http://zuse.zib.de

Title:	Zur Rezeption des Plankalküls von Konrad Zuse
Author(s):	Katja Schunke
Date:	1998
Published by:	Konrad Zuse Internet Archive
Source:	Essay - ZIA ID: 0685

The Konrad Zuse Internet Archive preserves and offers free access to the digitized original documents of Konrad Zuse's private papers and to other related sources.
The Konrad Zuse Internet Archive is a nonprofit service that helps scholars, researchers, students and other interested parties discover, use and build upon a wide range of content in a digital archive. For more information about the Konrad Zuse Internet Archive, please contact zusearchive@zib.de.

Your use of the Konrad Zuse Internet Archive indicates your acceptance of the Terms & Conditions of Use (http://zuse.zib.de/tou) including the following license agreement. If you do not accept the Terms & Conditions of Use you are not permitted to use the material.

This work by Konrad Zuse Internet Archive is licensed under a
Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported License
(http://creativecommons.org/licenses/by-nc-sa/3.0/).
Based on a work at http://zuse.zib.de

Attribution (BY) - You must attribute the work in the manner specified by the author or licensor (but not in any way that suggests that they endorse you or your use of the work). Attribute with "Konrad Zuse Internet Archive (http://zuse.zib.de)".

Noncommercial (NC) - You may not use this work for commercial purposes.

Share Alike (SA) - If you alter, transform, or build upon this work, you may distribute the resulting work only under the same or similar license to this one.

The usage of this document requires the consideration of possible third party copyrights, and might necessitate obtaining the consent of the copyright holder. The Konrad Zuse Internet Archive assumes no liability with respect to the rights of third parties. The Konrad Zuse Internet Archive is not responsible for the claims of any third party resulting from any infringement of copyright laws.

Zur Rezeption des Plankalküls von Konrad Zuse. Die Einordnung in die Programmiersprachenkonzepte zur Zeit seiner Veröffentlichung

Diplomarbeit

Technische Universität Berlin

Fachbereich Informatik

Studiengebiet Informatik und Gesellschaft

Prof. Dirk Siefkes

Betreuer: Prof. Dirk Siefkes

eingereicht von: Katja Schunke

Matr.-Nr. 113524

am 21. Mai 1998

(korrigierte und überarbeitete Version September 1999)

Inhaltsverzeichnis

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

1. Einleitung

Der Computerpionier Konrad Zuse hat neben dem Bau seiner automatischen Rechengeräte umfangreiche theoretischen Studien betrieben. In diesem Zusammenhang hat er bereits 1945 mit dem *Plankalkül* die erste höhere Programmiersprache entwickelt. Dieser Tatbestand blieb jahrzehntelang weitgehend unbeachtet, zumal keine vollständige Veröffentlichung des Plankalküls vorlag. Auf die allgemeine Entwicklung von Programmiersprachen, die in den 50er Jahren begann, hatte der Plankalkül nur geringfügigen Einfluß.

Erst im Zusammenhang mit dessen vollständiger Veröffentlichung zu Beginn der 70er Jahre, wurde dem Plankalkül kurzfristig ein größeres Interesse entgegen gebracht. Zu diesem Zeitpunkt sind einige Arbeiten entstanden, die sich inhaltlich mit den Sprachkonzepten des Plankalküls auseinandersetzen. Von besonderem Interesse ist dabei die jeweils vorgenommene Einordnung des Plankalküls in bestehende Programmiersprachenkonzepte, d.h. dessen Zuordnung zu den funktionalen, logischen oder objekt-orientierten Programmiersprachen bzw. zu den imperativen ‘von Neumann-Sprachen’.

Die Zielsetzung der vorliegenden Arbeit ist es zu untersuchen, wie der Plankalkül zum Zeitpunkt seiner Veröffentlichung von der Fachwelt bewertet wurde. Dabei ist von besonderem Interesse, welchen etablierten Programmiersprachen der Plankalkül gegenübergestellt wurde und ob damit gleichzeitig eine Einordnung des Plankalküls in bestehende Programmiersprachenkonzepte erfolgte. Erst anhand einer solchen Einordnung kann eine abschließende Bewertung der im Plankalkül realisierten Sprachkonzepte vorgenommen werden.

Da der Plankalkül bis heute weitgehend unbekannt geblieben ist, werden in *Kapitel 2* die Hintergründe für dessen Entwicklung und die Einordnung des Plankalküls in Konrad Zuses Gesamtwerk dargestellt. Zusätzlich wird auf die Pläne von Konrad Zuse in bezug auf Möglichkeiten einer Implementierung eingegangen.

Um eine Beurteilung der betrachteten Rezeptionen vornehmen zu können, ist eine eigene Annäherung an den Plankalkül im Rahmen dieser Arbeit zwingend notwendig. Die Originalfassung des Plankalküls von 1945 besteht aus einer Sammlung von Beispielprogrammen mit zusätzlichen verbalen Beschreibungen der Funktionsweise der einzelnen Sprachkonstrukte. Da der inzwischen übliche Sprachgebrauch zur Beschreibung von Programmiersprachen teilweise erheblich von der von Konrad Zuse gewählten Beschreibung abweicht, wird in *Kapitel 3* eine verbale Sprachbeschreibung des Plankalküls erstellt. Dadurch wird es möglich, die in den Rezeptionen vorgenommenen Analysen des Plankalküls zu überprüfen und die vorgenommene Bewertung einzuschätzen.

Obwohl Konrad Zuse seit Ende der 40er Jahre wiederholt versucht hat, das Interesse am Plankalkül zu wecken, ist dieser erst 1972 vollständig veröffentlicht worden. In *Kapitel 4* sind die verschiedenen Veröffentlichungsversuche zusammengestellt. Die Gründe für ihr Mißlingen werden aufgezeigt.

Die in den 70er Jahren erschienen Rezeptionen zum Plankalkül werden in *Kapitel 5* vorgestellt und auf die darin

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

vorgenommenen Analysen und Bewertungen hin untersucht. Es werden folgende Rezeptionen herangezogen:

- 1972: *Zuses "Plankalkül", ein Vorläufer der Programmiersprachen – gesehen vom Jahr 1972* von F. L. Bauer und H. Wössner
- 1975: *Eine Untersuchung des Plankalküls im Vergleich mit algorithmischen Sprachen* von Joachim Hohmann
- 1975: *Gesichtspunkte zur Beurteilung algorithmischer Sprachen* von Konrad Zuse
- 1977: *Beschreibung des Plankalküls* von Konrad Zuse
- 1977: *The Early Development of Programming Languages* von Donald E. Knuth und Luis Trabb Pardo

Bei der Analyse der einzelnen Arbeiten wird zunächst nach den Hintergründen zu Motivation und Entstehung gefragt, daran anschließend erfolgt die Beschreibung der methodischen Vorgehensweise bei der Analyse des Plankalküls. Sofern dabei auffällige Besonderheiten oder Schwerpunktsetzungen vorgenommen wurden, werden diese gesondert betrachtet. Abschließend wird die jeweilige Bewertung des Plankalküls dargestellt.

Die zusammenfassende Betrachtung und Bewertung der analysierten Rezeptionen erfolgt in *Kapitel 6*. Im Mittelpunkt steht dabei die jeweilige Einordnung des Plankalküls in die unterschiedlichen Programmiersprachenkonzepte, dabei werden zusätzlich festgestellte Besonderheit bei der Analyse des Plankalküls dargestellt. Aufgrund der sehr unterschiedlichen Herangehensweise an die Analyse des Plankalküls, kann eine Unterscheidung zwischen den deutschsprachigen Texten und der Arbeit D. E. Knuth und L. Trabb Pardo aus den USA getroffen werden. Letztere betrachten den Plankalkül ausschließlich unter historischen Gesichtspunkten, ermöglichen aber den direkten Vergleich des Plankalküls mit FORTRAN I. In Deutschland wird der Plankalkül hingegen vor dem Hintergrund der Software-Krise diskutiert. Dabei ist vor allem die strikte Fokussierung auf den Vergleich mit ALGOL 68 auffallend, welche auch bei den Arbeiten von Konrad Zuse zu finden ist.

1. Der Plankalkül

Konrad Zuse (1910 - 1995) ist hauptsächlich durch seine Entwicklungen von automatischen Rechengeräten bis zur Mitte dieses Jahrhunderts bekannt. Es ist inzwischen historisch unbestritten, daß er 1941 mit dem Rechenautomaten Z3 den ersten voll funktionsfähigen programmgesteuerten Computer der Welt fertiggestellt hat.

Weniger bekannt ist hingegen, daß er bereits 1945 die Programmiersprache *Der Plankalkül* entwickelt hat, die als die erste höhere Programmiersprache angesehen werden kann. Der Plankalkül ist vor allem deshalb beachtenswert, weil er in den behandelten Anwendungsproblemen bereits weit über mathematisch-numerische Problemstellungen hinausgeht. Er wurde vielmehr hauptsächlich für logische Rechenvorgänge konzipiert, und die von Konrad Zuse erstellten Programmbeispiele konzentrieren sich deshalb auf die Bereiche der *Prädikatenlogik* und der *Graphentheorie*. Darüber hinaus gibt es Ansätze zur Automatisierung des Schachspiels, die in der *Schachtheorie* zusammengefaßt sind.

Im Folgenden wird die kurze Entwicklungsgeschichte des Plankalküls wiedergegeben. Auch wenn dieser nie über das Stadium einer theoretischen Arbeit hinausgekommen ist, hat Konrad Zuse doch konkrete Vorstellungen über eine mögliche Implementierung des Plankalküls entwickelt. Diese werden ebenfalls dargestellt.

1. *Zur Entstehung des Plankalküls*

Konrad Zuse steht in dem Ruf ein Praktiker und ‘genialer Bastler’ zu sein. Es wird dabei in der Regel kaum beachtet, daß er seine sämtlichen technischen Entwicklungen durch vorangehende und begleitende theoretische Studien fundiert hat. So hat Zuse z.B. als Voraussetzung für die Entwicklung seines ersten automatischen Rechengerätes Z1, Ende der 30er Jahre, die entsprechenden theoretischen Konzepte entwickelt, auf deren Grundlage er sich z.B. für die Verwendung der binären Zahlendarstellung und der Gleitkomma-Arithmetik entschieden hat. Diese Grundkonzepte hat er dann in der Folge für den Bau seiner weiteren Rechengeräte beibehalten.

Eine Folge dieser Überlegungen war seine Beschäftigung mit der Formalisierung und Automatisierung von

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

logischen Berechnungen, die Zuse dann während des Krieges in einem Dissertationsvorhaben mit dem Titel *Ansätze einer Theorie des allgemeinen Rechnens unter besonderer Berücksichtigung des Aussagenkalküls und dessen Anwendung auf Relaisschaltungen* untersuchen wollte. Der Leitsatz dieser Arbeit war: *"Rechnen heißt* [...]*: Aus gegebenen Angaben nach einer Vorschrift neue Angaben zu bilden."*. Die ersten gedanklichen Voraussetzungen für den Plankalkül waren damit getan.

Das Ende des 2. Weltkrieges und die damit verbundene Flucht aus Berlin, setzten diesen Überlegungen ein Ende. Die Dissertation wurde nie fertiggestellt.

Diese Flucht aus Berlin Anfang 1945, bei der die gerade fertiggestellte Z4 mitgenommen werden konnte, hat Konrad Zuse in das Dorf Hinterstein im Allgäu geführt. Dort bestand für ihn jedoch keine Möglichkeit der weiteren praktischen Arbeit an seinen Rechengeräten. Vielmehr stellte sogar die Reparatur der bei der Flucht beschädigten Z4 ein großes Problem dar. Erst 1949 gelang es Konrad Zuse, seine bereits 1941 gegründete Firma zum Bau von automatischen Rechengeräten in Bad Hersfeld, Hessen unter dem Namen ZUSE KG neu zu gründen.

Auf Grund der fehlenden Möglichkeiten für seine praktischen Arbeiten an der Z4 hat Konrad Zuse zunächst seine theoretischen Überlegungen weiter verfolgt. In diesem Zusammenhang entstand 1945 der Plankalkül, der als ein Bestandteil seiner Dissertation gedacht war. Die entsprechende theoretische Arbeit ist innerhalb kürzester Zeit entstanden. Sie war demzufolge teilweise inkonsistent und enthielt in einzelnen Programmbeispielen logische Fehler. Da es in der unmittelbaren Nachkriegszeit keine Möglichkeit für wissenschaftliche Veröffentlichungen gab und Konrad Zuse darüber hinaus nicht über die finanziellen Mittel verfügte, die er für die notwendige Überarbeitung des Plankalküls hätte aufbringen müssen, verblieb der Plankalkül letztendlich in dieser unbearbeiteten und fehlerhaften ‚Rohfassung'.

Die vollständige Veröffentlichung dieser Originalfassung des Plankalküls erfolgte erst 1972. In den dazwischen liegenden Jahrzehnten hat es einige Versuche von Konrad Zuse gegeben, den Plankalkül in die inzwischen entstandene Diskussion um Programmiersprachen einzubringen. Es ist ihm aber nicht gelungen ein entsprechendes Interesse am Plankalkül zu wecken.

2. *Zu einer Implementierung des Plankalküls*

Der Plankalkül wurde nie implementiert. Er lag auch in keiner Fassung vor, die eine Implementierung ermöglicht hätte.

Die von Konrad Zuse bis 1945 entwickelten programmgesteuerten Rechengeräte, einschließlich der Z4, waren nicht für eine Implementierung des Plankalküls geeignet. Aber bereits während des Krieges wurde von Konrad Zuse das Konzept für ein ‚logistisches' Rechengerät entwickelt, auf dem der Plankalkül eingesetzt werden sollte. 1944 hat er ein entsprechendes Versuchsmodell gebaut, welches aber, wie die meisten seiner Rechengeräte, im Krieg zerstört wurde. Nach dem Krieg konnte er nicht die finanziellen Mittel aufbringen, um das 'logistische' Rechengerät zu bauen und damit die Voraussetzung für eine Implementierung des Plankalküls zu schaffen.

Ende der 50er Jahre sah Konrad Zuse dann die Möglichkeit einer Implementierung des Plankalküls auf seiner elektronischen Rechenanlage Z22. Aber auch diesmal ist ihm die Finanzierung nicht gelungen. Vielmehr sah er sich gezwungen einen ALGOL-Compiler auf der Z22 einzusetzen, wollte er sich die für den Erhalt seiner Firma unbedingt benötigte finanzielle Unterstützung der Deutschen Forschungsgesellschaft (DFG) erhalten. Diese hatte die Vergabe von Geldern an die Unterstützung der allgemeinen Durchsetzung der Programmiersprache ALGOL geknüpft.

2. Beschreibung des Plankalküls

Die vorliegende Beschreibung des Plankalküls basiert auf den Sprachbeschreibungen der in dieser Arbeit zu analysierenden Rezeptionen des Plankalküls. Bei der Zusammenstellung wurde die Richtigkeit der Quellenangaben nicht überprüft. Außerdem erhebt diese Sprachbeschreibung keinen Anspruch auf Vollständigkeit. Vielmehr wurden die Rezeptionen auf Gemeinsamkeiten in der Schwerpunktsetzung untersucht, auf denen dann die Gliederung dieser Arbeit aufbaut.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Problematisch war bei der Zusammenstellung dieser Sprachbeschreibung, daß die betrachteten Rezeptionen in den 70er Jahren entstanden sind: zum einen hat sich der allgemeine Sprachgebrauch zur Beschreibung von Programmiersprachen zwischenzeitlich verändert, zum anderen sind sämtliche Arbeiten sehr stark auf damals aktuelle Programmiersprachen fixiert, vor allem auf ALGOL 68, welche hauptsächlich zum Vergleich herangezogen wurde. Daher sind die in den Rezeptionen verwendeten verbalen Sprachbeschreibungen des Plankalküls größtenteils unverständlich, da die Notation von ALGOL 68 heutzutage nicht mehr gebräuchlich ist.

Es mußte eine erneute ‚Übersetzung' des Plankalküls in den heute üblichen, und hoffentlich neutraleren Sprachgebrauch vorgenommen werden, um eine Wertung des Plankalküls, wie sie durch den direkten Bezug zu einer anderen Programmiersprache bereits bei der Sprachbeschreibung entsteht, zu vermeiden. Es war demzufolge teilweise notwendig, bei Verständnisschwierigkeiten die originale Beschreibung des Plankalküls von 1945 heranzuziehen. Diese ist aus heutiger Sicht teilweise eher einsichtig als die stark an der ALGOL-Notation orientierten Beschreibungen der 70er Jahre, da Konrad Zuse gezwungen war, die Konstrukte des Plankalküls anhand einer gängigen mathematischen Notation zu beschreiben.

1. ***Ein Beispiel zur Einführung in die Notation des Plankalküls***

Als Einführung in die Beschreibung des Plankalküls wird zunächst ein Plankalkül-Programm zur *Syntaxanalyse von aussagenlogischen Ausdrücken* vorgestellt. Auf eine inhaltliche Erläuterung dieses Beispieles wird bewußt verzichtet, da es ausschließlich darauf ankommt, einen optischen Eindruck von der ungewöhnlichen Notation des Plankalküls zu vermitteln.

Als einzige Anmerkung zur Notation sei gesagt, daß die über den einzelnen Zeilenblöcken in eckigen Klammern angegebenen Zahlen der Darstellung von Kommentaren dienen. Die zugehörigen Erläuterungen werden im Anschluß an das Programm aufgeführt.

