Siegfried Wendt
Universität Kaiserslautern

Formalismen und Anschauung

1. Das Interesse an Formalismen

Die Systeme der Informatik sind heute schon sehr komplex und werden in Zukunft noch sehr viel komplexer sein. Diese Systeme entstehen durch Entwurfsentscheidungen von Menschen, und Komplexitätsbeherrschung liegt nur vor, wenn alle Entwurfsentscheidungen begründet und akzeptiert werden können. Dabei sind es wieder Menschen, denen gegenüber etwas begründet werden muß und die etwas akzeptieren können. Deshalb hängt Komplexitätsbeherrschung sehr stark von der Fähigkeit von Menschen ab, effizient miteinander zu kommunizieren.

Jedes Informatiksystem ist ein System zur Abwicklung von Formalismen, die einem bestimmten Zweck dienen. Dabei ist ein Formalismus eine Menge von Vorschriften zur Transformation von Formen. Aus einer vorgegebenen Form werden nach vorgegebenen Regeln neue Formen erzeugt, wobei die Anwendung der Regeln keine Zuordnung von Bedeutungen zu den Formen erfordert. Die Wahl der Formen und der Transformationsregeln trifft der Mensch im Hinblick auf den beabsichtigten Zweck.

Die Menschen, die sich beruflich mit Formalismen befassen, lassen sich grob in zwei Klassen einteilen je nachdem, ob ihr primäres Interesse dem Formalismus oder dem Zweck gilt. Man kann die beiden Klassen überspitzt wie folgt charakterisieren: In der einen Klasse befinden sich all diejenigen, die zuerst einen Formalismus erfinden und analysieren und danach erst nach einer nützlichen Anwendung dafür suchen; in der anderen Klasse findet man all diejenigen, die vor einem praktischen Problem stehen und dafür einen nützlichen Formalismus suchen. Bei den Mitgliedern der ersten Klasse dominiert das mathematische Interesse, wogegen bei den Mitgliedern der zweiten Klasse das Ingenieursinteresse dominiert.

Bei der Behandlung komplexer Informatiksysteme muß man selbstverständlich mit der Betrachtung des Zwecks beginnen und im Laufe der Betrachtung zu den Formalismen übergehen. Eine Überbewertung der Formalismen liegt vor, wenn bei der Betrachtung zu früh auf die Formalismen übergegangen wird, und entsprechend liegt eine Unterbewertung der Formalismen vor, wenn man bei der Betrachtung zu spät auf die Formalismen übergeht. Dabei bedeutet der Übergang zur Betrachtung von Formalismen, daß man die Formen und deren Transformationen in einer solchen Weise betrachtet, daß dabei mit den betrachteten Elementen und Strukturen kein intuitives Verständnis mehr verbunden sein muß.

Dies bedeutet selbstverständlich nicht, daß man mit der Verwendung formaler Begriffe erst beginnen sollte, wenn man in die formale Betrachtung eintritt. Auch in dem Teil einer Systembetrachtung, worin es um den Zweck und damit um ein intuitives Systemverständnis geht, dient es der sprachlichen Präzision und damit dem Kommunikationswirkungsgrad, wenn man Begriffe aus der formalen Welt mit anschaulicher Interpretation verwendet. Es ist hier insbesondere an die Begriffswelt der Mengenlehre und der formalen Logik gedacht.

2. Der Zweck der Anschauung

Die Wahlmöglichkeit, sein primäres Interesse auf die Formalismen oder deren Zweck zu richten, findet ihre Entsprechung in zwei Fragen, die Dijkstra in [Dijkstra–89] zur Charakterisierung zweier streng auseinanderzuhaltender Problembereiche formuliert hat. Die eine Frage lautet: Ist das spezifizierte System das gewollte? Und die andere Frage lautet: Erfüllt das realisierte System die Spezifikation? Mit der ersten Frage wird das *Validierungsproblem* charakterisiert, mit der zweiten Frage das *Verifikationsproblem*. Das Verifikationsproblem läßt sich als mathematisches Problem behandeln, wenn die formale Spezifikation vollständig ist, d.h. wenn bei der Interpretation der Spezifikation ganz auf ein intuitives Verständnis verzichtet werden kann. Bisher wurde noch kein wirklich komplexes System aus der industriellen Praxis in diesem Sinne vollständig formal spezifiziert, und es ist auch nicht abzusehen, ob dies jemals geschehen wird. Dies ist aber keine Frage der grundsätzlichen Machbarkeit, sondern eine Frage der Wirtschaftlichkeit.

