributed Computing«). Folglich beruhen auch die komplexesten Programme und Softwaresysteme letztlich auf unterster Ebene stets auf der Ausführung einfacher Elementarfunktionen, allerdings operativ ausgeführt auf immer leistungsfähigeren Schaltsystemen, inzwischen in Milliarden Takten pro Sekunde.

Softwaretechnik unterstützt mithin die schwierigen Vorgänge der Modellierung und Formalisierung von Aspekten sozialer Praxis (bzw. von Naturprozessen) gewissermaßen von unten, von den Bedingungen maschinell ausführbarer Algorithmen aus in Richtung aufgabenbezogener Funktionen in dem Bestreben, durch Analysen sozialer Praktiken gewonnene, sprachlich artikulierte Anforderungen mittels »operationaler Formen« unterschiedli-cher Abstraktion leichter umsetzen zu können (Floyd & Klaeren 1999).

Den Anfang machen problemorientierte Programmiersprachen, die in diesem Sinne die Umsetzung funktionaler Spezifikationen in maschinell ausführbare Programme vereinfachen:

- angelehnt an den Lambda-Kalkül entsteht 1960 die funktionale Programmiersprache LISP (McCarthy 1960), die später weitere dieser Art nach sich zieht, oder
- zur gleichen Zeit wird orientiert am Rechnermodell der Turingmaschine ALGOL 60 (Backus et al. 1960) als erste imperative Programmiersprache und Stammvater vieler Nachfolger wie Pascal oder Java geschaffen.

Mit der raschen Zunahme von Umfang und Vielfalt der Software-Artefakte wird neben der Überwindung der artikulatorischen Distanz auch die Beherrschung der Komplexität von Software immer wichtiger. Dazu werden mit der Entwicklung abstrakter Datentypen und Methoden objektorientierter Analyse, Entwurf und Programmierung bedeutende Hilfsmittel und Methoden zur Modularisierung komplexer Software geschaffen. Sie bilden eine wichtige Voraussetzung dafür, komplexe Softwaresysteme wie oben skizziert in übereinander geschichtete und damit in ihrem Zusammenwirken überschaubarere

Funkti-onseinheiten zu gliedern (»Software-Architektur«). Damit verbunden entsteht mit UML eine weithin verwendbare Methode der Modelldarstellung (Booch 1993, Jacobson et al. 1999). Zudem führen Bemühungen um Weiter- und Wiederverwendung von erprobten Software-Modulen zu umfangreichen Programmbibliotheken und Software-Frameworks.

Andere Anstrengungen gelten der Verringerung der artikulatorischen Distanz bei jeweils bestimmten Klassen von Aufgaben, indem für die dabei verwendeten Berechnungsverfahren aufgabenangemessene und anpassbare Präsentations- und Handhabungsfunktionen, sog. Benutzungsoberflächen, eingeführt werden, die ihrerseits automatisch ausführbar sind, gleichwohl Nutzern Möglichkeiten bieten können, Funktionen begrenzt abzuwandeln (sog. »End User Design«; vgl. Fischer 2012), so etwa für FEM und CAD, für werkstattorientierte NC-Programmierung, für Textverarbeitung und Präsentationssoftware. Auf leichtere Zugänglichkeit von Softwareartefakten zur rechnerunterstützten Aufgabenbewältigung im gewohnten Arbeitskontext richten sich auch Bemühungen um die einfachere instrumentelle und mediale Nutzung vernetzter Computersysteme als einem Medium der Kommunikation und Kooperation (HCI & CSCW).

Allerdings haben einerseits all diese zweifellos hilfreichen Anstrengungen der Softwaretechnik zu systematisch-methodischen Vorgehensweisen beträchtliche Rebound-Effekte zur Folge: Je zuverlässiger und wirksamer sich diese Methoden und Hilfsmittel zeigen, desto größere und komplexere Projekte mit wiederum erhöhter Intransparenz werden anzugehen gewagt. Andererseits ändert das an dem in der Softwaretechnik leider gewohnten, verglichen mit anderen Technikfeldern extrem hohen Anteil an Projekten wenig, die entweder vollständig oder zumindest teilweise scheitern, indem sie ihre Zeit- und Geldbudgets bei weitem überziehen, ohne den geplanten Funktionsumfang je zu erreichen. Die Gründe für diese persistente Softwarekrise sind hauptsächlich in der trotz aller Bemühungen weiter bestehenden Lücke zwischen den stets erforderlichen formalen Modellen und der sprachlichen Beschreibung sozialer Praktiken zu suchen, für die ein Computersystem geschaffen werden soll. Sie kann eben prinzipiell nicht mit formalen Mitteln, sondern letzt-lich, auf welcher Ebene der Modellierung auch immer, nur durch intuitiv kreative Akte lebendigen Arbeitsvermögens überwunden werden, mittels derer Einsichten in soziale Praktiken gewonnen und passende Software-Module versuchsweise geschaffen und er-probt werden. Sie ist und bleibt methodisch-formalisiertem Zugriff entzogen (das zeigt sich bereits bei der Formalisierung der Mathematik im Zusammenspiel von operationaler und meta-mathematischer Ebene, s.o. 3).