Beispiel 1: Programm zur 'Syntaxanalyse von aussagenlogischen Ausdrücken'

Diese ungewöhnliche und eher abschreckend wirkende Notation stellt vermutlich das größte Hindernis dar, sich dem Plankalkül zu nähern. Die inhaltliche Bedeutung des Programmes bleibt vollkommen unklar.

Leider ist es Konrad Zuse nicht gelungen, durch die Verwendung von geeigneten Erläuterung die Funktionsweise der von ihm aufgestellten Programme zu vermitteln, wie die Wiedergabe der zum obigen Beispiel gehörenden Kommentare zeigt:

1. *Das erste Zeichen muss ein Anfangszeichen sein.*
2. *z_0 ist das letzte Zeichen der gerade untersuchten Aufbauliste. z_1 das folgende.*
   Das erste z_0 wird als Negationszeichen angenommen, damit $\overset{\text{Sq}}{0}\ (\overset{\text{z,}}{0},\ \overset{\text{z)}}{1})$ *erfüllt ist.*
3. *Die "Klammerbilanz" ε ist am Anfang 0.*
4. *Das nächste Glied der Liste* $\overset{\text{V}}{0}$ *ist das neue z_1; gibt es kein solches, so gehe über zu* [10].
5. *Zwischen z_0 und z_1 muss die Folgebedingung Sq1 bestehen.*
6. *Ist z_1 ein Kla-Zeichen, so wird ε um Eins erhöht.*
7. *Ist z_1 ein Klz-Zeichen, so wird ε um Eins erniedrigt.*
8. *ε muss stets grösser oder gleich Null sein.*
9. *z_1 ergibt das neue z_0. Gehe zurück zu* [4].
10. *Das letzte z_0 muss ein Schlusszeichen sein.*
11. *ε muss am Ende Null sein.*

Es erscheint nicht unwahrscheinlich, daß der Plankalkül ein weitaus größeres Interesse gefunden hätte, wenn die dessen Notation zumindest ansatzweise intuitiv zu erfassen wäre und damit die Abläufe der Beispielprogramme verständlich würden.

1. ***Das allgemeine Vercodungsprinzip für Bezeichnungen***

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Der Plankalkül basiert auf einer systematischen Codierung der in einer Formel verwendeten Objekte. Diese werden im allgemeinen durch einen genormten Anfangsbuchstaben in

Verbindung mit einem nachfolgenden Zifferncode, der entweder aus einer einzelnen Ziffer oder aus durch Punkte voneinander getrennte Ziffern bestehen kann, dargestellt:

P	für Programme,
U	für Unterprogramme,
V	für Eingabewerte,
Z	für Zwischenwerte,
R	für Resultatwerte,
C	für Konstante,
S	für Angaben zur Objektstruktur und
A	für Angaben zur Objektart.

Tabelle 1: Vercodungsprinzip für Bezeichnungen

Da der Plankalkül über keine eindeutige Sprachbeschreibung verfügt, sondern ausschließlich auf Beispielen beruht, sind in der Originalfassung des Plankalküls Abweichungen von diesem allgemeinen Vercodungsprinzip enthalten. So wird z.B. nicht zwischen Groß- und Kleinschreibung von Bezeichnungen unterschieden. Trotzdem kann man von einer Allgemeingültigkeit für den Aufbau des Plankalküls ausgehen.

2. *Die zweidimensionale Notation des Plankalküls*

Der Plankalkül verwendet eine zweidimensionale Notation, bestehend aus *Grundzeile*, *Indexzeile*, *Komponentenzeile* und *Strukturzeile*.

In der Grundzeile wird der zu berechnende Formelausdruck angegeben. Die folgenden, maximal drei weiteren Zeilen dienen vor allem der Indizierung. Sie werden ausschließlich für zusätzliche Informationen zu den im Formelausdruck verwendeten Objekten benötigt, um deren eindeutige Identifizierung zu ermöglichen. Diese zusätzlichen Informationen werden direkt unter das in der Grundzeile benannte Objekt geschrieben, so daß sämtliche Objektinformationen als 4-Tupel geschrieben werden.

Die Kennzeichnung der einzelnen Zeilen eines Formelausdruckes erfolgt durch das Voranstellen eines Kennbuchstabens. Diese Kennzeichnung ist zwingend, da zum einen die Komponentenzeile ‘K’ entfallen kann, wenn in ihr keine Angaben enthalten sind. Zum anderen kann die vierte Zeile entweder Angaben zur Datenstruktur oder zur Datenart enthalten. Der jeweilige Fall wird durch das Voranstellen der Kennzeichen ‘S’ oder ‘A’ unterschieden. Der Hauptzeile wird keine Zeilenbezeichnung vorangestellt.

1. Die Hauptzeile

Die Hauptzeile enthält die zu berechnende Formel, deren Darstellungsform aus der Notation mathematischer Formeln abgeleitet ist. Eine derartige Formel besteht demnach aus den Bezeichnungen der in der Berechnung verwendeten Objekte, den benötigten Operationszeichen und den Zeichen für Programmstrukturen. Darüber hinaus kann die Grundzeile auch einfache Anweisungen enthalten. Formeln und Anweisungen unterscheiden sich vor allem dadurch, daß letztere keinen Wert liefern.

Die Operationszeichen bestehen vor allem aus *Klammerzeichen*, die für die Strukturierung, also für die Aufteilung bzw. für die Zusammenfassung von Formelausdrücken, verwendet werden, und welche sich über mehrere neben- und untereinanderliegende Ausdrücke bzw. Zeilensätze erstrecken können.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Diese Form der Strukturierung von Formelausdrücken wurde von Konrad Zuse der besseren Übersicht wegen gewählt.

Zusätzlich werden *senkrechte Trennstriche* zur Programmstrukturierung verwendet, die eine Abgrenzung zwischen zwei Anweisungen darstellen. Damit können auch mehrere aufeinanderfolgende Formelausdrücke in einen Zeilenausdruck geschrieben werden. Inhaltlich ist der senkrechte Trennstrich gleichbedeutend mit der Trennung verschiedener Anweisungen in einzelne Zeilensätze. Es liegt in der Entscheidung des Programmierers, welche Darstellungsform zur Abgrenzung von Ausdrücken er verwendet, um eine Übersichtlichkeit des Programmablaufes zu gewährleisten.

2. Die Indexzeile

Die 2. Zeile enthält den sogenannten 'Variablenindex' und wird durch 'V' gekennzeichnet. Dieser Variablenindex ermöglicht die Unterscheidung der in einer Formel verwendeten Variablen.

Der Unterschied zu dem in der Mathematik gebräuchlichen Index besteht vor allem darin, daß der Variablenindex des Plankalküls fester Bestandteil der Variablenbezeichnung ist. Damit dient die Indexzeile ausschließlich der Fortsetzung der bereits in der Hauptzeile eingeführten Variablenbezeichnung.

Eigentlich könnte damit diese Zeile entfallen, da die Indizierung einer Variablen ebenso gut in der Hauptzeile erfolgen könnte. Zuse betont allerdings, daß die Einführung einer zusätzlichen Zeile vor allem deshalb notwendig ist, da sie "[...] *der Arbeitsweise von Schreibmaschinen entgegen* [kommt]*, bei denen tiefgesetzte Indices stets einige Schwierigkeiten bereiten und* [...] *zum anderen einen klaren Überblick in bezug auf die verwendeten Variablen usw.* [gibt]*, die beim PK* [Plankalkül] *im wesentlichen nur durch ihre Indices unterschieden sind."*

3. Die Komponentenzeile

Der in der 3. Zeile angegebene Komponentenindex ermöglicht den Zugriff auf eine bestimmte Komponente einer in der Hauptzeile angegebenen zusammengesetzten Datenstruktur, z.B. auf die Komponenten einer Paarliste.

Dieser Komponentenzugriff kann durch einfache oder mehrstufige Indizierung erfolgen. Die Wahl der verwendeten Zugriffsart erfolgt in Abhängigkeit von der Komplexität der Zusammensetzung der Datenstruktur.

Die Angabe eines Komponentenindex ist nur dann notwendig, wenn auf einzelne Komponenten eines Objektes zugegriffen werden soll. Enthält die Komponentenzeile keine Angabe, soll die Datenstruktur als Ganzes in die Berechnung eingehen.

Die Funktion der Komponentenzeile läßt sich durch folgende Beispiele verdeutlichen:

V | V V | 3 K | S | m ×2 ×1.n	*Die Variable V_3 ist eine Paarliste von m Paaren der Struktur 2.1.n* [sic!] *und soll als Ganzes in die Rechnung eingehen.*
V | V V | 3 K | i S | 2 ×1.n	*Von der Paarliste V_3 soll das i. Paar genommen werden (Struktur 2.1.n* [sic!]*). (i kann dabei ein laufender Index sein.)*
V | V V | 3 K | i0 S | 1.n	*Von dem i. Paar der Paarliste V_3 soll das Vorderglied (erstes Element des Paares) genommen werden (Struktur 1.n).*

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

V | V V | 3 K | i0.7 S | 0	*Von dem Vorderglied des i. Paares der Paarliste V_3 soll der Ja-Nein-Wert Nr. 7 genommen werden (Struktur S0 = Ja-Nein-Wert).*

Beispiel 2: Funktion der Komponentenzeile

4. Die Strukturzeile

In der letzten Zeile können Angaben zur Struktur bzw. zur Art der in der Berechnung enthaltenen Datenobjekte vorgenommen werden. Allerdings besteht keine Verpflichtung dazu.

Die Strukturzeile dient demnach eher der Programmdokumentation, als daß sie einen Einfluß auf die eigentliche Berechnung der angegebenen Formel hat: *"Die vierte Zeile stellt an sich nur eine mitunter redundante Ergänzung dar und gibt Hinweise auf die Struktur der verwendeten Objekte. Der PK* [Plankalkül] *wurde ja von vornherein geschaffen, um beliebig komplizierte Datenstrukturen verarbeiten zu können. Eine übersichtliche Form der Kennzeichnung dieser Struktur schien daher erforderlich."*

Bei der Angabe der Struktur in der Strukturzeile kann die sonst erforderliche Strukturkennzeichnung 'S' entfallen, da eine eindeutige Kennzeichnung bereits durch die Zeilenbezeichnung vorgenommen worden ist. Damit verkürzen sich z.B. die folgenden Strukturangaben

S\|S1.n m×S1.n S0 S2 σ

zu S\| 1.n m×1.n 0 2 σ .

Entsprechendes gilt, wenn die vierte Zeile zur Kennzeichnung einer Datenart dient. Die Angabe 'A' in der Artbezeichnung kann entfallen.

Die Angabe von Struktur oder Art eines Datenobjektes kann entfallen, wenn dessen wiederholtes Vorkommen innerhalb eines Programmes zu redundanten Angaben führt.

3. *Zur Verwendung einer eindimensionalen Notation*

Die ungewöhnliche Form der zweidimensionalen Notation hat Konrad Zuse gerade deshalb gewählt, weil es sich seiner Meinung nach um eine für den Programmierer sehr übersichtliche Darstellungsform handelt: *"Die Darstellung durch Zeilensätze erscheint zunächst befremdend, da sie ungewohnt ist. Tatsächlich erlaubt sie aber die größtmögliche Klarheit, wobei auf kleinem Raum die verschiedenen zur Kennzeichnung eines Objektes nötigen Angaben gut unterscheidbar konzentriert sind."*

Die besondere Notation des Plankalküls sollte also hauptsächlich eine geeignete mathematische Darstellung von Rechenabläufen ermöglichen. Es war Konrad Zuse bewußt, daß sich eine derartige Darstellungsform nicht für eine Implementierung des Plankalküls eignen würde. Dafür hätte der Plankalkül in eine eindimensionale Form gebracht werden müssen, wodurch er sich vermutlich der Notation von eingeführten Programmiersprachen angenähert hätte: *"Der PK* [Plankalkül] *entstand ja zunächst lediglich unter dem Gesichtspunkt, eine eindeutige Darstellung der Rechenabläufe zu finden. Bei einer Implementierung müssen die Programme des PK* [Plankalkül] *in die Form einer linearen Zeichenfolge gebracht werden. Das ist zweifelsohne durch Verwendung von Darstellungsformen möglich, die den heute üblichen algorithmischen Sprachen entsprechen."*

Die Überführung des Plankalküls in eine eindimensionale Notation könnte nach Ansicht von Konrad Zuse *"*[...] *sehr einfach dadurch geschehen, daß die zu einem Objekt gehörenden und auf die einzelnen Zeilen verteilten Indizes auf die Hauptzeile gehoben werden, wobei dann die links hinausgezogenen Zeilenkennzeichnungen V, K, A* [oder S; Anm. d. Verf.] *in jedem einzelnen Falle vor die Indizes gesetzt werden müssen.*

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Zur Demonstration sei folgende Anweisung gewählt:

	V	=	V	⇒	R
V	0		2		0
K	10				
A	9		9		0

ausgerichtete Form:

V0K10A9 = V2A9 ⇒ R0A0

Man erkennt leicht, daß die im PK [Plankalkül] *gewählte Darstellung durch Zeilensätze eine bessere Übersichtlichkeit ergibt."*

Aus heutiger Sicht läßt sich allerdings durchaus eine Form der Notation denken, welche die Ausdrucksmöglichkeiten des Plankalküls wiedergibt, aber trotzdem übersichtlicher als die von Konrad Zuse vorgeschlagene Methode ist. Die zweidimensionale Notation des Plankalküls dient ausschließlich der eindeutigen Identifizierung von Datenobjekten. Für die eigentliche Anweisung ist die Hauptzeile vollkommen ausreichend. Daher ist es nicht einsichtig, daß für die in der Index- und in der Komponentenzeile enthaltenen Informationen, die ausschließlich der Selektion eines Datenelementes innerhalb des angegebenen Datenobjektes dienen, jeweils eine zusätzliche Zeile verwendet wird. Durch die Definition einer sinnvollen Namenskonvention für Datenobjekte wäre es möglich, die Informationen der Index- und der Komponentenzeile mit der Objektbezeichnung zu verbinden und sie damit auf die Grundzeile zu ziehen.

Darüber hinaus dient die Strukturzeile ausschließlich der Kennzeichnung der Struktur bzw. Art des verwendeten Objektes. Gerade dies ist nach Konrad Zuse ein besondere Vorteil des Plankalküls gegenüber anderen Programmiersprachen, da durch die direkte Zuordnung der Struktur zu den Objekten innerhalb eines Formelausdrucks dessen zusätzliche Deklaration zu Beginn eines Programmes entfallen kann. Allerdings erkennt Zuse auch, daß die Strukturangaben häufig redundant sein können. In diesen Fällen kann auf dies entsprechenden Angaben verzichtet werden. Gerade dadurch geht aber der vermeintliche Vorteil dieser Notation wieder verloren, da nun nicht auf den ersten Blick bei jedem Datenobjekt eines Formelsatzes dessen Struktur oder Art erkennbar ist.

Die heute gebräuchliche Form einer einmaligen Deklaration von Datenstrukturen oder Datenarten innerhalb eines Programmes erscheint dann letztendlich doch sinnvoller, zumal mit den im Plankalkül vorgesehenen redundanten Angaben von Strukturinformationen kein zusätzlicher Informationsgewinn verbunden ist.

Es erscheint demzufolge durchaus denkbar, eine eindimensionale Notation für den Plankalkül zu finden, in der die notwendigen Objektinformationen erhalten bleiben und trotzdem an Übersichtlichkeit gewonnen wird.

4. *Einfache Datenobjekte*

Vom logischen Grundkonzept her kennt der Plankalkül ausschließlich sogenannte *atomare Datenobjekte*, aus denen sich sämtliche weiteren Datenobjekte durch systematisches Zusammensetzen aufbauen lassen.

Allerdings hat Konrad Zuse diesen systematischen Ansatz zum Aufbau von Datenobjekten nicht konsequent durchgeführt und zusätzliche die sogenannten *einfachen Datenobjekte* eingeführt. Denn in der praktischen Umsetzung bedeutet die konsequente Rückführung eines jeden Datenobjektes auf das atomare Datenobjekt, daß auch sämtliche weiteren einfachen Datenobjekte, d.h. auch solche von nicht zusammengesetzter Struktur, für jedes Programm neu konstruiert werden müssen.

1. Atomare Datenobjekte

Formal eingeführt ist im Plankalkül ausschließlich das atomare Datenobjekt, welches aus dem einzelnen Bit besteht und demzufolge nur die boolschen Werte ‘L’=’true’ und ‘0’=’false’ annehmen kann. Aus diesen atomaren Datenobjekten lassen sich sämtliche weiteren benötigten Datenobjekte

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

durch systematisches Zusammensetzen aufbauen. Das atomare Datenobjekt wird im Plankalkül als *Ja-Nein-Wert* bezeichnet.

2. Standard-Datenarten

Zusätzlich zu den atomaren Datenobjekten werden im Plankalkül noch weitere einfache Datenobjekte verbal eingeführt, die als Angaben- bzw. Datenarten bezeichnet werden und mit dem Buchstaben ‘A’ gekennzeichnet sind.

Dabei sind folgende Standard- oder einfache Datenarten definiert:

A8	Zahl allgemein, ohne besondere Festlegung
A9	positive ganze Zahl
A10	positive bzw. negative ganze Zahl
A11	positive rationale Zahl
A12	positive bzw. negative rationale Zahl
A13	komplexe Zahl

Tabelle 2: Standard-Datenarten des Plankalküls

Diese Standard-Datenarten können zwar wiederum auf das atomare Datenobjekt zurückgeführt werden, sie werden aber innerhalb des Plankalküls als einfache, d.h. nicht weiter zerlegbare Datenobjekte angesehen und damit dem atomaren Datenobjekt gleichgestellt.