Im Gegensatz zum Verifikationsproblem läßt sich das Validierungsproblem grundsätzlich nicht zu einem mathematischen Problem machen. Es geht ja hierbei um die Frage, ob das spezifizierte System das gewollte sei. Die Information über das Gewollte befindet sich aber eingeschlossen in den Köpfen von Menschen. Wenn sie dieses Gewollte in Schrift und Zeichnung formulieren, können sie schon nicht mehr sicher sein, ob das Geschriebene und Gezeichnete tatsächlich das ausdrückt, was sie wollten. Je anschaulicher ihre Formulierungsmittel sind, mit denen sie ihr Gewolltes ausdrükken, desto leichter wird es ihnen fallen, das Formulierte daraufhin zu überprüfen, ob es mit dem Gewollten übereinstimmt. Je unanschaulicher die Formulierungsmittel sind, um so eher kann es vorkommen, daß der Wollende etwas formuliert, von dem er glaubt, es beschreibe das Gewollte, während es tatsächlich vom Gewollten abweicht. Nicht jede Sprache eignet sich deshalb in gleicher Weise für die Formulierung des Gewollten.

Normalerweise wird eine Sprache, die zur Bewältigung des Validierungsproblems angemessen ist, nicht auch für die Bewältigung des Implementierungsproblems angemessen sein. Das *Implementierungsproblem* besteht darin, für das Gewollte eine Formulierung zu finden, die man dem Computer übergeben kann, damit dieser dann zum gewollten System wird. Dem Computer muß man selbstverständlich eine vollständig formale Formulierung übergeben, denn dieser kennt keine intuitive Semantik. Für den Computer ist es nur wichtig, daß die Formulierung des Gewollten formalsemantisch vollständig und widerspruchsfrei ist.

Aus diesen Überlegungen folgt, daß man mehrere Formulierungen benötigt, um von dem Willen, ein System mit bestimmten Eigenschaften zu realisieren, zu dem tatsächlich realisierten System zu gelangen. An einem sehr einfachen Beispiel soll die Notwendigkeit von unterschiedlichen Formulierungen veranschaulicht werden.

3. Ein Beispiel

3.1 Anschauliche Beschreibung

Bei diesem Beispiel geht es gar nicht um ein komplexes System, sondern um einen einfachen Stapel. Als Sprache, bei deren Verwendung man ziemlich sicher sein kann, daß das Formulierte mit dem tatsächlich Gewollten übereinstimmt, benutzt man im gegebenen Fall zweckmäßigerweise die Begriffswelt der Mengenlehre und verbindet diese mit der Anschauung.

Da das Wort ”Stapel” in drei verschiedenen Bedeutungen benutzt wird, sollte man sorgfältig darauf achten, daß bei jeder Verwendung dieses Wortes eindeutig klar ist, in welcher Bedeutung es jeweils aktuell benutzt wird. Mit dem Wort ”Stapel” kann eine Speichervariable gemeint sein oder ein aktueller Speicherinhalt oder ein System, welches aus einem Speicher und einem Akteur besteht, der von seiner Umgebung Anfragen oder Änderungswünsche bezüglich des Speicherinhalts entgegennimmt und diese beantwortet bzw. ausführt.

Wenn man von einem Stapel im Sinne eines Speicherinhalts spricht, meint man damit eine linear geordnete endliche Menge, und man verbindet damit die Anschauung einer Menge von flachen Gegenständen, die man auf einer horizontalen Fläche zu einem Stapel aufgetürmt hat. Deshalb bezeichnet man das in der Linearordnung am einen Ende liegende Element als das unterste und das am anderen Ende der Ordnung liegende Element als das oberste.