Zudem sind an umfangreicheren softwaretechnischen Projekten mehrere Akteure mit unterschiedlichen Fachkompetenzen, Sichtweisen und Interessen beteiligt. Daher erfordert die Artikulation funktionaler Anforderungen und deren Umsetzung in formale Berechnungsmodelle diskursive Prozesse der Verständigung auf ein hinreichend geteiltes Verständnis der in Rede stehenden sozialen Praxis sowie der Ziele und Methoden ihrer Veränderung. Diesen praxistheoretischen Einsichten entsprechen auch die in Siegen entwickelten Konzepte und Vorgehensweisen der Sozio-Informatik und des Grounded Design (Rohde et al. 2017; Wulf et al. 2018).

Allerdings erscheint es aus mehreren Gründen problematisch, sich zur Erklärung der notwendigen Verständigungsprozesse bei Kommunikation und Kooperation in der Wissensarbeit, insbesondere bei »Kooperation ohne Konsens«, des relativ vagen und ad hoc eingeführten Konzepts der »Boundary Objects« (Star 2010) zu bedienen. Abgesehen davon, dass dabei durch »Framing« unangemessene Vergleiche zu sog. »verteilten KI-Systemen« gezogen werden, durch die sich im Kontext sozialer Praktiken vollziehende Interpretatio-nen unzulässig auf formale Schluss-Schemata reduziert werden, stützt sich dieses Konzept ebenfalls wesentlich auf den noch weniger klaren Informationsbegriff als einem semanti-schen Chamäleon ab (s. oben 4). Stattdessen wäre es nach den bisherigen Ausführungen doch leicht und einfach möglich, sich bei der Analyse und Erklärung des Problems der »Kooperation ohne Konsens« ebenfalls auf den triadischen Zeichenbegriff und den darauf aufbauenden wohl elaborierten und etablierten Begriffsapparat (formaler) Sprachen mit endlichen Alphabeten und dafür geltenden Regeln abzustützen. Verständigungsprozesse wären dann ebenso wie bei der durch Computernetze medial vermittelten Kommunikati-on und Kooperation dadurch einsichtig zu bewältigen, dass alle beteiligten Akteure ihre Sicht der Dinge in einer aufgrund von Vereinbarungen geteilten, mehr oder weniger for-malen Sprache der Modellbildung auszudrücken gefordert sind und die zu interpretieren sie sich durch Aneignung der dafür geltenden Regeln befähigen müssen (zur Empirie hierzu vgl. Brödner 2010).

8 Evolutionäres und partizipatives Projektmanagement

Diese Einsichten aus einer praxistheoretische Perspektive auf Gestaltung und Gebrauch von Computertechnik in Zeichenprozessen sozialer Praxis haben auch hinsichtlich der Vorgehensweise beträchtliche Konsequenzen für das Projektmanagement. Erstens ist zu berücksichtigen, dass mit der Modellierung und Formalisierung sozialer Praxis, d.h. mit ihrer Beobachtung, der sozialen Aushandlung funktionaler Anforderungen und darauf fussender Gestaltung und Aneignung der im Computersystem implementierten Funktionen massiv in die soziale Praxis interveniert wird. Das bedeutet, dass der gesamte Prozess selbstbezüglich ist: der Gegenstand der Beobachtung und Modellierung wird durch den Vorgang des Beobachtens und Modellierens verändert und gerät in Bewegung. Dem muss methodisch Rechnung getragen werden. Zweitens ist zu beachten, dass sich die Gebrauchstauglichkeit und zweckgemäße Wirksamkeit der gestalteten Funktionen nur im tatsächlichen Gebrauch, in einem Praxistest eines nutzbaren Systems, beurteilen lassen.
Aus beiden ärgerlichen, aber nicht umgehbaren Tatsachen ergibt sich, dass Gestaltung, Einführung oder auch Anpassung eines Computersystems nur gelingen können, wenn sie als integraler Teil eines umfassenderen Prozesses der Organisationsentwicklung verstanden und organisiert werden. Bislang noch zu oft praktizierte Vorgehensweisen, die sich auf eine einmalige, umfassende Anforderungsanalyse mit nachfolgenden Entwicklungs-, Test- und Einführungsphasen gründen, sind dabei von vornherein zum Scheitern verurteilt. Vielmehr macht die Tatsache der sozialen Einbettung semiotischer Maschinen ein reflexiv angelegtes, evolutionäres Vorgehen mit kurzen, überschaubaren