Wie beim atomaren Datenobjekt ist es möglich, aus diesen einfachen Datenarten beliebig komplex aufgebaute zusammengesetzte Datenarten zu erzeugen. Eine derartige Definition muß nicht für jedes einzelne Programm erfolgen, in denen die entsprechende Datenart verwendet werden soll. Vielmehr ist es ausreichend eine zusätzliche Datenart im Rahmen einer Programmgruppe einmalig zu definieren, womit diese dann für sämtliche darin enthaltenen Programme zugänglich ist.

5. ***Zusammengesetzte Datenobjekte***

Mit dem Begriff der *Struktur einer Angabe* wird im Plankalkül der komponentenmäßige Aufbau eines zusammengesetzten Objektes verstanden, was der heutzutage gebräuchlichen Verwendung des Begriffes *Datenstruktur* entspricht.

1. Kennzeichnung von Datenstrukturen

Die Kennzeichnung einer Datenstruktur und deren Komponenten erfolgt durch ein "*einfaches ‘Normiertes Code-Verfahren’* [...]*, in dem die Komponenten einer jeden einzelnen Struktur, Unterstruktur usw. für sich numeriert werden. Bei zusammengesetzten Strukturen erhalten wir dann einen mehrstufigen Zifferncode, womit jede beliebige einfache oder zusammengesetzte Komponente herausgegriffen werden kann. Auch die Komponentenbezeichnung hat ihre Struktur, sie besteht nämlich aus einem n-Tupel von ganzen Zahlen, welche durch Punkte untereinander getrennt sind."* Diesem Zifferncode wird das Strukturkennzeichen ‘S’ vorangestellt.

Der Plankalkül kennt einige wenige *Standard-Datenstrukturen*, für deren Kennzeichnung ein *Strukturenkalkül* entwickelt wurde. Da darüber hinaus aber eine Vielzahl an Datenstrukturen frei definiert werden kann, ist die Einführung von variablen Strukturzeichen erforderlich, die es ermöglichen, auch beliebig komplexe Datenstrukturen und deren Komponenten eindeutig zu kennzeichnen.

1. Standard-Datenstrukturen

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Zur Kennzeichnung des Aufbaus der Standard-Datenstrukturen wird im Plankalkül ein Strukturenkalkül eingeführt, bei dem das Strukturkennzeichen ‘S’ mit einer feststehenden Kennzahlenkomponente verknüpft wird:

S0	boolscher Wert, d.h. ein atomares Datenobjekt
S1	eine Folge von boolschen Werten
S1.n	eine Folge von *n* boolschen Werten
S1.4	Tetrade, d.h. 4 Bit

Tabelle 3: Standard-Datenstrukturen

2. Variable Datenstrukturen

Um die Kennzeichnung von beliebigen, d.h. variablen, Datenstrukturen zu ermöglichen, werden im Plankalkül die *variablen Strukturkennzeichen* σ und τ eingeführt, die jeweils eine beliebige Struktur des zu kennzeichnenden Datenobjektes darstellen. Dabei beziehen sich σ bzw. τ innerhalb eines Programmes immer auf dieselbe (variable) Strukturdefinition.

Es ergeben sich folgende zusätzliche Komponenten des Strukturenkalküls:

σ	Zeichen für beliebige Strukturen
$2 \times \sigma$	Paar von Werten, bei dem beide Elemente des Paares von gleicher Struktur sind
(σ, τ)	Paar von Werten, dessen erstes Element die Struktur σ und dessen zweites Element die Struktur τ hat
$n \times \sigma$	Reihe von Werten der Struktur σ , die im Plankalkül als *Liste* bezeichnet wird
$m \times 2 \times \sigma$	Paarliste
$2 \times n \times \sigma$	ein Paar von Listen
$m \times n \times \sigma$	Matrix mit *m* Zeilen und *n* Spalten der Struktur σ
(S0,S1.n)	Struktur zur Darstellung einer ganzen Binärzahl, bei der das Vorzeichen durch S0 und die *n* Binärziffern durch S1.n dargestellt werden

Tabelle 4: Variable Datenstrukturen

3. Dynamische variable Datenstrukturen

Eine besondere Form der variablen Datenstrukturen stellen die dynamischen Strukturen dar, bei denen weder die Anzahl an Elementen noch deren Struktur festgelegt sind. Dafür wird ein *allgemeines beziehungsloses Leerstellenzeichen* □ eingeführt, welches stellvertretend für eine ganze Zahl zur Festlegung der Anzahl an Elementen einer Struktur herangezogen wird.

Im Gegensatz zu den variablen Strukturkennzeichen σ und τ , besteht keine Beziehung zwischen verschiedenen Leerstellenzeichen □ innerhalb eines Programmes, d.h. diesen können durchaus unterschiedliche Werte zugeordnet werden.

Der Strukturenkalkül läßt sich damit um folgende Elemente erweitern:

$\square \times \sigma$	allgemeinstes Strukturzeichen einer Liste, bei dem Anzahl und Struktur der Elemente variabel gehalten sind

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

$\square \times 2\sigma$	Paarliste, bei der die Glieder der einzelnen Paare von der gleichen Struktur σ sind
$\square \times (\sigma, \tau)$	Paarliste, bei der die Vorderglieder von der Struktur σ und die Hinterglieder von der Struktur τ sind

Tabelle 5: Dynamische variable Datenstrukturen

Der Unterschied in der Verwendung des allgemeinen Leerstellenzeichens □ und der Variablen *m* geht sogar nach Auffassung von Konrad Zuse nicht eindeutig aus dem Plankalkül hervor: *" '□' steht hier stellvertretend für eine ganze Zahl, jedoch nicht für ein beliebiges Zeichen. An sich bezeichnet in der Auffassung des PK* [Plankalküls] *bereits die Form* $\mathfrak{m} \times \sigma$ *eine dynamische Struktur, wobei* m *variabel sein kann."*

Ein sinnvoller Unterschied in der Interpretation bei der Verwendung dieser beiden Zeichen könnte aber z.B. darin bestehen, daß bei einer Datenstruktur der Form $\mathfrak{m} \times \sigma$, während der Programmbearbeitung die Anzahl der Strukturelemente konstant mit *m* festgelegt ist. Bei der Verwendung von $\square \times \sigma$ kann sich die Anzahl der Strukturelement während eines Programmdurchlaufes verändern, d.h. die Größe der Datenstruktur wird dynamisch angepaßt.

2. Definition von Datenstrukturen

Die Erzeugung von komplexen Datenstrukturen erfolgt im Plankalkül durch das Zusammensetzen der einzelnen Strukturelemente. Die erstmalige Erzeugung einer Datenstruktur erfolgt durch eine entsprechende Definitionsangabe innerhalb einer Programmgruppe. Ein Objekt einer derartig definierten Datenstruktur kann dann innerhalb eines Programmes durch Zuweisung erzeugt werden.

1. Konstruktion von Datenstrukturen durch das 'Paket-Prinzip'

Bei der erstmalige Definition einer Datenstruktur erfolgt dies nach dem sogenannten *Paket-Prinzip*. Dabei werden, ausgehend von den einfachen Datenobjekten und Standard-Datenarten, diese zu immer komplexeren Datenobjekten (die Pakete) zusammengefaßt. Diese bereits zusammengesetzten Datenobjekte können dann wiederum in Datenobjekte der nächsthöheren Strukturstufe integriert werden. Die Schachtelungstiefe derartig erzeugter Datenstrukturen ist beliebig.

Die unter Verwendung des Paket-Prinzipes erzeugten Datenstrukturen repräsentieren eine bestimmte Baumstruktur, in der nur an den Blattenden Datenwerte enthalten sind.

2. Strukturgleichungen zur Definition von Datenstrukturen

Die Definition von Datenstrukturen erfolgt im Plankalkül durch die Verwendung von *Strukturgleichungen*. Dabei wird in Form einer Gleichung die Art der Strukturzusammensetzung einer neuen Datenstruktur aus bereits bestehenden Datenstrukturen definiert. So setzt sich z.B. eine Bitkette aus einer Folge von n atomaren Datenobjekten mit dem Strukturkennzeichen S0 zusammen. Diese Bitkette wird durch die Strukturgleichung $\mathrm{S1.n} = \mathfrak{n} \times \mathrm{S0}$ definiert.

Ebenso wie zusammengesetzte Datenarten müssen komplexe zusammengesetzte Datenstrukturen nur einmal innerhalb einer Programmgruppe definiert werden. Damit können sie von jedem beliebigen Programm dieser Programmgruppe verwendet werden.

3. Zugriff auf Komponenten von Datenstrukturen

Der Zugriff auf ausgewählte Komponenten einer zusammengesetzten Datenstruktur erfolgt durch die explizite Angabe eines Selektors in der Index-Zeile.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Dieser Selektor ist mehrstufig, d.h. für jede Strukturebene der Datenstruktur wird eine Kennziffer vergeben, die den Pfad durch den Strukturbaum anzeigt. Diese Ziffern werden durch Punkte getrennt aneinandergesetzt, wobei die Reihenfolge der Strukturebenen von links absteigend angeordnet sind. Innerhalb einer Strukturebene werden sämtliche darin enthaltene Komponenten, von links aufsteigend, durchnumeriert.

Auf diese Art und Weise kann auf jeden beliebiges Element einer beliebig komplex aufgebauten Datenstruktur zugegriffen werden.

4. Rekursive Datenstrukturen

Mit den im Plankalkül verwendeten Strukturgleichungen zur Definition von Datenstrukturen ist es nicht möglich, diese rekursiv aufzubauen.

Eine derartige Konstruktion wurde aber auch von Konrad Zuse, ebenso wie die rekursive Programmierung, abgelehnt: *"Nach Aussage Zuses war beim Entwurf des PK auch nicht an die Einführung von rekursiv aufgebauten Strukturgleichungen gedacht. Der Wunsch nach dem Einbau einer solchen Möglichkeit in den PK* [Plankalkül] *wurde erst später an seinen Autor herangetragen."*

Die Möglichkeiten rekursiver Programmaufrufe sind im Plankalkül hingegen gegeben, wie noch aufgezeigt werden wird (vgl. Kapitel 3.11.3).

6. ***Datentypen***

Neben den Datenarten und Datenstrukturen kennt der Plankalkül noch den Begriff der *Datentypen*. Die Datentypen des Plankalküls werden allerdings nicht in dem Sinne verwendet, wie dies nach heutigem Sprachgebrauch üblich ist. Wie bereits ausgeführt werden derartige Datentypen im Plankalkül unter dem Begriff Datenart verwendet.

Die Datentypen des Plankalküls werden durch 'T' und einen nachfolgenden Index bezeichnet und dienen der Verdeutlichung der Semantik von Objekten einer bestimmten Datenart. Dies kann vor allem dann sinnvoll sein, wenn identisch aufgebauten Strukturen Daten mit unterschiedlicher Bedeutung zugeordnet werden, wie z.B. x- und y-Koordinaten. Auch wenn es im allgemeinen nicht notwendig erscheint, eine entsprechende semantische Unterscheidung durchzuführen, kann im Plankalkül bei Bedarf eine entsprechende Typkennzeichnung eingeführt werden.

Von der Verwendung von Datentypen wird in den Beispielen des Plankalküls allerdings nur wenig Gebrauch gemacht, so daß ihre Einführung letztendlich überflüssig erscheint.

7. ***Anweisungen***

Die im Programmablauf auszuführenden Rechenanweisungen sind in der Grundzeile enthalten. Für die Darstellung der in diesen Anweisungen verwendeten Rechenoperationen wird die aus der Mathematik bekannte Formelnotation verwendet.

Darüber hinaus kennt der Plankalkül die *zusammengesetzte* und die *bedingte Anweisung* und als weitere Operation die *Zuweisung*. Außerdem besteht die Möglichkeit zur Formulierung von *Wiederholungsanweisungen*, bei denen mit dem *variablen Schlußzeichens FIN* eine bemerkenswerte Form des Schleifenabbruchs zur Verfügung steht.

1. Die Zuweisung '⇒'

Die Zuweisung wird im Plankalkül durch das Zeichen '⇒' definiert und als *Rechenplangleichung* bezeichnet. Sie stellt die wichtigste Konstruktion für den Aufbau von Programmen im Plankalkül dar. Die Interpretation der einzelnen Bestandteile einer Zuweisung entspricht der in der Mathematik üblichen Notation. Dabei steht auf der linken Seite der Zuweisung der zu errechnende Ausdruck und auf der rechten Seite das Datenobjekt, der das errechnete Ergebnis zugewiesen werden soll.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Die folgenden Beispiele sollen die Funktionsweise der Zuweisung im Plankalkül verdeutlichen:

V: 0 ⇒ 1 S: 1.n 1.n	Dem Zwischenwert Z_1 wird der Wert des Zwischenwertes Z_0 zugewiesen.
V: Z 1 ⇒ Z 1 S: 1.n 1.n 1.n	Der ganzzahlige Zwischenwert Z_1 wird um '1' erhöht.
V: (V, V) ⇒ R 0 1 0 S: σ σ 2σ	Die Datenstrukturen V_0 und V_1 werden im dem Resultatwert R_0 zu einer Liste von Paaren zusammengefaßt.

Beispiel 3: Funktionsweise der Zuweisung im Plankalkül

2. Das variable Schlußzeichen 'FIN'

Das variable Schlußzeichen FIN^i (mit i = 1, 2, ..., n) ist die einzige im Plankalkül enthaltene Sprunganweisung. Sie ermöglicht es, eine bestimmte Anzahl von Strukturebenen eines Programmes zu überspringen, wobei es sich dabei ausschließlich um Vorwärtssprünge handelt. Der Grad i gibt dabei die Anzahl der beim Verlassen einer Wiederholungsanweisung zu überspringenden Strukturebenen wieder. FIN^i wird fast ausschließlich in Verbindung mit Wiederholungsanweisungen verwendet.

Die durch FIN^i gekennzeichneten Sprünge haben folgende Bedeutung:

FIN	wird im Anweisungsteil eines Programmes aufgerufen, wenn die Ausführung der folgenden Anweisung für das zu errechnende Ergebnis sinnlos ist. Es bedingt den Sprung an das Ende der entsprechenden zusammengesetzten Anweisung.
FIN oder FIN^1	bedingt den Sprung an das Ende der zu wiederholenden Anweisung. Die Wiederholungsanweisung wird allerdings noch nicht verlassen, sondern es wird mit dem nächsten Schleifendurchlauf fortgesetzt.
FIN^2	bedingt das Verlassen der Wiederholungsanweisung. Es wird mit der auf die Wiederholungsanweisung folgenden Anweisung im Programmablauf fortgesetzt.
FIN^i, mit i>2	wird nur bei geschachtelten Wiederholungsanweisungen verwendet und stellt das Überspringen (von innen nach außen) der dem Grad i entsprechenden Schachtelungstiefe dar. Der Schachtelungsgrad i kann nur einen der maximalen Schachtelungstiefe entsprechenden Wert annehmen.

Tabelle 6: Bedeutung des variablen Schlußzeichens FIN^i

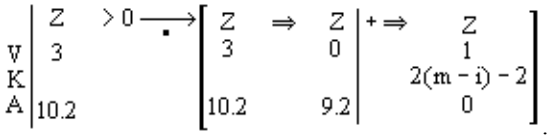
In den gängigen Programmiersprachen kann die Bedeutung des einfachen Schachtelungsgrades FIN^1 durch eine entsprechenden if-Abfrage ausgedrückt werden. FIN-Anweisungen mit höheren Schachtelungsgraden sind hingegen nur noch im Zusammenhang mit einem 'unsauberen' Programmierstil ausdrückbar, z.B. durch die Verwendung von 'goto'.

Das variable Schlußzeichen FIN ermöglicht einen strukturierten Programmierstil, auch wenn der jeweils erforderliche Schachtelungsgrad nicht immer intuitiv festgelegt werden kann.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

3. Zusammengesetzte Anweisungen

   Zusammengesetzte Anweisungen bestehen aus einer Folge von atomaren Anweisungen, d.h. aus Anweisungen, die sich nicht weiter zerlegen lassen. Diese atomaren Anweisungen werden durch einen senkrechten Strich im Programmablauf oder durch den Übergang zu einer neuen Programmzeile voneinander getrennt. Die Verbindung mehrerer atomarer Anweisungen zu einer zusammengesetzten Anweisung erfolgt durch eine entsprechende Klammerung:



Beispiel 4: zusammengesetzte Anweisung

   Die Verarbeitung dieser Anweisungen erfolgt sequentiell.

4. Die bedingte Anweisung ‘$\overset{B}{\cdot}\!\!\longrightarrow A$’

   Bedingte Anweisungen haben im Plankalkül die allgemeine Form $\overset{B}{\cdot}\!\!\longrightarrow A$ und entsprechen der einfachen if-Abfrage ‘if B then A end’. Die Anweisung A wird nur dann ausgeführt, wenn die Bedingung B den boolschen Wert ‘true’ liefert. Anderenfalls wird die Anweisung A übersprungen und der Programmablauf mit der daran anschließenden Anweisung fortgesetzt. Die in gängigen Programmiersprachen übliche Alternative zu einer Bedingung, die durch die else-Verzweigung einer if-Abfrage zum Ausdruck kommt, ist im Plankalkül nicht vorgesehen.