Wenn man von einem Stapel im Sinne eines Systems spricht, welches aus einem Speicher und einem darauf zugriffsberechtigten Akteur besteht, verbindet man mit dem Akteur ein ganz bestimmtes Repertoire von Anweisungen, die er von der Systemumgebung entgegennehmen kann. Es wird hier das folgende Repertoire aus fünf Anweisungen betrachtet:

CLEAR	Nach Ausführung dieser Anweisung soll der Speicher leer sein, d.h. der Stapel soll anschließend kein einziges Element mehr enthalten.
PUSH(Element)	Das als Parameter dieser Anforderung übergebene Element soll als neues oberstes auf den bisherigen Stapel gelegt werden.
POP	Das aktuell oberste Element soll entfernt werden. Diese Anweisung ist unzulässig, wenn der aktuelle Stapel leer ist.
HEIGHT	Diese Anfrage soll mit der Angabe beantwortet werden, wieviele Elemente aktuell im Speicher liegen.
TOP	Diese Anfrage soll mit der Angabe beantwortet werden, welches Element aktuell als oberstes im Speicher liegt. In gleicher Weise wie die Anweisung POP ist auch die Anweisung TOP unzulässig, wenn der aktuelle Stapel leer ist.

Man beachte, daß es sich bei der gegebenen Beschreibung des Anweisungsrepertoires für das Stapelsystem nicht um eine formale Beschreibung dieses Systems handelt, obwohl in dieser Beschreibung die präzise Begriffswelt der Mengenlehre verwendet wird. Denn bei der Angabe der Bedeutung der verschiedenen Anforderungen mußte auf die Anschauung einer vertikal im Raum liegenden, geordneten endlichen Menge von Dingen Bezug genommen werden. Nun aber soll versucht werden, eine formale Spezifikation des Stapelsystems zu skizzieren.

3.2 Formale Beschreibung

Die formale Spezifikation wird hier nicht vollständig ausgeführt; es geht lediglich darum, dem Leser das grundsätzliche Prinzip der formalen Spezifikation zu vermitteln. Dabei ist die hier gewählte Form der formalen Spezifikation [Bart./Parnas–78] nicht die einzig mögliche; es gibt durchaus unterschiedliche Ansätze zur formalen Spezifikation von Systemen. Der hier gewählte Ansatz wurde gewählt, weil er sich für den verfolgten Zweck besonders gut eignet. Es soll nämlich gezeigt werden, daß ein Mensch, der noch keine intuitive Vorstellung von dem spezifizierten System hat, aus einer formalen Spezifikation, die sich auf keinerlei Intuition stützt, keine intuitive Vorstellung gewinnen kann.

An dieser Stelle muß noch etwas mehr über die Rolle der Intuition im Bereich der formalen Beschreibungen gesagt werden. Eine formale Beschreibung muß selbstverständlich immer Beschreibungselemente verwenden, deren Bedeutung dem Leser schon bekannt ist, bevor er mit dem Lesen der Beschreibung beginnt. Mit diesen Elementen muß der Leser also eine intuitive Semantik verbinden. Neben diesen Elementen enthält eine formale Beschreibung nur noch Bezeichner für Elemente, die der eigentliche Gegenstand der Beschreibung sind, d.h. deren Bedeutung erst durch die Beschreibung selbst festgelegt wird.

Diejenigen Elemente einer Beschreibung, mit denen man schon vor dem Lesen vertraut sein muß, weil ihre Bedeutung nicht aus der Beschreibung selbst hervorgeht, sind die Elemente der Beschreibungsmethodik; die anderen Elemente dagegen, deren Bedeutung erst durch die Beschreibung selbst vermittelt werden soll, sind die Elemente des Beschriebenen. Die Elemente der Beschreibungsmethodik bilden eine sog. *Metawelt*; die Elemente des Beschriebenen bilden die sog. *Gegenstandswelt*.