Revisionsschleifen unumgänglich, in dem wiederholt Schleifen mit Schritten der Anforderungsanalyse, Sys-temgestaltung, Implementation, Erprobung und formativen Evaluation erreichter Resulta-te wiederholt durchlaufen werden.
In einer solchen Entwicklungsspirale werden die Systemfunktionalität und deren Aneignung im Rahmen restrukturierter Prozesse in jeweils kleinen, bewusst begrenzten Schritten hervorgebracht und so den Nutzern wiederholt Gelegenheit zur Bewertung und Einflussnahme geboten. Die einzelnen Lernschleifen bleiben so hinsichtlich Anforderungen und Entwicklungsaufgaben überschaubar und halten Risiken des Scheiterns in Grenzen. Bewährte Methoden des Softwareengineering sind dabei notwendiger Bestandteil des Vorgehens, reichen aber allein bei weitem nicht hin. Vielmehr müssen sich die Beteiligten dabei insgesamt über alle Aspekte ihrer im Entstehen begriffenen neuen sozialen Struktur verständigen, was insbesondere die produktive Verbindung verschiedener Sichtweisen, die Bewältigung von Konflikten und den Ausgleich unterschiedlicher Interessen einschließt. Gestützt auch auf Erfahrungen aus der Aktionsforschung (Baskerville & Wood-Harper 1996), der ähnliche grundsätzliche Einsichten zugrund liegen, sind diese Überlegungen in das in Siegen entwickelte Forschungskonzept des Grounded Design (Rohde et al. 2017) eingeflossen.

9 Fazit: Was ist (Sozio-)Informatik in praxistheoretischer Perspektive?

Vor dem Hintergrund dieser Ausführungen wird eine über das derzeit (noch) vorherrschende Selbstverständnis der Fachdisziplin hinausgehende, praxistheoretische Position vertreten: Informatik ist eine Ingenieurdisziplin, die technische Artefakte zur zweckmäßigen und gebrauchstauglichen maschinellen Verarbeitung von Signalen (›Daten‹) – nicht von ›Information‹ – in Zeichenprozessen sozialer Praxis analysiert, gestaltet und bewertet.
Von früheren Ingenieurdisziplinen (Hoch- & Tiefbau, Maschinenbau, Elektrotechnik, chemische oder biologische Verfahrenstechnik) unterscheidet sie sich grundlegend dadurch, dass nicht allein Naturprozesse, sondern vor allem auch soziale Interaktionsprozesse analysiert, modelliert und bewertet werden. Diese Besonderheit erfordert auch besondere Vorgehensweisen und Methoden.
Ein Kennzeichen der Praxis sozialer Interaktion ist es, dass es darin ›Objektivität‹ nur insoweit gibt, als die Gemeinschaft der beteiligten sprach- und handlungsfähigen Akteure sie als gemeinsame Wirklichkeit erlebt. Notwendige Bedingung dafür ist, dass sich kommunikativ handelnde Akteure miteinander über das verständigen, was in der geteilten Wirklichkeit ihrer sozialen Praxis geschieht und in ihr bewirkt werden soll. Als im Rahmen einer sozialen Praxis mittels Zeichen kommunikativ handelnde Akteure müssen Gestalter und Anwender wie Nutzer dabei verwendeter Computer als technischen Artefakten maschineller Datenverarbeitung folglich eine gemeinsame Wirklichkeit mit geteilten Interpretationsschemata für ihre Gegenstände und die Art und Weise, sie zu gebrauchen, entwickeln.