   Bedingte Anweisungen können mehrfach ineinander verschachtelt sein, wobei die einzelnen bedingten Ausdrücke durch Klammerung voneinander abgetrennt werden müssen.

5. Die Wiederholungsanweisung ‘W’

Wiederholungsanweisungen werden mit einem ‘W’ gekennzeichnet und haben die Form

$$W\left[\begin{matrix} B & \overset{}{\cdot}\!\!\longrightarrow & A \\ \overline{B} & \Rightarrow & FIN^2 \end{matrix}\right],$$

wobei der Anweisungsteil A nur solange ausgeführt wird, wie die Bedingung B den Wert ‘true’ liefert. Ist dies nicht der Fall erfolgt der Abbruch der Wiederholungsanweisungen durch das Verlassen mit der Sprunganweisung FIN^2. Durch diese wird der Sprung auf die Struktuebene außerhalb der Wiederholungsschleife erreicht.

Da im Plankalkül explizit vorausgesetzt wird, daß jede Wiederholungsanweisung mit der entsprechenden Abbruchanweisung $\overline{B} \Rightarrow FIN^2$ endet, kann diese weggelassen werden.

Damit hat die Wiederholungsanweisung dann die Form

$$W\left[B \overset{}{\cdot}\!\!\longrightarrow A\right].$$

Es besteht die Möglichkeit, mehrere dieser Wiederholungsanweisungen ineinander zu verschachteln. Dabei kann durch die entsprechende Verwendung des variablen Schlußkennzeichens FIN^i das Verlassen einer Wiederholungsanweisung auch über mehrere Schachtelungsebenen hinweg initiiert werden. Für den Fall, daß bei geschachtelten Wiederholungsanweisungen verschiedene Laufvariable auf unterschiedlichen Schachtelungsebenen auftreten, müssen diese durch die Angabe eindeutiger Indizes in der V-Zeile für den Variablenindex gekennzeichnet werden.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Der Plankalkül kennt darüber hinaus unterschiedliche Formen von Wiederholungsplänen, welche über fest verankerte Laufvariablen verfügen. Damit können bei den verschiedenen Durchläufen einer Wiederholungsanweisung die auszuführenden Rechenanweisungen variiert werden. Diese Wiederholungspläne entsprechen damit der do-while-Anweisung in gängigen Programmiersprachen.

8. *Der Aufbau von Programmen*

Programme werden im Plankalkül als *Rechenpläne* bezeichnet. Jedes Programm wird durch die Angabe einer eindeutigen Programmbezeichnung und einen Spezifikationsteil eingeleitet, in dem die Eingangs- und Resultatparameter des Programmes definiert werden. In dem anschließenden Anweisungsteil wird die zu bearbeitende Rechenvorschrift durch eine beliebige Folge von einzelnen Anweisungen ausgedrückt.

1. Programmparameter

Jedem Programm werden ein oder mehrere *Eingabe- und Resultatparameter* zugewiesen. In der Originalfassung des Plankalküls sind keine parameterlosen Programme vorgesehen, diese sind aber prinzipiell im Plankalkül formulierbar. Dies kann zum Beispiel für die Berechnung von unveränderlichen Werten innerhalb eines Unterprogrammes der Fall sein. Das Ergebnis wird dann über den Resultatparameter an das aufrufende Programm zurückgegeben.

1. Eingabeparameter

Eingabeparameter entsprechen im Plankalkül den formalen Parametern eines Programmes. Dabei müssen sämtliche Werte, die die Berechnung einer Formel beeinflussen können, an das Programm übergeben werden. Der Plankalkül kennt demzufolge keine globalen Parameter, sondern ausschließlich lokale Parameter. Demzufolge kennt der Plankalkül auch keine Seiteneffekte, da diese nur bei der Verwendung von globalen Variablen auftreten können.

2. Zwischenwerte

Zwischenwerte treten nur lokal innerhalb eines Programmes auf. Sie werden vor allem in zyklischen Programmstrukturen zur Zwischenspeicherung von Werten eingesetzt. Dabei ist im Gegensatz zu den übrigen Programmparametern die Veränderung des aktuellen Wertes während eines Programmablaufes zulässig. Eine Veränderung der Eingangswerte innerhalb eines Programmablaufes hingegen nicht. Ebenso werden die Resultatparameter ausschließlich am Ende des Programmdurchlaufes mit den errechneten Werten gefüllt.

Sollen Zwischenwerte als Eingangsparameter für einen Unterprogrammaufruf oder als Resultatparameter des aktuellen Programms dienen, so muß dies durch eine explizite Zuweisung des Zwischenwertes an den entsprechenden Eingangs- oder Resultatparameter erfolgen.

3. Resultatparameter

Resultatparametern kann ebenso wie Programmen und Unterprogrammen eine Bezeichnung und auch das allgemeine Plangruppenzeichen Δ zugewiesen werden. Dies stellt dann eine besondere Form eines Unterprogrammaufrufes dar. So kann z.B. der Resultatwert R_0 des Programmes P17 außerhalb von P17 durch die Angabe von $\overset{R17}{0}$ verwendet werden. Die Wertzuweisung an R_0 erfolgt in diesem Fall durch einen Aufruf von P17 als Unterprogramm des aktuellen Programms.

4. Konstanten

Konstanten sind im Plankalkül entsprechend der mathematischen Verwendung Werte, denen einmalig und für den gesamten Propgrammablauf ein fester, unveränderbarer Wert zugewiesen wird. Die Darstellung von Konstanten kann auf verschiedene Weise erfolgen.

Zum einen können Konstanten als Bitketten dargestellt werden. Diese Bitketten werden dabei

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

entweder in der Form ‘L’=’true’ und ‘0’=’false’ abgebildet oder in der Form ‘+’=’true’ und ‘-‘=’false’. Die Verwendung der zweiten Darstellungsform kann allerdings zu Verständnisschwierigkeiten der zu berechnenden Formel führen, da im Plankalkül keinerlei Unterscheidung im Hinblick auf die Verwendung der entsprechenden mathematischen Operationszeichen ‘+’=’plus’ und ‘-‘=’minus’ vorgenommen wird.

Darüber hinaus können Konstanten entsprechend der in der Mathematik üblichen Notation auch mit Ziffern bezeichnet werden, um konstante Zahlenwerte zu bezeichnen.

Eine weitere Darstellungsform von Konstanten kann durch die Bezeichnung ‘C’ und die Zuordnung eines Index erfolgen. Es wird allerdings nicht erläutert, welchen Wert eine entsprechend bezeichnete Konstante repräsentiert.

2. Der Randauszug

Der Spezifikationsteil eines Programmes, der sogenannten Randauszug, wird dem eigentlichen Programm vorangestellt und beinhaltet die Struktur- und Artdefinitionen der im Anweisungsteil verwendeten Eingangs- und Resultatparameter, die als *Randwerte* bezeichnet werden. Die Struktur- und Artdeklaration dieser Randwerte erfolgt in Form einer Zuweisung, die sich nur durch ein vorangestelltes ‘R’ in der Hauptzeile von den im Anweisungsteil verwendeten Zuweisungen unterscheidet.

Links vor den Randauszug wird das *Plankennzeichen* gesetzt, mit welchem dem Programm eine eindeutige Programmbezeichnung zugeordnet wird.

So kann z.B. einem beliebigen Programm P9.18 der folgende Randauszug zugeordnet sein:

```
P9.18   | R  (V)  ⇒  (R,  R)
      V |    0        0   1
      A |   9.2       9.2  0
```

Beispiel 5: Randauszug

Dabei stellt V0 einen Eingangs- und R0 und R1 die beiden Resultatparameter des Programmes dar, denen im Verlauf des folgenden Anweisungsteils Werte der angegebenen Strukturen zugewiesen werden müssen.

Da der Randauszug kein fester Bestandteil eines Programmes, sondern diesem nur zugeordnet ist, kann derselbe Randauszug verschiedenen Programmen zugeordnet werden. Es müssen nur die Strukturen der im Randauszug definierten Eingangs- und Resultatparameter mit denen des Programmes übereinstimmen.

3. Anweisungsteil für Rechenvorschriften

Die im Anweisungsteil enthaltenen Rechenvorschriften stellen das eigentliche Programm dar. Die einzelnen Anweisungen werden dabei entsprechend ihrer Bearbeitungsreihenfolge sequentiell ausgeführt. Dabei werden einzelne Anweisungen durch senkrechte Striche oder durch den Übergang auf eine neue Zeile voneinander getrennt.

Ein Begrenzungszeichen für Programmabschnitte oder das gesamte Programm, wie es z.B. durch ‘begin-end’ ausgedrückt werden kann, gibt es im Plankalkül nicht. Da der Plankalkül allerdings in keiner implementierfähigen Version vorliegt, ist das Fehlen eines solchen Begrenzungszeichens auch nicht relevant. Vielmehr werden sämtliche Programmbeispiele des Plankalküls in den verschiedenen Veröffentlichungen durch das Einfügen von Absätzen und Kommentartexten voneinander getrennt.

4. Kommentare

Im Plankalkül werden Kommentare als wichtiges Hilfsmittel zur Programmdokumentation eingesetzt, wobei

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

diese allerdings nicht gesondert gekennzeichnet sind. Es wird auch keine einheitliche Darstellungsform verwendet. Vielmehr können Kommentare je nach Bedarf an beliebiger Stelle vor, zwischen oder nach den Rechenanweisungen eines Programmes angegeben werden. Damit erfolgt die Verwendung von Kommentaren im Plankalkül entsprechend der Anwendung von Kommentaren in der Mathematik.

9. ***Bildung von Programmgruppen***

Der Plankalkül kennt die Möglichkeit, einzelne Programme und Unterprogramme zu Programmgruppen unter einem gemeinsamen Kennzeichen zusammenzufassen. Diese Programmgruppen werden als *Plangruppen* oder *Plangebäude* bezeichnet. Dabei werden Programmen, die in einem gemeinsamen Kontext verwendet werden, zusammengefaßt. Dies können z.B. sämtliche Programme der Schachtheorie sein.

Die Zugehörigkeit eines Programmes zu einer Programmgruppe ist durch die Übernahme eines entsprechenden Gruppenkennzeichens in die Programmbezeichnung gekennzeichnet. Bei dessen Auswahl ist zu beachten, daß es nur einmal vergeben werden darf. Nur dadurch kann die eindeutige Zugehörigkeit eines Programmes zu einer Programmgruppe gewährleistet werden. Solange ausschließlich Programme aus einer Programmgruppe verarbeitet werden, ist die Angabe des eindeutigen Plangruppenzeichens nicht erforderlich. Dieses kann dann durch das variable Plangruppenzeichen Δ ersetzt werden. Nur bei einem Wechsel der Programmgruppe oder bei der Verarbeitung von Programmen aus verschiedenen Programmgruppen muß das eindeutige Gruppenkennzeichen verwendet werden.

An das Gruppenkennzeichen wird, abgetrennt durch einen Punkt, die eigentliche Programmbezeichnung angehängt. Für die Bezeichnung von Programmen und Programmgruppen sind ausschließlich numerische Bezeichner zulässig.

Die folgenden Beispiele sollen die Möglichkeiten zur Kennzeichnung von Programmen verdeutlichen:

P12	Programm mit der Bezeichnung 12
P3.7	Programm mit der Bezeichnung 7 aus der Programmgruppe 3
PΔ .14	Programm mit der Bezeichnung 14 aus der Programmgruppe Δ

Beispiel 6: Kennzeichnung von Programmen und Programmgruppen

Die Einführung von Programmgruppen erschien Konrad Zuse vor allem aus Gründen der Übersichtlichkeit bei der Vergabe von Programmbezeichnungen notwendig, zumal das von ihm gewählte Verfahren der fortlaufenden Numerierung schnell unübersichtlich wird: *"Wie bereits ausgeführt, ist es jedoch nicht möglich, sämtlichen überhaupt möglichen Programmen verschiedene Codeziffern zuzuteilen. Daher werden Programmgruppen gebildet."*

Die zusätzliche Einführung des variablen Plangruppenzeichens ist allerdings nicht so ohne weiteres ersichtlich. Es scheint vielmehr kein Unterschied darin zu liegen, ob z.B. mit den Programmen P3.13, P3.14 und P3.27 gearbeitet wird, oder ob statt dessen die Bezeichnung PΔ 13, PΔ 14 und PΔ 27 angegeben wird. Gründe für dessen Einführung werden nicht angegeben.

Die Gruppierung von inhaltlich zusammengehörenden Programmen zu Plangebäuden und deren explizite Kennzeichnung durch ein Gruppenkennzeichen weist eine gewisse Ähnlichkeit mit dem heutzutage üblichen Bibliothekskonzept auf. Dabei ist es ebenfalls nicht erforderlich bei der Verwendung von Programmen deren vollständige Pfadangabe zur eindeutigen Identifizierung anzugeben, solange diese sich in demselben Verzeichnis befinden. Durch die Verwendung von Programmgruppen verfügt der Plankalkül über ein Konzept zur Strukturierung von Programmen.

Die Verwendung des Gruppenkennzeichens gilt entsprechend für Unterprogramme, da im Plankalkül jedes Programm als Unterprogramm eines anderen Programmes aufgerufen werden kann.

10. ***Unterprogramme***

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Im Plankalkül wird nicht zwischen Hauptprogrammen und Unterprogrammen unterschieden. Jedes Programm kann als Unterprogramm eines anderen Programmes dienen.

1. Der Aufruf von Unterprogrammen

   Der Aufruf von Unterprogrammen erfolgt im Plankalkül ausschließlich in der Form von Funktionsaufrufen. Nur so ist die weitere Verwendung der gelieferten Resultatparameter des Unterprogrammes im aufrufenden Programm möglich. Hingegen ist ein Programmaufruf durch das Verwenden einer Anweisung nur beim Aufruf eines Hauptprogrammmes sinnvoll, da in diesem Fall die errechneten Resultatparameter nicht weiter verwendet werden können.

   Der Aufruf und damit die Kennzeichnung eines Unterprogrammes erfolgt durch die Ersetzung des 'P' in der Programmbezeichnung durch ein 'R'. Der Aufruf eines Programmes $P9.10\binom{V}{0}$ als Unterprogramm lautet demzufolge $R9.10\binom{Z}{0}$.

   Damit wird deutlich gemacht, daß der durch den Unterprogrammaufruf gelieferte Resultatparameter für den weiteren Verlauf der Berechnung des aufrufenden Programmes benötigt wird. Als Eingangsparameter werden entsprechende Zwischenwerte des aufrufenden Programmes mitgegeben.

   Es besteht keine Beschränkung in Bezug auf die Schachtelungstiefe von Unterprogrammen. Wie bei geschachtelten Funktionsaufrufen in der Mathematik üblich, werden geschachtelte Unterprogrammaufrufe durch Klammerung strukturiert. So kann z.B. der geschachtelte Aufuf der Programme PΔ 13 und PΔ 11 aus einem Hauptprogramm heraus in der folgenden Form vorgenommen werden:

$$R\Delta 13\left(R\Delta 11\binom{Z,\ Z}{7\ \ 13}\right)$$

Beispiel 7: Schachtelung von Programmen

   Dabei dient der Resultatparameter von RΔ 11 als Eingabeparameter für RΔ 13. Die Eingabewerte Z_7 und Z_{13} für RΔ 11 sind Zwischenwerte des aufrufenden Hauptprogrammes.

   1. Der Standardaufruf

      Die Standardform eines Unterprogrammaufrufes erfolgt durch die Angabe der Programmbezeichnung und der zugehörenden Parameterliste für Eingangs- und Resultatparameter:

	R9.10	(V)	⇒	(Z,	Z)
V		0		1	2
S		1.n		1.n+1	0

Beispiel 8: Standardaufruf eines Unterprogrammes

      Als Ergebnis wird eine Datenstruktur geliefert, deren Elemente aus den einzelnen Resultatparametern bestehen. Die für die Weiterverarbeitung im aufrufenden Programm benötigten Parameter können dann aus dieser Datenstruktur selektiert werden.

   2. Der spezielle Aufruf

Eine Sonderform des Unterprogrammaufrufes besteht darin, daß bereits beim Aufruf des Unterprogrammes der zurückzuliefernde Resultatparameter angegeben wird.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Beispiel 9: Sonderform des Aufrufes eines Unterprogrammes

Durch die Angabe des Index des gewünschten Resultatparameters des aufgerufenen Unterprogrammes, wird ausschließlich dieser an das aufrufende Programm zurückgeliefert. Mögliche weitere Resultatparameter stehen damit nicht für die Weiterverarbeitung zur Verfügung.

Diese Form des Unterprogrammaufrufes ist eigentlich bereits Bestandteil der Standardform, es entfällt lediglich die Auswahl des benötigten Resultatparameters aus der entsprechenden Datenstruktur. Ein derartiger Aufruf macht dann Sinn, wenn gerade diese zusätzliche Anweisung im Programmablauf vermieden werden soll, was zur Entstehungszeit des Plankalküls aus Gründen der geringen Kapazität von Rechenanlagen durchaus sinnvoll war. Aus heutiger Sicht kann diese Sonderform entfallen, um aus Gründen der Übersichtlichkeit sämtliche Unterprogrammaufrufe einheitlich zu gestalten.