Weiter oben wurde gesagt, daß sich eine formale Beschreibung nicht auf intuitive Anschauung stützen darf. Diese Aussage muß nun dahingehend präzisiert werden, daß der Verzicht auf intuitive Anschauung nur für die Gegenstandswelt, nicht aber für die Metawelt gelten kann.

Durch die nun anschließende Betrachtung einer formalen Spezifikation des Stapelsystems werden die zuletzt gemachten Aussagen über die Notwendigkeit der Unterscheidung zwischen der Metawelt und der Gegenstandswelt für den Leser verständlicher werden. Im betrachteten Beispiel besteht die Gegenstandswelt aus dem Stapelsystem und insbesondere aus dem Repertoire der Anweisungen an dieses System. Die zentralen Begriffe der Metawelt, die im folgenden erklärt werden sollen, sind: die Anweisungsfolge, das Legalitätsprädikat, die Folgenäquivalenz und die Ausgabefunktion.

In der hier vorzustellenden Metawelt geht es ausschließlich um die Beschreibung von Gegenstandswelten, die jeweils aus einem sequentiell zu betreibenden System bestehen, welches nacheinander auf Anweisungen reagiert, die aus einem endlichen Repertoire stammen. Deshalb ist die Anweisungsfolge ein zentraler Begriff der Metawelt. Jede *Anweisungsfolge* hat eine endliche Länge und ist als eine mögliche Folge von Anweisungen zu verstehen, die das System seit seiner Inbetriebnahme bis zum aktuellen Zeitpunkt entgegengenommen hat. Da nicht unbedingt in jedem Zustand des Systems jede Anweisung aus dem Repertoire akzeptiert werden kann, läßt sich bezüglich der Anweisungsfolgen das *Legalitätsprädikat* definieren. Eine Anweisungsfolge ist legal, wenn sie von dem System vollständig akzeptiert werden kann.

Zwei Anweisungsfolgen sind einander *äquivalent*, wenn sie bezüglich der Festlegung des zukünftigen Systemverhaltens austauschbar sind. Die Austauschbarkeit bezüglich der Festlegung des zukünftigen Systemverhaltens liegt genau dann vor, wenn für jede beliebige Folgenverlängerung die

Reaktionen des Systems unabhängig davon sind, an welche der beiden äquivalenten Folgen die Verlängerung angehängt wird.

Der Argumentbereich der *Ausgabefunktion* ist die Menge aller legalen Anweisungsfolgen, die nicht die Länge Null haben. Die Ausgabefunktion ordnet jeder solchen Anweisungsfolge dasjenige Ausgabeelement zu, das von dem System als Reaktion auf die letzte Anweisung in der Folge geliefert wird.

Im betrachteten Fall äußert sich die Metawelt durch die folgenden Symbole:

λ	ist das Legalitätsprädikat.
a	ist eine Variable, die jeweils mit einer Anweisungsfolge zu belegen ist.
$\bullet$	ist der Konkatenationsoperator zum Aneinanderhängen zweier Teilfolgen.
$\equiv\equiv$	symbolisiert die Äquivalenz der links vom Symbol stehenden Folge mit der rechts vom Symbol stehenden Folge.
ω	ist die Ausgabefunktion.

Darüberhinaus werden Symbole verwendet, die in der Arithmetik und in der Aussagenlogik üblich sind, z.B. $=$, $+$, $\rightarrow$.

Wenn man die Bedeutung der Symbole für die Elemente der Metawelt kennt, liest man die nachfolgenden formalen Ausdrücke, die einen Auszug aus der formalen Spezifikation des Stapelsystems darstellen, verhältnismäßig leicht. Dies liegt daran, daß man durch die Bezeichnungen für die Elemente aus der Gegenstandswelt darauf hingewiesen wird, daß es sich hier um eine formale Beschreibung eines Stapelsystems handelt, von dem man bereits eine intuitive Vorstellung hat. Diese intuitive Vorstellung entsteht nicht erst durch die formale Beschreibung, sondern entstand bereits anläßlich einer nichtformalen Beschreibung, wie sie im Abschnitt 3.1 zur Einführung des Anweisungsrepertoires gegeben wurde.