Window	Central Concern	Principal Stories
Computation	What can be computed; limits of computing.	Algorithm, control structures, data structures, automata, languages, Turing machines, universal computers, Turing complexity, Chaitin complexity, self-reference, predicate logic, approximations, heuristics, non-computability, translations, physical realizations.
Communication	Sending messages from one point to another.	Data transmission, Shannon entropy, encoding to medium, channel capacity, noise suppression, file compression, cryptography, reconfigurable packet networks, end-to-end error checking.
Coordination	Multiple entities cooperating toward a single result.	Human-to-human (action loops, workflows as supported by communicating computers), human-computer (interface, input, output, response time); computer-computer (synchronizations, races, deadlock, serializability, atomic actions).
Automation	Performing cognitive tasks by computer.	Simulation of cognitive tasks, philosophical distinctions about automation, expertise and expert systems, enhancement of intelligence, Turing tests, machine learning and recognition, bionics.
Recollection	Storing and retrieving information.	Hierarchies of storage, locality of reference, caching, address space and mapping, naming, sharing, thrashing, searching, retrieval by name, retrieval by content.

Great Principles of Computing (Denning 2003)

Zum Kernbestand informatischer Bildung muss daher neben der Befähigung zur gebrauchstauglichen Gestaltung und Bewertung datenverarbeitender Artefakte auch die Befähigung zu problemzentrierter transdisziplinärer Interaktion und zum systematischen Dialog mit Anwendern und Nutzern der Computer-Artefakte gehören. Diese Perspektive hat sich auch die Siegener Forschungstradition zur Sozio-Informatik schon früh zu eigen gemacht (Rohde et al. 2017; Wulf et al. 2018). Ihre praktische Bewährung in einer Vielzahl von Projekten spricht dafür, sie in konzentrierter Form fortzuführen.

Dieser Sicht entsprechen auch die nachstehenden »Principles of Computing«, die letztlich alle auf maschineller Ausführung berechenbarer Funktionen beruhen. Sie kennzeichnen die technischen Grundfunktionen, mithilfe derer Restrukturierungen von Zeichenprozessen sozialer Praxis in der Gesellschaft, in Produktionsbetrieben und Verwaltungen vorangetrieben werden können.

10 Literatur

Andelfinger, U. (1997): Diskursive Anforderungsanalyse. Ein Beitrag zum Reduktionsproblem bei Systementwicklungen in der Informatik, Frankfurt/M: Peter Lang

Arendt, H. (1956): Vita Activa, Stuttgart: Kohlhammer

Babbage, C. (1833): Die Ökonomie der Maschine, Nachdruck der Originalübersetzung von The Economy of Machinery and Manufacture, hg. von P. Brödner, Berlin: Kulturverlag Kadmos

Babbage, C.. (1837): On the Mathematical Powers of the Calculating Engine, in: Randell 1982

Backus, J.W.; Bauer, F.L.; Green, J.; Katz, C.; McCarthy, J.; Perlis, A.J.; Rutishauser, H.; Samelson, K.;
Vauquois, B.; Wegstein, J.H.; Wijngaarden, A. van & Woodger, M. (1960): Report on the Algo-rithmic Language ALGOL 60, CACM 3 (5), 299-314

Bateson, G., 1980: Mind and Nature. A Necessary Unity, Toronto: Bantam Books

Baskerville, R.L & Wood-Harper, A.T. (1996): A critical perspective on action research as a meth-od for information systems research, Journal of Information Technology 11, 235-246

Bernays, E. (1928): Propaganda, New York: Liveright Publishing Corp.

Booch, G. (1993): Object-Oriented Analysis and Design, London: Prentice Hall (3rd. edition 2007)

Brödner, P. (2020): Das Produktivitätsparadoxon der Computertechnik, in: H.J. Bontrup & J. Daub (Hg.): Digitalisierung und Technik – Fortschritt oder Fluch? Perspektiven der Produktiv-kraftentwicklung im modernen Kapitalismus, Köln: PapyRossa, 114-144

Brödner, P., (2010): Wissensteilung und Wissenstransformation, in: Moldaschl, M. & Stehr, N. (Hg.): Wissensökonomie und Innovation. Beiträge zur Ökonomie der Wissensgesellschaft, Mar-burg: Metropolis, 455-480

Brödner, P. (1997): Der überlistete Odysseus. Über das zerrüttete Verhältnis von Menschen und Maschinen, Berlin: edition sigma

Coy, W.; Nake, F.; Pflüger, J.-M.; Rolf, A.; Seetzen, J.; Siefkes, D.; Stransfeld, R. (Hg.) (1992): Sicht-weisen der Informatik, Braunschweig Wiesbaden: Vieweg

Denning, P.J. (2003): Great Principles of Computing, CACM 44 (11), 15-20

Fischer, G. (2012): End User Development and Meta-Design: Foundations for Cultures of Partici-pation, in: Dwivedi, A. & Clarke, S. (ed.): End-User Computing, Development, and Software En-gineering. New Challenges, IGI Global, 202-226