2. Sonderformen von Unterprogrammen

Neben der allgemeinen Form von Unterprogrammen, bei der jedes beliebige Programm aus einem anderen Programm heraus als Unterprogramm aufgerufen werden kann, sind im Plankalkül noch zwei Sonderformen von Unterprogrammen, nämlich die *'echten' Unterprogramme* und die *'offenen' Unterprogramme*, vorgesehen.

1. 'Echte' Unterprogramme

   Als 'echte' Unterprogramme werden solche Programme bezeichnet, bei denen von vornherein feststeht, daß sie ausschließlich als Unterprogramme für ein ganz bestimmtes Hauptprogramm dienen sollen. Demzufolge wird diesen Unterprogrammen das gesonderte Kennzeichen 'PZ' in Verbindung mit dem kennzeichnenden Zifferncode des Hauptprogrammes zugeordnet. Die Kennzeichnung durch die Verwendung eines zusätzlichen 'Z' in der Bezeichnung erfolgt analog der Bezeichnung von innerhalb eines Programmes auftretenden Zwischenwerten, die mit 'Z' bezeichnet werden.

   Bis auf diese besondere Bezeichnung unterscheiden sich die 'echten' Unterprogramme in ihrer Verwendung in keiner Weise von den sonstigen Programmen bzw. Unterprogrammen. So verfügen sie z.B. auch über einen eigenen Randauszug.

2. 'Offene' Unterprogramme

Einen weiteren Sonderfall bei der Behandlung von Unterprogrammen stellen die 'offenen' Unterprogramme dar.

Grundsätzlich stellen Programme im Plankalkül in sich streng abgeschlossene modulare Einheiten dar. Es kann aber auch angebracht sein, bestimmte Programmabschnitte, welche eine nicht abgeschlossene Folge von Anweisungen enthalten, zu einem 'offenen' Unterprogramm zu deklarieren und damit aus dem Hauptprogramm herauszuziehen. Dies kann z.B. dann von Vorteil sein, wenn bestimmte Anweisungsfolgen eines Programmes wiederholt und an verschiedenen Stellen im Programmablauf ausgeführt werden müssen. Anstatt diese Anweisungsfolgen redundant in den Programmablauf aufzunehmen kann der Aufruf des entsprechenden offenen Unterprogrammes gesetzt werden.

'Offene' Unterprogramme werden mit der Bezeichnung 'U' und einem Zifferncode bezeichnet. Sie sind dabei ebenfalls ausschließlich einem einzigen Hauptprogramm zugeordnet, in welchem sie beliebig verwendet werden können.

Beim Aufruf eines 'offenen' Unterprogrammes ist zu beachten, daß diese über keinen eigenen Randauszug verfügen, da sie keine in sich abgeschlossenen Module bilden. Sie müssen vielmehr direkt

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

in die Umgebung des aufrufenden Programmes integriert werden, was vor allem bei der Verwendung und Bezeichnung von Eingabe-, Zwischen- und Resultatwerten zu beachten ist. Diese müssen zwischen aufrufendem Programm und 'offenem' Unterprogramm identisch sein.

3. Rekursive Programmaufrufe

Eine rekursive Programmierung ist im Plankalkül realisierbar, da jedes Programm als Unterprogramm eines anderen Programmes aufgerufen werden kann. Damit ist eine beliebige Schachtelung von Programmaufrufen möglich. Da sämtliche Programme des Plankalküls ausschließlich auf lokalen Variablen arbeiten und der Plankalkül deshalb keine Seiteneffekte kennt, können auch keine Deklarationskonflikte bei rekursiven Programmaufrufen auftreten. Als einzige notwendige Veränderung im Plankalkül muß die Variationsmöglichkeit des Randauszuges eines Programmes zugelassen werden, da bei jedem rekursiven Aufruf eines Programmes dessen Randwerte verändert werden müssen.

Ein Beispiel für den rekursiven Einsatz eines Programmes stellt die Lösung von n! dar. Die rekursive Definition von n! ergibt sich wie folgt:

$$0! = 1$$
$$n! = n \times (n-1)!$$

Das zugehörige Plankalkül-Programm PΔ 19.2 hat dann die folgende Form:

Randauszug:

	R (V) ⇒ R
V	0 0
A	9 9

Programm:

V =0 ⟶ 1 ⇒ R
0 0

V >0 ⟶
0

V	×	RΔ19.2	(V	-1) ⇒	R
0		0	0		0
V	-1	⇒	V		
0			0		

Beispiel 10: Programm PΔ 19.2 zur rekursiven Lösung von n!

Die Originalversion des Plankalküls enthält allerdings keine rekursiven Programmaufrufe und auch noch Ende der 70er Jahre erklärte Konrad Zuse, daß die iterative Programmlösung der Rekursion vorzuziehen sei, selbst wenn dadurch der Programmausdruck komplexer werden sollte: *"Die allgemeine Benutzung dieses Verfahrens* [der Rekursion] *würde es erübrigen, für iterative Programme besondere syntaktischen Formulierungen einzuführen (W-Pläne* [= Wiederholungsanweisungen; Anm. d. Verf.]*). Das Arbeiten mit besonderen iterativen Programmen ist jedoch in den meisten Fällen wesentlich eleganter, wodurch die verhältnismäßig einfachen syntaktischen Erweiterungen gerechtfertigt sind."*

1. Zur Veröffentlichung des Plankalküls

Obwohl bereits 1945 verfaßt, wurde der Plankalkül erst 1972 erstmalig in der vollständigen Version veröffentlicht. In den dazwischenliegenden 27 Jahren hat es zwar einige Versuche der Veröffentlichung von Seiten Konrad Zuses, in Form von Vorträgen und Fachbeiträgen, gegeben. Das Interesse der Fachwelt konnte der Plankalkül dadurch allerdings nicht gewinnen.

Als Grund für dieses mangelnde Interesse an seinen Arbeiten nimmt Zuse selbst an, daß die Zeit dafür noch nicht 'reif' war. Die Herausbildung einer Fachwelt, welche sich für eine Problemstellung, wie sie der Plankalkül darstellte, interessieren konnte, lag noch in den Anfängen. Hinzu kam eine strenge Fixierung der meisten an der Entwicklung von Rechenanlagen beteiligten europäischen Wissenschaftler auf die Forschungsergebnisse in den USA. Diese galten schließlich noch für viele Jahre als das Ursprungsland des Computers.

Weitere Gründe für das Desinteresse könnten in der Person Zuses gelegen haben. Wie er selbst angibt, war er *"der*

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Fachwelt zwar nicht unbekannt, galt aber doch mehr oder weniger als Praktiker." Dadurch schien ihm bis in die 60er Jahre die wissenschaftliche Anerkennung seiner Arbeiten verwehrt worden zu sein, was sich besonders bei der in den 50er Jahren einsetzenden Geschichtsschreibung zur Entwicklung des Computers bemerkbar machte. Diese war zunächst auch in Europa ausschließlich auf die USA konzentriert. Besonders kränkend muß für Konrad Zuse gewesen sein, daß sich seiner Aussage nach sogar solche Wissenschaftler an einer derartig verfälschenden Geschichtsschreibung beteiligt haben, die ihn persönlich kannten. Dies war zum einen Prof. Walther vom IPM Darmstadt, zu dessen Institut Konrad Zuse während des Krieges eines engeren Kontakt pflegte, und zum anderen Prof. Stiefel und seine Mitarbeiter von der ETH Zürich, die aufgrund ihrer Arbeit mit der Z4 bis in die 50er Jahre hinein mit Konrad Zuse zusammengearbeitet haben.

Erst in Folge der 1970 erschienenen Biographie von Konrad Zuse erwachte ein gewisses Interesse auch am Plankalkül, woraus sich letztendlich die Möglichkeit zur Veröffentlichung der Originalfassung von 1945 ergab.

1. ***Gründe für die Nicht-Veröffentlichung 1945***

   Zum Entstehungszeitpunkt des Plankalküls 1945 waren die Möglichkeiten für dessen Veröffentlichung denkbar ungünstig. Zunächst einmal bestanden in der unmittelbaren Nachkriegszeit in Deutschland keine Möglichkeiten für die Veröffentlichung wissenschaftlicher Arbeiten. Erschwerend kam noch hinzu, daß es Konrad Zuse zu Kriegsende ins Allgäu verschlagen hat, so daß auch die Kontakte abgebrochen waren, die er während des Krieges zu denjenigen wissenschaftlichen Instituten aufgebaut hatten, die sich ebenfalls mit der Automatisierung von Rechenvorgängen beschäftigt hatten. Eine derartige Verbindung hätte vermutlich einen gewichtigen Einfluß auf die Veröffentlichung und auch die allgemeine Anerkennung seiner Arbeiten gehabt.

   Außerdem war der Plankalkül als Entwurf entstanden, von dem Konrad Zuse durchaus bewußt war, daß er einer grundlegenden Überarbeitung bedurfte, um einer wissenschaftlichen Bewertung standhalten zu können. Immerhin sollte die Arbeit über den Plankalkül Bestandteil seiner immer noch geplanten Dissertation werden. Eine derartige Überarbeitung war für Zuse aber zum damaligen Zeitpunkt nicht realisierbar.

   Er selbst war mit dem Wiederaufbau seiner Firma beschäftigt, so daß er sich wieder der praktischen Entwicklung seiner Rechengeräte, vor allem der Z4, widmen mußte. Andererseits verfügte er nicht über die finanziellen Möglichkeiten zur Anstellung der für die Überarbeitung des Plankalküls benötigten Mitarbeiter.

   Zusätzlich zu den inhaltlichen Mängeln des Plankalküls lagen der Entscheidung, den Plankalkül nicht zu veröffentlichen oder auch nur weiterführende Informationen darüber zu verbreiten, vermutlich auch wirtschaftliche Interessen zu Grunde. Seit er in den 30er Jahren mit der Entwicklung von automatischen Rechengeräten begonnen hatte, war er daran interessiert, seine sämtlichen Erfindungen durch Patente schützen zu lassen. Es erscheint naheliegend, daß dies zumindest in den ersten Jahren eine zusätzliche Motivation darstellte, den Plankalkül nicht zu veröffentlichen.

2. ***Mehrere Veröffentlichungsversuche 1948***

   1948 hat Konrad Zuse dann verschiedene Versuche unternommen, das sich langsam herausbildende Fachpublikum auf dem Gebiet der Rechenanlagen für den Plankalkül zu interessieren. Allerdings war keiner dieser Versuche erfolgreich.

   Im Studienjahr 1947/48 erhielt Konrad Zuse die Einladung zu einem Gastvortrag an der Universität München, im Colloquium für Formale Logik von Professor Britzelmayr. Zuse nutzte die Gelegenheit, den Plankalkül anhand des in der Sprachbeschreibung vorgestellten Beispieles zur ‘Syntaxanalyse von aussagenlogischen Ausdrücken’ vorzustellen. Dieser Vortrag stieß auf weitgehendes Unverständnis, da die Zuhörer nicht über die notwendigen Vorkenntnisse über Zuses Rechengeräte verfügten. Darüber hinaus war Professor Britzelmayr stärker inhaltlich an dem vorgestellten Beispielprogramm interessiert, als an dessen Darstellungsmethode.

   Anläßlich einer Tagung der Gesellschaft für Mathematik und Mechanik (GAMM) am
   24. September 1948 in Göttingen hielt Konrad Zuse diesen Vortrag erneut, konnte aber auch diesmal kein

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Interesse für den Plankalkül erwecken.

Im Dezember 1948 wurde dieser Vortrag dann unter dem Titel *Über den Allgemeinen Plankalkül als Mittel zur Formulierung schematisch-kombinativer Aufgaben* in der Fachzeitschrift 'Archiv der Mathematik' veröffentlicht. Auch diesmal ohne Resonanz. Wie Zuse selber bemerkte: *"Die Sache verpuffte jedoch völlig."*

Immerhin ist diese Arbeit bis heute die wohl bekannteste von Zuses Veröffentlichungen zum Plankalkül. Zumindest wird sie in Arbeiten über Konrad Zuse am häufigsten als Quelle angegeben, sofern der Plankalkül darin erwähnt wird. Da es sich bei diesen Arbeiten allerdings in der Regel um historische Betrachtungen des Lebenswerkes von Konrad Zuse handelt, findet meistens keine inhaltliche Betrachtung des Plankalküls statt.

3. *Ein weiterer Veröffentlichungsversuch 1959*

Anfang der 50er Jahre bestand eine enge Zusammenarbeit zwischen Konrad Zuse und Professor Stiefel und seinen Mitarbeitern vom Institut für angewandte Mathematik an der ETH Zürich, da dort seit 1950 die Z4 in Betrieb war. Während dieser Zeit bot sich wiederholt Gelegenheit über den Plankalkül im Zusammenhang mit der allgemeinen Einführung von algorithmischen Sprachen zu diskutieren. Aber auch diese Gespräche förderten den Bekanntheitsgrad und die Anerkennung in keiner Weise. Wie Zuse selbst zugibt, lag der Grund dafür hauptsächlich darin, daß er nicht genügend Zeit für seine wissenschaftlichen Arbeiten und deren Veröffentlichung erübrigen konnte: *"But, in order to influence this development* [of the introduction of algorithmic languages]*, it would have been inevitably necessary to spend most of my time for this purpose and to publish my ideas. So our cooperation was fading away, and each of us was going his own way."* Konrad Zuse war mit dem Aufbau seiner Firma beschäftigt.

Bis zum Ende der 50er Jahre hat es keine weiteren Veröffentlichungen zum Plankalkül gegeben. Erst 1959 erschien *Über den Plankalkül* in der neu herausgegeben Fachzeitschrift 'Elektronische Rechenanlagen'.Auf wenigen Seiten versucht Konrad Zuse darin, die wichtigsten Konzepte des Plankalküls darzustellen und anhand eines Beispiels aus der Schachtheorie zu verdeutlichen. Dabei betont er ausdrücklich, daß er den Plankalkül nicht als Alternative zu der gerade in der Entwicklung befindlichen Programmiersprache ALGOL verstanden wissen wollte. Vielmehr verfolgte er die Absicht, die im Plankalkül enthaltenen Konzepte zur Diskussion anzubieten, um auf diese Weise positive Anregungen in die allgemeine Diskussion um die Entwicklung von Programmiersprachen einfließen zu lassen: *"Ziel dieses Aufsatzes ist es nicht, den Plankalkül in der damals geschaffenen Form zu propagieren, sondern einen Diskussionsbeitrag zur Verfeinerung und Ergänzung heute bereits benutzter Formelsprachen zu geben. Gewisse Zeichen, wie z.B. das Ergibt-Zeichen, konnten sich bereits weitgehend durchsetzen."*

Aber auch mit dieser Veröffentlichung ist es Zuse nicht gelungen, ein größeres Interesse in der Fachwelt zu wecken. Der Hauptgrund hat vermutlich darin gelegen, daß in dieser Arbeit der Plankalkül kaum verständlicher dargestellt wird als in der Veröffentlichung von 1948. Andererseits war Konrad Zuse aber immer noch nicht bereit, die Originalfassung des Plankalküls von 1945 vollständig zu veröffentlichen, welche einem weitergehenden Verständnis des Plankalküls hätte dienlich sein können. Damit konnte auch mit dieser Veröffentlichung kein Interesse am Plankalkül geweckt werden.

4. *Ein letzter Versuch 1968*

Obwohl Konrad Zuse seit Ende der 40er Jahre relativ viele Arbeiten veröffentlicht hat, erscheint erst Ende der 60er Jahre eine dritte Arbeit zum Plankalkül.

Anläßlich einer Tagung der Nachrichtentechnischen Gesellschaft (NTG) über 'Teilnehmerrechensysteme' in Bad Hersfeld, dem Sitz der Zuse KG, hält Konrad Zuse einen Vortrag über die historische Entwicklung von Multiple Access-Systemen und die daraus resultierenden Anforderungen an eine dabei einzusetzende Programmiersprache. Er hebt dabei besonders hervor, daß der Plankalkül aufgrund seiner logischen Grundkonzepte besonders dafür geeignet ist, diese Anforderungen zu erfüllen: *"Der Plankalkül erscheint wegen seiner Universalität hervorragend geeignet, als Bezugssprache zwischen verschiedenen Systemen zu dienen."*

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

1968 wurde *Gesichtspunkte zur sprachlichen Formulierung in Vielfachzugriffssystemen unter Berücksichtigung des Plankalküls* in dem zugehörigen Tagungsband veröffentlicht.

Der Teil über den Plankalkül ist in dieser Arbeit allerdings noch kürzer als in den beiden vorangegangenen Veröffentlichungen, so daß es nicht verwundert, daß wiederum keine Resonanz aus der Fachwelt erfolgt. Vielmehr könnte das konsequente Beharren von Konrad Zuse auf der Universalität und den aktuellen Einsatzmöglichkeiten des inzwischen über zwanzig Jahre alten Plankalkül etwas hilflos gewirkt haben, zumal er selbst in seiner Arbeit wiederholt den historischen Hintergrund des Plankalküls betont. Demzufolge hat vermutlich niemand ernsthaft in Betracht gezogen, daß der Plankalkül von Interesse für die aktuellen Probleme der Informationstechnik sein könnte.

5. ***Erwachendes Interesse am Plankalkül ab 1970***

Erst mit der Veröffentlichung seiner Biographie *Der Computer - Mein Lebenswerk* von 1970 ist ein größeres Interesse der Fachwelt am Plankalkül erwacht.