$\lambda[\ a\ ] \rightarrow \lambda[\ a \bullet \mathrm{PUSH}(\mathit{Element})\ ]$

$\lambda[\ a \bullet \mathrm{POP}\ ] = \lambda[\ a \bullet \mathrm{TOP}\ ]$

$a \bullet \mathrm{HEIGHT} \equiv\equiv a$

$a \bullet \mathrm{PUSH}(\mathit{Element}) \bullet \mathrm{POP} \equiv\equiv a$

$\lambda[\ a \bullet \mathrm{TOP}\ ] \rightarrow (\ a \bullet \mathrm{TOP} \equiv\equiv a\ )$

$\omega[\ a \bullet \mathrm{PUSH}(\mathit{Element}) \bullet \mathrm{TOP}\ ] = \mathit{Element}$

$\omega[\ a \bullet \mathrm{PUSH}(\mathit{Element}) \bullet \mathrm{HEIGHT}\ ] = 1 + \omega[\ a \bullet \mathrm{HEIGHT}\ ]$

$\omega[\ \mathrm{HEIGHT}\ ] = \omega[\ a \bullet \mathrm{CLEAR} \bullet \mathrm{HEIGHT}\ ] = 0$

4. Die Notwendigkeit zweier Beschreibungen

Daß eine formale Beschreibung tatsächlich nicht geeignet ist, ein intuitives Verständnis für die Gegenstandswelt zu vermitteln, konnte der Autor in vielen Experimenten empirisch nachweisen. Er hat dazu die Ausdrücke der formalen Spezifikation des Stapelsystems derart umgeschrieben, daß er anstelle aller auf den Stapel hinweisenden Bezeichner andere Bezeichner eingesetzt hat, mit denen man a priori keinerlei Anschauung verbindet. So wurden anstelle der Bezeichner CLEAR, PUSH, POP, HEIGHT und TOP die Bezeichner RAL, LING, HUM, HOOM und SES eingesetzt. Die derart modifizierte formale Spezifikation blieb praktisch allen Versuchspersonen unverständlich, und der Überraschungseffekt war stets verhältnismäßig groß, wenn man ihnen die Rückübersetzung der Bezeichner verriet.

Formale Beschreibungen sind zwar das beste Mittel zur Lösung des Mißverständlichkeitsproblems; das betrachtete Beispiel zeigt aber, daß sie nicht geeignet sind, das Unverständlichkeitsproblem zu lösen. Das *Mißverständlichkeitsproblem* besteht in der Möglichkeit, daß jemand beim Lesen einer Beschreibung glaubt, alles verstanden zu haben, was der Schreiber sagen wollte, der Schreiber aber dennoch etwas anderes gemeint hat. Das *Unverständlichkeitsproblem* dagegen besteht darin, daß der Leser eine Beschreibung zwar als Form zur Kenntnis nehmen kann, ihr aber keine schlüssige Information entnehmen kann. Man benötigt deshalb immer zwei Arten von Beschreibungen. Zur Vermittlung eines intuitiven Systemverständnisses braucht man eine nichtformale Beschreibung, und zur Korrektur möglicher Mißverständnisse und Informationsdefizite, welche die nichtformale Beschreibung möglicherweise hinterläßt, braucht man anschließend eine formale Beschreibung, bei der man aber die Bezeichner so wählen muß, daß der Bezug zum intuitiven Verständnis erhalten bleibt.