Floyd, C. (2002): Developing and Embedding Autooperational Form, in: Dittrich, Y.; Floyd, C.; Klischewski, R. (Eds.): Social Thinking – Software Practice, Cambridge (MA): MIT Press, 5-28

Floyd, C. & Klaeren, H. (1999): Informatik als Praxis und Wissenschaft, Tübinger Studientexte Informatik und Gesellschaft, hg. von Sylvia Rizvi & Herbert Klaeren, Universität Tübingen

Giddens, A., 1988: Die Konstitution der Gesellschaft. Grundzüge einer Theorie der Strukturie-rung, Frankfurt/M: Campus

Gödel, K. (1931): Über formal unentscheidbare Sätze der principia mathematica und verwandter Systeme I, Monatshefte für Mathematik und Physik 38 (1), 173-198

Hausdorff, F. (1897): Sant' Ilario. Gedanken aus der Landschaft Zarathustras (unter dem Pseudo-nym Paul Mongré), zitiert in Semiosis 19, Int. Zeitschrift für Semiotik und Ästhetik 5 (3), 69

Hermes, H. (1972): Aufzählbarkeit, Entscheidbarkeit, Berechenbarkeit, 2. Aufl., Berlin u.a.: Sprin-ger

Hilbert, D. (1922): Die logischen Grundlagen der Mathematik, Mathematische Annalen 88 (1-2), 151-156

Jacobson, I.; Booch, G. & Rumbaugh, J. (1999): The Unified Software Development Process. UML., Reading (MA): Addison-Wesley

Kleene, S.C. (1952): Introduction to Metamathematics, Amsterdam: North Holland

Krämer, S. (1988): Symbolische Maschinen. Die Idee der Formalisierung in geschichtlichem Ab-riss, Darmstadt: Wissenschaftliche Buchgesellschaft

McCarthy, J. (1960): Recursive Functions of Symbolic Expressions and Their Computation by Ma-chine, Part I, CACM 3 (4), 184-195

Nake, F. (2001): Das algorithmische Zeichen, in: Bauknecht, W.; Brauer, W.; Mück, T. (Hg.): In-formatik 2001. Tagungsband der GI/OCG Jahrestagung, 736-742

Nöth, W. (2002): Semiotic Machines, Cybernetics and Human Knowing 9 (1), 5-22

Orlikowski, W.J., 2000: Using Technology and Constituting Structures: A Practice Lens for Studying Technology in Organizations, Organization Science 11 (4), 404-428

Peirce, C.S. (1983): Phänomen und Logik der Zeichen, Frankfurt/M: Suhrkamp

Pipek, V. & Wulf, V. (2009): Infrastructuring: Toward an Integrated Perspective on the Design and Use of Information Technology, JAIS 10, Special Issue, 447-473

Randell, B. (Hg.) (1982): The Origins of Digital Computers: Selected Papers, 3. Aufl., Berlin Hei-delberg: Springer

Reckwitz, A. (2003): Grundelemente einer Theorie sozialer Praktiken. Eine sozialtheoretische Perspektive, Zeitschrift für Soziologie 32 (4), 282-301

Rohde, M.; Brödner, P.; Stevens, G.; Betz, M. & Wulf, V. (2017): Grounded Design – a Praxeologi-cal IS Research Perspective, Journal of Information Technology 32 (2), 163-179

Ropohl, G. (1991): Technologische Aufklärung. Beiträge zur Technikphilosophie, Frankfurt/ M: Suhrkamp

Star, S.L. (2010): This is Not a Boundary Object: Reflections on the Origin of a Con-cept«, in: Science, Technology, & Human Values 35 (5), 601–617

Taylor, C. (2017): Das sprachbegabte Tier. Grundzüge des menschlichen Sprachvermögens, Ber-lin: Suhrkamp

Thaler, R.H. & Sunstein, C.R. (2010): Nudge. Wie man kluge Entscheidungen anstößt, Berlin: Ull-stein

Turing, A.M. (1936): On Computable Numbers. With an Application to the Entscheidungsprob-lem, Journal of Symbolic Logic, 230-265

VDI (Hg.), 1991: Technikbewertung – Begriffe und Grundlagen. Erläuterungen und Hinweise zur VDI-Richtlinie 3780, VDI Report 15, Düsseldorf: VDI

Wulf, V.; Pipek, V.; Randall, D.; Rohde, M.; Schmidt, K. & Stevens, G. (2018): Socio-Informatics. A Practice-Based Perspective on the Design and Use of IT Artifacts, Oxford: Oxford University Press