Das darin enthaltenen Kapitel über den Plankalkül umfaßt immerhin 14 Seiten und ist damit das umfangreichste, was seit dem Entstehen des Plankalküls zu diesem veröffentlicht wurde. Zum ersten Mal beschreibt Zuse den gedanklichen Hintergrund des Plankalküls. Dadurch konnte er deutlich machen, daß dieser nicht als fixe Idee, sozusagen aus dem Nichts, entstanden ist, sondern vielmehr einen Bestandteil seiner unvollendeten Dissertation darstellt.

Die Konzepte des Plankalküls werden im Vergleich zu den vorangegangenen Veröffentlichungen ausführlich hergeleitet und anhand eines Beispiels aus der Schachtheorie erläutert.

6. ***Veröffentlichung des Plankalküls 1972***

Aufgrund der Unterstützung der Gesellschaft für Mathematik und Datenverarbeitung (GMD) wurde es dann 1972 möglich, den Plankalkül in seiner Originalfassung zu veröffentlichen. Allerdings überwog bei dieser Veröffentlichung auch von Seiten Konrad Zuses bereits das historische Interesse am Plankalkül, denn *"der Plankalkül* [wurde] *aus historischen Gründen bewußt mit Fehlern veröffentlicht"*.

Zusammen mit einem erläuterndem Kommentar zum Plankalkül von Konrad Zuse, wurde in Auszügen auch dessen unvollendete Dissertation abgedruckt. Damit wurde deutlich, daß beide Arbeiten als eine zusammengehörige Einheit zu betrachten sind.

In der Folge dieser Veröffentlichung sind dann einige Arbeiten entstanden, die sich ausführlicher mit den Inhalten des Plankalküls auseinandergesetzt und dessen Einordnung in bestehende Programmiersprachenkonzepte versucht haben.

2. Zur Rezeption des Plankalküls

Die vollständige Veröffentlichung entfachte ein kurzfristiges inhaltliches Interesse der Fachwelt am Plankalkül in den 70er Jahren, wobei die daraus resultierenden Arbeiten bereits überwiegend den Charakter von historischen Würdigungen hatten. Hierbei bilden auch die Arbeiten, die Konrad Zuse selbst zum Plankalkül verfaßt hat, keine Ausnahme, zumal bereits die Originalversion des Plankalküls in einem historischen Kontext veröffentlicht wurde.

Als erste haben sich F. L. Bauer und H. Wössner in ihrem Artikel *Zuses "Plankalkül", ein Vorläufer der Programmiersprachen - gesehen vom Jahr 1972* mit den Sprachkonzepten des Plankalküls beschäftigt. Mit Konrad Zuses eigenen Worten haben sie es damit aber auch *"*[...] *unternommen, dem Plankalkül seinen Platz in der Geschichte der Computersprachen zuzuweisen."*

Dieser Artikel wurde gleichzeitig in einer englischsprachigen Version veröffentlicht, wodurch die Existenz des Plankalküls zum ersten Mal in den USA bekannt wurde. Als Reaktion auf diesen Artikel entstand die vermutlich einzige US-amerikanische Auseinandersetzung mit dem Plankalkül, nämlich *The Early Development of Programming Languages* von D. E. Knuth und L. Trabb Pardo aus dem Jahr 1977.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

In Deutschland konnten in der zweiten Hälfte der 70er Jahre mit Unterstützung der Gesellschaft für Mathematik und Datenverarbeitung, die auch die Erstveröffentlichung des Plankalküls ermöglicht hat, der Deutschen Forschungsgemeinschaft und der Siemens AG folgende Forschungsstudien zum Plankalkül erstellt werden:

- 1975 die als Dissertation angelegte *Untersuchung des Plankalküls im Vergleich mit algorithmischen Sprachen* von Joachim Hohmann,
- die zeitgleich und in Zusammenarbeit mit J. Hohmann entstandenen *Gesichtspunkte zur Beurteilung algorithmischer Sprachen* von Konrad Zuse und
- 1977 die ebenfalls von Konrad Zuse stammende *Beschreibung des Plankalküls'*.

Diese Arbeiten werden im Folgenden ausführlicher beschrieben und auf ihre Bewertung des Plankalküls, vor allem im Vergleich mit etablierten Programmiersprachen wie ALGOL und FORTRAN, untersucht.

1. ***F. L. Bauer/H. Wössner: Zuses "Plankalkül", ein Vorläufer der Programmiersprachen - gesehen vom Jahr 1972 (1972)***

   F. L. Bauer und H. Wössner wollen ihren Artikel ausdrücklich als *"historische Würdigung dieses frühen Entwurfs einer Programmiersprache"* verstanden wissen, wobei dieser aber gleichzeitig dazu beitragen soll, den Bekanntheitsgrad des Plankalküls zu erhöhen. Dieses ist ihnen besonders eindrucksvoll durch die Veröffentlichung der englisch-sprachigen Version in den *Communications of the ACM* gelungen.

   Darüber hinaus nehmen sie die Analyse des Plankalküls zum Anlaß, sich kritisch mit dem aktuellen Stand im Bereich der Entwicklung von Programmiersprachen auseinander und diesen in Bezug zu den aktuellen Diskussionen im Zusammenhang mit der Software-Krise zu setzen. Die Untersuchung des Plankalküls dient demzufolge auch dazu, dessen Sprachkonzepte auf ihre Verwendungsmöglichkeit bei der zukünftigen Entwicklung von Programmiersprachen hin zu untersuchen.

   Um dies zu ermöglichen wurde die Plankalkül-Terminologie in 'moderne' Notation, d.h. in ALGOL 68-Notation umgesetzt. Zur Veranschaulichung wird das in Kapitel 3.1 vorgestellte Programmbeispiel zur 'Syntaxanalyse von aussagenlogischen Ausdrücken' analysiert und in ALGOL 68-Notation 'übersetzt'. Zusätzlich werden einige Beispiel-Programme aus der Schachtheorie wiedergegeben.

   1. Bestehende Kontakte zu Konrad Zuse

      F. L. Bauer hat Konrad Zuse bei dessen Vortrag im Colloquium über Formale Logik im Studienjahr 1947/48 an der TU München kennengelernt und bei dieser Gelegenheit zum ersten Mal dessen Ideen zum Plankalkül gehört. Bauer selbst bemerkt dazu, *"*[...] *daß diese Begegnung meine wissenschaftliche Entwicklung nachhaltig beeinflußt hat."*

      Allerdings kann durch diesen Vortrag ausschließlich Interesse am Plankalkül, oder vielleicht besser: an der Programmierung von automatischen Rechenanlagen allgemein, geweckt worden sein. Denn wie bereits dargestellt, ist der Vortrag von Konrad Zuse bei den Teilnehmern des Colloquiums auf allgemeines Unverständnis gestoßen. F. L. Bauer selbst gibt zu, daß er damals große inhaltliche Verständnisschwierigkeiten hatte, die noch zusätzlich vergrößert wurden, da Professor Britzelmayr sich vor allem auf inhaltliche Probleme des angeführten Beispielprogramms zur 'Syntaxanalyse von aussagenlogischen Ausdrücken' konzentrierte. Immerhin entstand bei Bauer damals das Interesse an zusätzlichen schriftlichen Ausführungen zum Plankalkül, deren Herausgabe aber von Konrad Zuse verweigert wurden.

      In späteren Jahren muß der fachliche Kontakt zu Konrad Zuse recht eng gewesen sein, auch wenn die fachlichen Kontroversen zwischen dem Praktiker Konrad Zuse und dem theoretischen Informatiker F. L. Bauer wohl nie richtig beseitigt werden konnten: *"*[Es kam] *zu interessanten Diskussionen mit Vertretern der theoretischen Informatik. So habe ich mit Professor Bauer manchen Disput geführt, weil mir seine Auffassungen, wie die der meisten Vertreter der Informatik, in mancher Hinsicht zu theoretisch erschienen."* Diese immerwährende Diskrepanz zwischen dem 'Praktiker'

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Konrad Zuse und den 'Theoretikern' der Informatik war vermutlich auch die Ursache dafür, daß Konrad Zuse gerade in Bezug auf den Plankalkül nie die von ihm gewünschte fachliche Anerkennung seitens der Informatik erhalten hat.

Als Mitte der 50er Jahre die Entwicklung von ALGOL begann, konnte Zuse "[...] *gelegentlich Gespräche mit einigen Schöpfern des ALGOL, wie Rutishauser und Bauer, führen. Meist redeten wir aneinander vorbei. Der Grundgedanke des Plankalküls, eine Programmiersprache systematisch von ihren logischen Wurzeln aus aufzubauen, erschien übertrieben oder wurde als Ballast empfunden.*"

Ob diese Kontakte allerdings bereits Anfang der 70er Jahre, also zum Entstehungszeitraum des zu betrachtenden Artikels bestanden haben, läßt sich aus dem vorliegenden Material nicht entnehmen. Die Frage nach den Kontakten zwischen F. L. Bauer und Konrad Zuse ist vor allem für die Einschätzung des Artikels relevant. Wie noch gezeigt werden wird, ist für eine Interpretation der Analyse und Bewertung des Plankalküls durch F. L. Bauer und H. Wössner ausschlaggebend, ob der vorliegende Artikel in Absprache oder gar Zusammenarbeit mit Konrad Zuse entstanden ist, oder nicht.

Seit Ende der 50er Jahre war F. L. Bauer maßgeblich von europäischer Seite an der Entwicklung sämtlicher ALGOL-Versionen beteiligt. Zum Entstehungszeitpunkt des betrachteten Artikels war er Professor am Mathematischen Institut der TU München, H. Wössner war bei ihm als Assistent angestellt.

2. Verwendete Primärquellen

Die Frage nach der Quellenlage ist wichtig für die Einschätzung der vorgenommenen Bewertung des Plankalküls durch die Autoren. Leider ist sie nicht eindeutig zu beantworten. Es gibt aber eine Reihe von Anzeichen, die zu dem Schluß führen, daß der Plankalkül nicht im Original vorgelegen haben kann.

Im Literaturverzeichnis werden zwar die unvollendete Dissertation von Konrad Zuse und die Originalfassung des Plankalküls als unveröffentlichte Manuskripte angeführt, in der Arbeit selbst wird aber nur aus den bereits vorgestellten Veröffentlichungen zum Plankalkül von 1948, 1959 und 1968 und zusätzlich aus der Zuse-Biographie von 1970 zitiert. Ebenso sind die zur Verdeutlichung der Beschreibung des Plankalküls herangezogenen Programmbeispiele ausschließlich den Veröffentlichungen von 1948 (Syntaxanalyse für aussagenlogische Ausdrücke) und 1959 (Beispiele zur Schachtheorie) entnommen.

Da die Autoren für die Analyse des Plankalküls hauptsächlich das Plankalkül-Beispiel zur 'Syntaxanalyse für aussagenlogische Ausdrücke' verwendet haben, wirken sich die in der von ihnen verwendeten Version von 1948 enthaltenen Fehler besonders negativ auf die abschließende Bewertung des Plankalküls aus. Zur Verdeutlichung wird die von F. L. Bauer und H. Wössner verwendete Programmversion wiedergegeben, wobei *semantische* Änderungen in Bezug auf die in Kapitel 3.1 dargestellte Originalversion von 1945 durch Fettdruck hervorgehoben werden:

Beispiel 11: Programm zur Syntaxanalyse für aussagenlogische Ausdrücke in der korrigierten Fassung von F. L. Bauer/H. Wössner (1972)

Es erscheint unverständlich und auch unwahrscheinlich, daß Konrad Zuse vor allem die in den ersten beiden Zeilen enthaltenen Initialisierungs- und Positionierungsfehler selbst eingebaut hat.

Vielmehr scheint der Artikel *Über den Plankalkül als Mittel zur Formulierung schematisch kombinativer Aufgaben* (1948) vor seiner Veröffentlichung 'korrigiert' worden zu sein, und zwar vermutlich ohne die Mitwirkung von Konrad Zuse. Und zwar nicht nur, wie von D. E. Knuth und L. Trabb Pardo festgestellt, in Bezug auf das Zuweisungszeichen '⇒ ': "*Incidentially, the publishers of* [Zuse1948] *used the sign instead of ⇒ , but Zuse never actually wrote himself.*"

Dahingegen ist wahrscheinlich, daß die Erweiterung der dritten Zeile um die zusätzliche Abfrage $x \neq V0$ von Konrad Zuse vorgenommen wurde. Ebenso wie die Verwendung von $m \times \sigma$ statt $\square \times \sigma$

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

zur Kennzeichnung der variablen Größe von V0.

Im Folgenden wird bei der Einschätzung der durch F. L. Bauer und H. Wössner vorgenommene Bewertung des Plankalküls davon ausgegangen, daß die vollständige Version des Plankalküls bei der Erstellung des Artikels nicht vorgelegen hat.

3. Vorgehensweise bei der Analyse des Plankalküls

Zur Analyse und Bewertung der Plankalkül-Konzepte wird neben einem Beispiel aus der Schachtheorie auch das Programm zur 'Syntaxanalyse für aussagenlogische Ausdrücke' in ALGOL 68-Notation übersetzt. Dieses wird dabei um die aufgezeigten Fehler der verwendeten Version von 1948 korrigiert und erhält damit die folgende Form:

```
proc Sa = ([0: either] bits V0) bool: begin

bits Z0 := V0[0]; bool R := Az(Z0);

int eps := 0; if Kla (Z0) then eps := 1 fi;

for i to upb V0 while R do begin

bits Z1 := V0[i];

R := R ∧ Sq(Z0, Z1);

if Kla(Z1) then eps+ := 1 fi;

if Klz(Z1) then eps- := 1 fi;

R := R ∧ eps ≥ 0;

Z0 := Z1 end;

R ∧ Sz(Z0) ∧ eps = 0 end
```

Beispiel 12: Programm zur Syntaxanalyse für aussagenlogische Ausdrücke in ALGOL 68-Notation

Interessant ist der wertende Kommentar zu dieser Übersetzung: *"(Selbstverständlich enthält ALGOL 68 Möglichkeiten für eine effizientere Formulierung.)"* Da die Bewertung des Plankalküls überwiegend auf der Analyse dieses Programmes beruht, wird durch diese lapidare Anmerkung, gewollt oder nicht gewollt, der Eindruck erweckt, daß der Plankalkül *allgemein ineffizient* ist. Und dies vor allem im Vergleich zu der Programmiersprache ALGOL 68, die zur Entstehungszeit des Artikels als die Sprache mit den modernsten Konzepten betrachtet wurde und an deren Entwicklung F. L. Bauer beteiligt war.

Aus heutiger Sicht hingegen erscheint diese 'Ineffizienz' allerdings nicht verwunderlich. Bei einer 1:1-Übersetzung eines bestehenden Programmes in die Notation einer anderen Programmiersprache, kann aufgrund bestehender konzeptioneller Unterschiede beider Programmiersprachen in der Regel kein effizienter Programmcode erzeugt werden. Als Mangel der ersten Sprache, in diesem Fall also des Plankalküls, kann das aber nicht angesehen werden.

Die anhand des Beispielprogrammes aufgezeigten inhaltlichen Mängel des Plankalkül-Programmes werden hinfällig, wenn man davon ausgeht, daß die fehlerhafte Version von 1948 zugrundegelegt wurde. Dasselbe kann für die Mißverständnisse bei der Erläuterung der Konstruktion und Verwendung von Datenstrukturen im Plankalkül gelten:

- Der Aufbau von komplexen Datenstrukturen wird als das *rekursive* Zusammensetzen der einfachen binären Datenobjekte S0, die als 'primitive Objekte' bezeichnet werden, verstanden. Tatsächlich werden sämtliche Datenstrukturen durch konsequentes ineinander Verschachteln konstruiert.

- Die durch diese Konstruktionsvereinbarung bestehenden Zugriffsmöglichkeiten auf einzelne Komponenten oder Teilbereiche einer Datenstruktur werden nicht gesehen: *"Um das erste Element von* V0 *auszulassen, muß zuvor das Teilfeld, das genau dieses Element nicht enthält, gebildet werden."*. Diese Interpretation ist falsch. Vielmehr ermöglicht der Plankalkül durch eine entsprechende Angabe in der Struktur-Zeile den direkten Zugriff auf Komponenten oder Teilbereiche einer Datenstruktur. Dafür ist allerdings die Angabe des allgemeinen Leerstellenzeichens □

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

erforderlich, welches in der verwendeten Programmversion nicht enthalten ist.

Es ist bedauerlich, daß der Plankalkül aufgrund der schlechten Quellenlage mißverstanden werden mußte.

1. Der Einfluß des Plankalküls auf die Entwicklung von ALGOL

Interessant zu erwähnen ist noch, daß F. L. Bauer und H. Wössner dem Plankalkül einen zumindest indirekten Einfluß auf die Entwicklung von ALGOL 60 zuschreiben: *"Wenigstens drei Teilnehmern an dem Züricher ALGOL-Treffen vom Mai 1958 war* Zuses *Plankalkül einigermaßen bekannt, und die Beeinflussung mag unbewußt stärker gewesen sein, als es vordergründig ersichtlich war."*

Darüber hinaus wird wiederholt darauf hingewiesen, daß vor allem Heinz Rutishauser Bestandteile des Plankalküls aufgegriffen und in die Entwicklung von ALGOL 60 eingebracht hat:

- Die Initialisierung einer Variablen erfolgt ebenso wie die Wertzuweisung durch die Verwendung der Zuweisungsoperation.
- Die Beschränkung auf ausschließlich numerische Objekte. Ebenso wie der Plankalkül kennt ALGOL 60 keine alphanumerischen Datenobjekte.