5. Defizite der Informatik

Es dürfte leicht einzusehen sein, daß man sich unnötigerweise eine Menge schwieriger Probleme einhandelt, wenn man der Erstellung von Beschreibungen zur Vermittlung des intuitiven Systemverständnisses nicht die erforderliche Sorgfalt widmet. Selbstverständlich kann man trotz aller Sorgfalt keine guten Ergebnisse erzielen, wenn man nicht weiß, wie man methodisch an die Erstellung solcher Beschreibungen herangehen soll. Deshalb ist es eine zentrale Aufgabe der Informatik als Ingenieursdisziplin, geeignete Regeln und Methoden zu suchen und zu vermitteln, deren Beachtung bzw. Anwendung sicherstellt, daß Beschreibungen angefertigt werden, die dem Bedürfnis nach einem hohen Kommunikationswirkungsgrad gerecht werden. Leider hat die Informatik als wissenschaftliche Disziplin diese Aufgabe bisher sträflich vernachlässigt. Fast jede beliebige Stichprobe, bei der man irgendwelche Lehrbücher, wissenschaftliche Aufsätze oder Software–Handbücher studiert, offenbart den betrüblichen Sachverhalt, daß das methodische Beschreiben von Systemen zur Vermittlung eines intuitiven Systemverständnisses nicht nur nicht beherrscht, sondern noch nicht einmal ernsthaft versucht wird.

In der Informatik herrscht eine derart übermäßige Begeisterung für Formalismen und damit für Automatismen vor, daß dadurch die Wahrnehmung wesentlicher Probleme verhindert wird, die gelöst werden müßten, damit die Informatik zu einer Ingenieurwissenschaft wird. Es wird hier ja keineswegs verlangt, daß die Informatik sich nicht mehr mit Formalismen befassen solle. Es wird lediglich behauptet, daß die Informatiker dazu neigen, sich bei der Behandlung von Systemen voreilig in die Formalisierung zu stürzen, und daß sie sich dadurch unnötige, aber schwerwiegende Probleme einhandeln.

Die Schwerpunktsetzung auf die Formalismen und die Vernachlässigung der Erfordernisse der zwischenmenschlichen Kommunikation kommen auch in den Schriften zum Ausdruck, in denen graphische Darstellungsmittel wie Petrinetze, Statecharts, Klassenbäume o.ä. eingeführt werden. In diesen Einführungen werden die Strukturen primär als Formalismen eingeführt und nicht als Darstellungsmittel zur Gestaltung von Beschreibungen mit hohem Kommunikationswirkungsgrad. Das Layout der Graphiken, die von den Autoren als Beispiele gezeigt werden, ist in den meisten Fällen derart unbefriedigend, daß man sicher sein kann, daß die Autoren der Optimierung des Layouts keinerlei Aufmerksamkeit gewidmet haben. Auf eine optimale Nutzung der Gestaltwahrnehmungsfähigkeiten des Menschen kam es ihnen offensichtlich nicht an.

Zum Abschluß dieses Abschnitts will der Autor eine Begebenheit berichten, die er auf einem Softwaretechnologiekongress erlebt hat und die das in den hier vorgestellten Überlegungen behandelte Problem kurz und prägnant vor Augen führt. In der Diskussion, die sich an die Präsentation des Beitrags [Zuck–94] auf diesem Kongress anschloß, sagte eine Teilnehmerin: ”Aus Ihrer Darstellung von Methoden der Systembeschreibung habe ich nicht erkannt, wie man diese Beschreibungen zur automatischen Generierung von Programmcode nutzen könnte. Wenn man daraus aber keinen Programmcode generieren kann, sind Ihre Beschreibungen doch nutzlos.” Deutlicher hätte diese Informatikerin ihre Einäugigkeit nicht offenbaren können.

6. Literatur

[Bart./Parnas–78] W. Bartussek, D. L. Parnas: Using Assertions about Traces to Write Abstract Specifications for Software Modules.
Proceedings of the Symposium ”Information System Methodology”, Venice.
Lecture Notes in Computer Science; 1978.

[Dijkstra–89] Edsger W. Dijkstra: Dijkstra’s Reply to Comments.
in: Peter J. Denning: A Debate on Teaching Computer Science.
Communications of the ACM, vol. 32, p. 1414; 1989

[Zuck–94] Wolfgang Zuck: R/3–Basismodellierung.
Abschnitt C3 im Tagungsband 1 der SAP–Technologie–Tage,
Karlsruhe, 1994.