Zu einem späteren Zeitpunkt, und in Form eines historischen Rückblicks, gibt F. L. Bauer an, daß zusätzlich noch das Prozedurkonzept des Plankalküls und dessen Konzept der Typenspezifikation in ALGOL übernommen wurden. Über den Einfluß des Plankalküls auf die Entwicklung von FORTRAN und ALGOL schreibt er: *"Certainly, FORTRAN was absolutely independent. ALGOL, however, was not (probably therefore it was always better than FORTRAN). The ALGOL procedure concept, the type concept with specifications, the assignment concept was conceptually based on similar ideas in the Plankalkül. Quite frequently, this was done by the European ALGOL members subconsciously."*

1. Bewertung des Plankalküls

Der Plankalkül wird insgesamt als hochentwickelte Programmiersprache eingestuft, in der bereits überraschenderweise weitgehend die Standardelemente der in den 60er Jahren aktuellen Programmiersprachen enthalten sind. So ist z.B. das bereits im Plankalkül enthaltene Konzept zur Strukturierung von Objekten erst unter dem Eindruck der Software-Krise in die etablierten Programmiersprache aufgenommen worden. Daß diese Standardelemente teilweise vom Plankalkül übernommen wurden, wodurch dieser Tatbestand weit weniger beeindruckend wird, bleibt bei dieser Bewertung unberücksichtigt.

Als die wesentliche Nachteile des Plankalküls werden hingegen

- die ausschließliche Verwendung der Struktur- und Artkennzeichnung von Datenobjekten zu Dokumentationszwecken,
- die explizite Rückführung sämtlicher Datenarten auf das einzelne Bit,
- die umständliche Notation des Plankalküls und
- die fehlende Berücksichtigung von Möglichkeiten einer Implementation des Plankalküls

genannt.

Eingangs des Artikels wird die Betrachtung des Plankalküls nicht nur anhand historischer Gesichtspunkte begründet, sondern auch mit der kritischen Auseinandersetzung bestehender Programmiersprachen: *"Nicht nur historisches Interesse, sondern auch die notwendige kritische Reflexion über den heute erreichten Stand mit seinen möglichen Lücken und Schwächen lassen eine Betrachtung von* Zuses *Plankalkül angezeigt erscheinen."*

Dies kann dahingehend interpretiert werden, daß die Untersuchen des Plankalküls auch im Hinblick auf eventuell vorhandene zukunftsweisende Konzepte vorgenommen wurde. Bei der Bewertung des Plankalküls wird allerdings nicht weiter auf mögliche zukunftsweisende Konzepte des Plankalküls eingegangen. Der Plankalkül scheint die dahingehenden

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Erwartungen nicht erfüllt zu haben.

1. ***Joachim Hohmann: Eine Untersuchung des Plankalküls im Vergleich mit algorithmischen Sprachen (1975)***

   Diese Studie zum Plankalkül ist in enger Zusammenarbeit mit Konrad Zuse entstanden, der zeitgleich seine Arbeit über die *Gesichtspunkte zur Beurteilung algorithmischer Sprachen* erstellt hat. Beide Arbeiten wurden durch die Ergebnisse dieser Zusammenarbeit beeinflußt. Zusätzlich bestanden aber auch Kontakte zu F. L. Bauer und H. Wössner, wie der Danksagung zu entnehmen ist.

   Neben der intendierten historischen Würdigung des Plankalküls werden von dessen Analyse vor allem Anregungen für die zukünftige Entwicklung von Programmiersprachen und Programmiertechniken erwartet, um damit einen Beitrag zur Überwindung der Software-Krise leisten zu können

   1. Vorgehensweise bei der Analyse des Plankalküls

      Die Studie basiert auf einer gründlichen Durchsicht und Korrektur des Plankalküls, da dieser *"*[...] *aus historischen Gründen bewußt mit Fehlern veröffentlicht wurde"*. Gleichzeitig wurde eine ‚Übersetzung' der Terminologie und Notation des Plankalküls in den Sprachgebrauch der 70er Jahre vorgenommen, um dadurch eine Vergleichsgrundlage zu den gängigen Programmiersprachen zu schaffen. Insgesamt ist dabei eine umfangreiche Sprachbeschreibung des Plankalküls entstanden.

      Für diese Sprachbeschreibung wird überwiegend ALGOL 68 zur Veranschaulichung herangezogen, da diese Programmiersprache über die *"klarste Konzeption und exakteste Definition"* sämtlicher für die Analyse des Plankalküls herangezogenen Programmiersprachen verfügt und damit *"praktisch den gesamten Sprachumfang"* des Plankalküls bietet.

      Im Anschluß an die Sprachbeschreibung findet ein Vergleich des Plankalküls mit einigen etablierten Programmiersprachen statt. Das sind neben ALGOL 68 noch ALGOL60, FORTRAN, COBOL, PASCAL, PL/I und SIMULA. Dabei werden den in der Sprachbeschreibung herausgearbeiteten Konzepte des Plankalküls jeweils die entsprechenden Konstrukte der einzelnen Programmiersprachen gegenüber gestellt. Leider erfolgt dies nicht in Form einer tabellarischen Gegenüberstellung, wodurch bereits durch die Visualisierung dieses Vergleiches ein Eindruck von den Potentialen des Plankalküls entstehen würde.

      Zusätzlich ist von Nachteil, daß dieser verbale Vergleich der Sprachkonstrukte nicht durchgängig erfolgt. Vielmehr wird für jedes Sprachkonstrukt angegeben, welche etablierte(n) Programmiersprache(n) über ein ähnliches Konzept verfügen. Durch diese Form des Vergleichs ist leider keine direkte Bewertung des Plankalküls im Vergleich mit einer einzelnen Programmiersprache möglich. Die Analyse verliert dadurch an Aussagekraft.

      Im Anhang der Studie sind ausgewählte Beispielprogramme des Plankalküls angeführt, die in die verschiedenen zum Vergleich herangezogenen Programmiersprachen übersetzt wurden. Durch die Erstellung von lauffähigen Programmen sollte der Beweis der logischen Korrektheit der von Konrad Zuse entwickelten Beispielprogramme erbracht werden.

   2. Konzepte zur Lösung der Software-Krise

J. Hohmann sieht einen Ansatz zur Bewältigung der Software-Krise in der konsequenten Verwendung von strukturierten und modularen Programmiertechniken.

Der Plankalkül entspricht der von J. Hohmann vorgenommenen Definition einer *strengen Strukturierung*, da die dafür ausschließlich zulässigen Anweisungstypen im Sprachkonzept des Plankalküls enthalten sind:

- die einfache bedingte Anweisung mit <Bedingung> $\overset{\longrightarrow}{\cdot}$ <Anweisung>

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

- die einfache Wiederholungsanweisung mit W[<Bedingung> $\longrightarrow$ <Anweisung>]
- die zusammengesetzte Anweisungen mit [<Anweisung$_1$>| ...| <Anweisung$_n$>]
- das Verlassen einer Strukturebene
  nur über eine spezielle Sprunganweisung mit FIN$^{<\text{Strukturebene}>}$

Ebenso entspricht der Plankalkül der gegebenen Definition *modularer Programmierung*:

- Unterprogramme dürfen keine Seiteneffekte erzeugen,
- die Änderung von aktuellen Parametern und
- der Zugriff auf globale Variablen sind nicht zulässig.

Da der Plankalkül ausschließlich lokale Variablen kennt, entfällt der letzte Punkt, welcher den Zugriff auf globale Variablen untersagt. Demzufolge können bei der Verarbeitung von Unterprogrammen keine Seiteneffekte auftreten. Eine Änderung von aktuellen Parametern sieht der Plankalkül ebenfalls nicht vor, da während des Programmablaufes ausschließlich mit Zwischenwerten gearbeitet wird. Die Eingabewerte bleiben unverändert, während die Resultatparameter erst beim Verlassen des Programmes gefüllt werden.

Die konsequente Modularisierung von Programmen im Plankalkül zeigt einen Weg zur Bewältigung der Komplexität bei der Erstellung umfangreicher Software-Anwendungen.

1. Bewertung des Plankalküls

Zusammenfassend wird der Plankalkül als durchaus positiv eingeschätzt: *"Der PK* [Plankalkül] *reicht über die Möglichkeiten der am weitesten verbreiteten PS* [Programmiersprachen] *wie ALGOL 60, COBOL und FORTRAN hinaus und ist diesbezüglich nur mit neueren PS (siehe Referenzsprachen* [ALGOL 68, PASCAL, PL/I und SIMULA; Anm. d. Verf.]*) vergleichbar."*

Konrad Zuse selbst sieht das Ergebnis dieser Studie noch positiver: *"Er* [J. Hohmann] *kam zu dem Ergebnis, daß der Plankalkül, verglichen mit den eingeführten Computersprachen, dieselbe und in manchen Bereichen die größere Ausdrucksfähigkeit aufweist.* [...] *Die heute eingeführten algorithmischen Sprachen weisen zwar im Prinzip die Ausdrucksfähigkeit des Plankalküls auf, doch arbeiten sie oft mit umständlicheren Mitteln."*

1. Konrad Zuse: Gesichtspunkte zur Beurteilung algorithmischer Sprachen (1975)

Diese Studie ist gleichzeitig mit der Überarbeitung des Plankalküls durch J. Hohmann, und in enger Zusammenarbeit mit diesem, entstanden. Zielsetzung war es dabei, den Plankalkül dem aktuellen Entwicklungsstand von Programmiersprachen gegenüberzustellen und damit eine Diskussionsgrundlage für die Auseinandersetzung mit *"Vertretern der Datenverarbeitung und Informatik"* über den Plankalkül zu schaffen: Konrad Zuse hielt es für notwendig, den seiner Meinung nach *"oft sehr theoretischen Formulierungen der Informatik eine mehr anwendungsmässige Auffassung gegenüberzustellen"* und damit die *"mitunter verschiedenen Standpunkte der modernen Informatik und des Plankalküls"* gegeneinander abzugrenzen.

Ebenso wie in der Arbeit von J. Hohmann, soll der aktuelle Bezug zwischen den Konzepten des Plankalküls und den Problemen der Software-Krise hergestellt werden. Durch eine detaillierte Gegenüberstellung von allgemeinen Anforderungen an eine Programmiersprache, den jeweiligen Möglichkeiten des Plankalküls, und der entsprechenden Realisierung in den etablierten Programmiersprachen, bemüht sich Konrad Zuse eine Diskussion in der Fachwelt anzuregen und damit positiv zur Bewältigung der Software-Krise beizutragen.

1. Vorgehensweise bei der Analyse des Plankalküls

Die Studie besteht aus einer Zusammenstellung von als relevant erachteten Kriterien, welche die gewünschte Vergleichsgrundlage zwischen dem Plankalkül und den bereits etablierten Programmiersprachen ermöglichen sollen.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Die Analyse erfolgt für jedes, als *Gesichtspunkt* bezeichnete, Kriterium nach einem einheitlichen Schema:

- Allgemeines zum betrachteten Gesichtspunkt
- Behandlung des Gesichtspunktes durch den Plankalkül
- Behandlung des Gesichtspunktes in 'modernen algorithmischen Sprachen', wobei vor allem der Bezug zu ALGOL 68 hergestellt wird
- Schlußfolgerungen für zukünftige Entwicklungen von algorithmischen Sprachen

Die Zusammenstellung der betrachteten Kriterien scheint dabei vornehmlich auf die Charakteristika und Vorzüge des Plankalküls ausgerichtet zu sein, da diese sich weitgehend an Aufbau und Gliederung der Originalversion des Plankalküls orientieren.

Folgende Gesichtspunkte werden im Einzelnen betrachtet:

- Bezeichnungen
- Darstellungsform
- Datenstrukturen und Arten von Objekten
- Datenverarbeitung (Programmstruktur; dynamischer Programmablauf; Unterprogramme; Aufbau von Kennzeichen und Anweisungen; Besonderheiten der Programmgestaltung, wie z.B. Operatoren, Formeln, bedingte Anweisungen; Operatoren des Prädikatenkalküls)
- Referenzbegriff
- Mensch-Sprache-Computer (Kommentare; Maschinenbezogenheit von Programmiersprachen, wie z.B. allgemeine Hinweise auf die zur Verfügung stehenden Mittel zur Ausführung eines Programmes; Ein- und Ausgabehinweise; Speicherplatzorganisation; Dialogfähigkeit und Eingriffsmöglichkeiten; Compilergerechtigkeit)

Ebenso wie J. Hohmann und F. L. Bauer/H. Wössner stellt Konrad Zuse den direkten Vergleich zwischen dem Plankalkül und ALGOL 68 her, indem er eine Gegenüberstellung vornimmt. Dies ist umso interessanter, als Zuse ausdrücklich seine Abneigung gegenüber dieser Programmiersprache betont. Letztendlich verwendet er ALGOL 68 ausschließlich deshalb, weil diese Programmiersprache "[...] *typisch für die Auffassung der heutigen Informatiker ist und dadurch sowohl in positiver als auch in negativer Hinsicht eine Reihe von Anregungen geben kann.*"

1. Bewertung des Plankalküls

Durch die äußerst unglücklich gewählte Form der Strukturierung wirkt der Aufbau der Arbeit sehr monoton. Hinzu kommt das Problem, daß Konrad Zuse gerade die allgemeine Betrachtung der einzelnen Kriterien inhaltlich sehr weit faßt. Damit werden diese Abschnitte zu lang und zu wenig konkret. Die Lesbarkeit der Arbeit leidet dadurch erheblich.

Ebenso verfügt die Zusammenfassung der Arbeit über keinerlei Aussagekraft, da sie nur die Beziehung des Plankalküls zu den betrachteten Gesichtspunkten herstellt. Dabei ist doch gerade das Interessante an dieser Arbeit, daß sie nach einheitlichen Kriterien den direkten Vergleich zwischen dem Plankalkül und ALGOL 68 anstellt, selbst wenn diese Kriterien speziell auf den Plankalkül zugeschnitten erscheinen.

Leider muß sich der Leser die Informationen zu dem vorgenommenen Vergleich des Plankalküls mit ALGOL 68 selbst zusammensuchen, da eine entsprechende Zusammenstellung in der Arbeit fehlt. Aufgrund der bereits beschriebenen schlechten Lesbarkeit der Arbeit ist dies äußerst mühselig und deshalb letztendlich nur äußerst unzureichend zu realisieren.

1. ***Konrad Zuse: Beschreibung des Plankalküls (1977)***

   Der Arbeit ist keinerlei Information über die Intention ihrer Erstellung zu entnehmen. Sie verfügt weder über ein entsprechendes Vorwort, noch wird in der Einleitung darauf Bezug genommen. Die entsprechenden

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Informationen können ausschließlich dem Vorwort der Studie von 1975 entnommen werden.

1. Vorgehensweise bei der Analyse des Plankalküls

   Wie bereits dargestellt, war es Ziel der Studie von 1975, die inhaltliche Diskussion über den Plankalkül innerhalb der Informatik anzuregen. Ein Ergebnis war die Erkenntnis, daß die Voraussetzung für eine solche inhaltliche Diskussion ein einheitlicher Sprachgebrauch ist. Demzufolge war eine gründliche Überarbeitung des Plankalküls erforderlich, sowie und dessen Anpassung an die Terminologie der 70er Jahre.

   Die vorliegende Studie soll "[...] *z. T. einen Extrakt der vorliegenden Arbeit* [Zuse1975] [...] *und z.T. eine Ergänzung in bezug auf den Plankalkül dar*[stellen]." Wobei allerdings der 'Extrakt', d.h. die Zusammenstellung der wesentlichen Konzepte des Plankalküls, dessen Fehlen in der Arbeit von 1975 bereits bemängelt wurde, wiederum zu kurz kommt. Vielmehr wird die *Beschreibung des Plankalküls* ihrem Titel vollkommen gerecht: das Buch stellt vor allem eine Umsetzung der Originalfassung des Plankalküls in die Terminologie der 70er Jahre dar.

   Auch wenn explizit hervorgehoben wird, daß diese Arbeit eine von bisherigen Veröffentlichungen des Plankalküls unabhängige Darstellungsform anstrebt, besteht doch ein sehr enger Bezug zu dessen Originalfassung. Immerhin wird ausdrücklich empfohlen, diese zusätzlich heranzuziehen. Auch ist die Gliederung der Arbeit eng an diejenige der Originalfassung angelehnt. Es wird also Abschnitt für Abschnitt die ursprüngliche Darstellung des Plankalküls in einer überarbeiteten und kommentierten Fassung wiedergegeben. Das gilt nicht nur für die Sprachbeschreibung des Plankalküls, sondern auch für die Beispielprogramme der verschiedenen Programmkategorien, wie z.B. der Schachtheorie.

2. Bewertung des Plankalküls

Die inhaltliche Form dieser Arbeit ist absolut mangelhaft: bereits das 'Vorwort' geht ausschließlich auf inhaltliche Aspekte des Plankalküls ein. Auf eine Einleitung in das Thema wurde vollständig verzichtet, vielmehr wird sofort mit den 'Syntaktischen Regeln des Plankalküls' begonnen. Ebenso abrupt, wie diese Arbeit anfängt, hört sie auch auf: es gibt keinen abschließenden Teil, der die Ergebnisse der Arbeit kurz zusammenfassen und damit die Arbeit abrunden würde.

Insgesamt ist diese Studie deshalb ihrem Anspruch nicht gerecht geworden, als Diskussionsgrundlage für die Aktualität des Plankalküls zu dienen. Auch wird sie nicht zu weiterführenden Erkenntnissen bezüglich der Einsatzmöglichkeiten von Konzepten des Plankalküls im Zusammenhang mit der Lösung der Software-Krise geführt haben. Immerhin wird aber wirklich ein Beitrag zu einem besseren Verständnis der Originalfassung des Plankalküls geleistet.

2. *Bewertung der Studien Konrad Zuses zum Plankalkül*

Konrad Zuse war immer davon überzeugt, *"daß der seit 1946 in der Schublade liegende Plankalkül noch einmal praktische Bedeutung bekommen wird"*, zumal er den Plankalkül auch noch zu Beginn der 70er Jahre für den *"bei weitem ‚radikalste*[n]*' Ansatz für eine Computersprache"* hielt. Seine beiden Studien zum Plankalkül sollten dem Versuch dienen, den Plankalkül doch noch in die aktuelle Diskussion um geeignete Programmiersprachenkonzepte einzubringen. Leider konnte er dieses Ziel damit aber nicht erreichen.

Es wäre vermutlich besser gewesen, wenn er nicht kontinuierlich auf der Originalversion des Plankalküls beharrt hätte, sondern vielmehr selbst den Beweis dafür erbracht hätte, daß der Plankalkül auch nach 30 Jahren noch über Aktualität verfügte.

Dafür wäre aber nicht nur die durch J. Hohmann weitgehend vorgenommene Überarbeitung des Plankalküls in Form einer Korrektur der logischen und notationellen Fehler und Inkonsistenzen notwendig gewesen, bei der letztendlich immer die historische Dimension überwogen hat. Weit wichtiger wäre es gewesen, den Plankalkül in eine implementierbare Form zu bringen. Dies war immer einer der Kritikpunkte bei der

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Auseinandersetzung mit dem Plankalkül. Was nützen gute und unkonventionelle Sprachkonzepte, wenn nicht geklärt ist, ob diese überhaupt in der angedachten Form in einer lauffähigen Version auf einem Rechner implementiert werden können. Dazu hätte der Plankalkül in eine lesbare eindimensionale Notation gebracht werden müssen. Allerdings hätte diese nicht eine so komplizierte Form haben dürfen, wie sie Konrad Zuse bereits ebenfalls angedacht hatte.

Zusätzlich hätte endlich eine eindeutige Sprachdefinition des Plankalküls, z.B. in der Backus-Naur-Form, erarbeitet werden müssen. Nur so hätte der Plankalkül in einer ernstzunehmenden Diskussionsgrundlage vorgelegen. Bei einer entsprechenden Überarbeitung des Plankalküls wäre dann allerdings von seiner ursprünglichen Form kaum etwas erhalten geblieben.

Immerhin scheint es Ende der 70er Jahre seitens der Gesellschaft für Mathematik und Datenverarbeitung (GMD) noch den Versuch gegeben zu haben, den Plankalkül zu überarbeiten. Aber die Deutsche Forschungsgesellschaft (DFG), die bis dahin die Studie von J. Hohmann und die beiden Arbeiten von Konrad Zuse finanziert hatte, war an weiterführenden Untersuchungen zum Plankalkül nicht interessiert. Bei den wenig erkenntnisreichen Ergebnissen gerade der beiden Arbeiten von Konrad Zuse erscheint das rückblickend verständlich.

Zuse selbst hat diese Ablehnung allerdings als Bestandteil einer ‘Verschwörung’ gegen sich und seinen Plankalkül aufgefaßt, die verhindern sollte, daß der Plankalkül zu einer ‘Konkurrenz’ für die etablierten Programmiersprachen, wie FORTRAN, ALGOL und COBOL werden könnte: *"Allerdings waren mir deren Sachverständige* [der DFG; Anm. d. Verf.] *nicht allzusehr gewogen. Das Problem lag darin, daß es wirklich neutrale Sachverständige nicht gab: sie alle waren irgendwie an der Entwicklung, der Einführung oder dem Einsatz von Sprachen wie FORTRAN, ALGOL und COBOL und so weiter beteiligt oder beteiligt gewesen. Auch bei bestem Willen hätten sie wohl kaum den nötigen Abstand von ihrer eigenen Arbeit gewinnen können."*

Es erscheint allerdings wenig wahrscheinlich, daß ausschließlich derartig subjektive Gründe zur Einstellung der finanziellen Unterstützung geführt haben können. Seitens der DFG werden wohl vor allem wirtschaftliche Gründe für die Ablehnung weiterer Forschungsarbeiten über den Plankalkül im Vordergrund gestanden haben. Außerdem muß festgestellt werden, daß gerade Konrad Zuse nicht über den für eine grundlegende Überarbeitung erforderlichen Abstand zur Originalfassung des Plankalküls verfügte. Es erscheint unwahrscheinlich, daß eine weitere Studie zum Plankalkül weiterreichende Ergebnisse erzielt hätte als die bisherigen.

Letztendlich war aber zu diesem Zeitpunkt, Ende der 70er Jahre, das fachliche Interesse am Plankalkül auch schon wieder beendet. In den 80er Jahren fand der Plankalkül dann ausschließlich im Rahmen der Geschichte der Computerentwicklung Beachtung. Erst in jüngster Zeit scheint das Interesse der Informatik wieder zu erwachen, und es ist 1997 ein weiterer Artikel erschienen, der sich erneut inhaltlich mit dem Plankalkül auseinandersetzt.

3. *Donald E. Knuth/Luis Trabb Pardo: The Early Development of Programming Languages (1977)*

Eine vollständig andere Form der Annäherung an den Plankalkül als die bisher vorgestellten Arbeiten wählen Donald E. Knuth und Luis Trabb Pardo in ihrer Arbeit *The Early Development of Programming Languages*. Im Gegensatz zu den deutschen Autoren unterziehen sie den Plankalkül nicht dem direkten Vergleich mit einer etablierten Programmiersprache, bei dem die Programmbeispiele des Plankalküls in diese Programmiersprache übersetzt werden. Vielmehr wird versucht, einen vorab definierten Algorithmus in Plankalkül-Notation zu übersetzen. Die Bewertung des Plankalküls erfolgt dann anhand der im Plankalkül enthaltenen Ausdrucksmöglichkeiten.

In der Untersuchung werden 20 frühe Programmiersprachen vorgestellt und der Versuch unternommen, sie anhand von einheitlichen Kriterien zu bewerten. Innerhalb dieses zeitlichen Rahmens fallen die bis 1957 entwickelten Konzepte für sogenannte höhere Programmiersprachen, d.h. bis zu einem Zeitpunkt, zu dem der Versuch unternommen wurde, mit der Entwicklung von FORTRAN und ALGOL vereinheitlichende und

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

damit allgemeingültige Sprachkonzepte zu entwickeln. Die Arbeit basiert überwiegend auf unveröffentlichten Quellen zu den einzelnen Programmiersprachen.

Für den Plankalkül wird bemerkenswerter Weise auf die 1972 veröffentlichte, deutsche Originalfassung und den in demselben Band enthaltenen *Kommentar zum Plankalkül* zurückgegriffen. Die Anregung für die Aufnahme des Plankalküls in diese Arbeit, wurde vermutlich durch die englischsprachige Veröffentlichung des Artikels von F. L. Bauer und H. Wössner (1972) angeregt. Konrad Zuse war zu diesem Zeitpunkt auch in den USA als Entwickler der ersten programmgesteuerten Rechenanlage anerkannt. Bis zu dieser Veröffentlichung war dort allerdings vollständig unbekannt, daß er auch die erste höhere Programmiersprache entwickelt hat. Zumindest auf diesem Gebiet wähnten sich bis dahin die US-Amerikaner immer noch als Vorreiter der Entwicklung.

1. Vorgehensweise bei der Analyse des Plankalküls

   Um die sehr unterschiedlichen Konzepte der untersuchten Programmiersprachen beschreiben, vergleichen und bewerten zu können, entstand für die Autoren das Problem, eine einheitliche Vergleichsgrundlage herstellen zu müssen. Deshalb wählten sie den Weg, einen bestimmten, vorab festgelegten Algorithmus zur Berechnung einer mathematischen Funktion in jeder der betrachteten Programmiersprachen auszudrücken:

   Die in einer Datenstruktur abgelegten Werte sollen über genau zwei ineinander geschachtelte Schleifenkonstruktionen eingelesen und der jeweilige Funktionswert von

   $$f(t) = \sqrt{|t|} + 5t^3$$

   errechnet werden. Anschließend soll entweder dieser Funktionswert ausgegeben werden, oder eine entsprechende Meldung, sofern der Wert eine definierte Größenbeschränkung überschreitet.

   In der Notation des Plankalküls hat der entsprechende Algorithmus die folgende Form:

   Beispiel 13: Programm zur Bewertung des Plankalküls von D. E. Knuth/L. Trabb Pardo (1977)

   In der ersten Zeile wird die im Programm verwendete Datenart A2 deklariert. Diese besteht aus einer Liste von Paaren, deren Vorderglied eine ganze positive Zahl (Datenart A9) beinhaltet. Das Hinterglied ist durch die Datenart AΔ 1 als reelle Zahl in Binärdarstellung in der Form $A\Delta 1 = (3 \times S0, 7 \times S0, 22 \times S0)$ definiert.

   Die zugehörige Artdeklaration ist nicht Bestandteil des Programmes, sondern wird als global definiert für die entsprechende Programmgruppe angesehen. In den Vordergliedern sind die zu verarbeitenden Eingangswerte, in den Hintergliedern die errechneten Resultatwerte enthalten.

   In Plankalkül-Notation besteht der gesamte Algorithmus aus den beiden Programmen P1 und P2, einschließlich des zugehörigen Randauszuges, wodurch die jeweiligen Eingangs- und Resultatwerte deklariert werden. Dabei definiert P1 die Funktion *f(t)*, während P2 das eigentliche Hauptprogramm darstellt, in dem P1 mit wechselnden Eingabewerte innerhalb der speziellen Schleifenkonstruktion W2 aufgerufen wird.

   Der Plankalkül kennt keine Möglichkeiten der Ein- bzw. Ausgabe von Daten, sodaß die Ausgabe der gewünschten Meldung für den Fall, daß ein errechneter Funktionswert die vorgegebene Größenbeschränkung von 400 überschreitet, nicht erfolgen kann. Statt dessen wird die jeweilige Strukturkomponente durch die Zuweisung $(i + \infty)$ mit dem Wert ‚unendlich' gekennzeichnet.

2. Ein Vergleich des Plankalküls mit FORTRAN I

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Zum Abschluß der Untersuchung werden sämtliche vorgestellten Programmiersprachen in Form einer Tabelle miteinander verglichen. In Bezug auf den Plankalkül ist dieser Vergleich vor allem deshalb interessant, weil FORTRAN I darin enthalten ist, wodurch eine Einschätzung der Konzepte des Plankalküls in Bezug auf eine etablierte Programmiersprache möglich wird. Ein Vergleich des Plankalküls mit den übrigen untersuchten Programmiersprachen ist hingegen nicht sonderlich aussagefähig, da keine praktische Bedeutung erlangt hat.

Bewertungskriterien	**Plankalkül**	**FORTRAN I**
Jahr der Entwicklung	1945	1956
verwendete Arithmetik	integers floating-point numbers scaled numbers	integers floating-point numbers
Implementierung	<keine>	sehr gut
Lesbarkeit	schlecht	sehr gut
Kontollstrukturen	sehr gut	mittel
Datenstrukturen	sehr gut	mittel
Maschinenunabhängigkeit	gut	sehr gut
Einfluß auf die weitere Entwicklung	mittel	sehr gut
zuerst durch diese Programmiersprache eingeführte Konzepte	erste Programmiersprache hierarchischer Datenaufbau	Ein-/Ausgabeformate globale Optimierung

Tabelle 7: Vergleich des Plankalküls mit FORTRAN I

Diese Gegenüberstellung spiegelt deutlich die Einschätzung der Autoren über die unterschiedliche Herangehensweise bei Entwicklung des jeweiligen Sprachkonzeptes wider. Der deutliche Vorteil des Plankalküls liegt in dem ausgefeilten Konzept von Datendarstellung und Kontrollstrukturen.

An der Bewertung von FORTRAN I hingegen wird deutlich, daß bei der Entwicklung der etablierten Programmiersprachen vor allem deren Implementierung auf bestehenden Rechenanlagen im Vordergrund gestanden hat. Vielfältige Darstellungsmöglichkeiten komplexer Datenstrukturen, wie sie im Plankalkül vorgesehen sind, benötigen Speicherkapazitäten, welche aufgrund der eingeschränkten Kapazitäten der frühen Rechenanlagen nicht zur Verfügung gestanden haben. Bei den etablierten frühen Programmiersprachen konnte demzufolge der Schwerpunkt gerade nicht auf einem umfangreichen Konzept der Datenrepräsentation gelegt werden.

3. Bewertung des Plankalküls

Zusammenfassend kommen die beiden Autoren zu dem Schluß, daß im Plankalkül viele der wichtigsten Konzepte von Programmiersprachen realisiert worden sind. Dabei sind sie am stärksten von den vielfältigen Möglichkeiten beeindruckt, im Plankalkül komplexe, hierarchische aufgebaute Datenstrukturen darzustellen. Zumal dieses auch noch auf dem einfachen Konzept basiert, sämtliche Datenstrukturen letztendlich auf das einzelne Bit zurückzuführen: "[...]*we should stress the richness of datastructures provided by Zuse's language (even in its early form* [Zuse1944]*). This is, in fact, one of the greatest strengths of the Plankalkül; none of the other languages we shall discuss had such a perceptive notion of data, yet Zuse's proposal was simple and elegant*"

Den großen Nachteil des Plankalküls sehen sie zu Recht in den gravierenden syntaktischen Mängeln des Plankalküls, so daß es nicht möglich ist, Programme in einem leicht verständlichen Format zu formulieren. Im Gegensatz zu Konrad Zuses persönlicher Meinung, daß die zweidimensionale Notation des Plankalküls der Arbeit mit der Schreibmaschine

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

entgegenkommt, finden sie die gewählte Form der Notation weder leicht zu lesen noch zu schreiben.

Insgesamt bewerten Knuth und Trabb Pardo den Plankalkül als seiner Zeit weit voraus. Ihre große Anerkennung für die Leistung Konrad Zuses resultiert vor allem daraus, daß dieser der Entwicklung des Plankalküls nicht die eingeschränkten Möglichkeiten seiner bereits realisierten Rechengeräte zugrunde gelegt hat. Vielmehr hat sich Konrad Zuse zunächst darüber Gedanken gemacht, welche grundlegenden Konzepte dafür benötigt werden, jede beliebige Rechenvorschrift in einer formalen Notation wiederzugeben, um diese dann letztlich durch eine automatische Rechenmaschine berechnen zu lassen.

Die Entwicklung der meisten anderen untersuchten Programmiersprachen basierte in der Regel auf der umgekehrten Herangehensweise: es wurde vor allem darauf geachtet, was unter den gegebenen Hardware-Bedingungen möglich war zu implementieren, dafür wurde die Entwicklung fundierter Sprachkonzepte zurückgestellt.

1. Zur Einordnung des Plankalküls in bestehende Programmiersprachenkonzepte

Die Erkenntnis über den Einfluß des verwendeten Programmiersprachenkonzeptes auf die Qualität der erstellten Software entstand in der Auseinandersetzung mit der sogenannten *Software-Krise*, die seit Ende der 60er Jahre proklamiert wurde. In der Folge wurde vor allem das *Konzept der funktionalen Programmierung* als Alternative zu den etablierten *von Neumann-Sprachen*, wie z.B. ALGOL, FORTRAN und COBOL, diskutiert.

In den Rezeptionen des Plankalküls wurde wiederholt betont, daß neben dem historischen Interesse an dieser Programmiersprache auch untersucht werden sollte, ob sie Konzepte enthält, die zur Bewältigung der Software-Krise herangezogen und in bereits etablierte Programmiersprachen übernommen werden könnten. In diesem Zusammenhang erscheint es aus heutiger Sicht sinnvoll, daß in den entsprechenden Studien eine Zuordnung des Plankalküls zu einem der bestehenden Programmiersprachenkonzepte vorgenommen wurde.

Eine derartige Einordnung des Plankalküls hat aber nicht stattgefunden. Vielmehr wurde fast ausschließlich ALGOL 68 für den Vergleich und die Bewertung des Plankalküls herangezogen, ohne jemals zu untersuchen, ob diese beiden Programmiersprachen überhaupt auf demselben Sprachkonzept basieren. Nur in diesem Fall scheint aber eine einheitliche Vergleichsgrundlage gegeben. Die sehr konträren Bewertungen des Plankalküls durch die Autoren machen das Fehlen dieser Vergleichsgrundlage deutlich.

1. *Die Software-Krise*

Gegen Ende der 60er Jahre wurde erkannt, daß die Kosten für die Erstellung von Software überproportional zu den Hardware-Kosten angestiegen waren, und es war absehbar, daß sich dieser Trend noch verstärken würde. Zusätzlich veränderte sich die Tätigkeit der Programmierung zunehmend von der Neuerstellung von Programmen weg, hin zur Wartung und Erweiterung bestehender, immer komplexer werdender Software-Systeme. Dabei mußte festgestellt werden, daß die bestehenden Forderungen nach Wartbarkeit, Erweiterbarkeit, Wiederverwendbarkeit und Korrektheit der erstellten Software in der Regel nicht gewährleistet werden konnten.

Die hauptsächliche Ursache dafür wurde in dem undisziplinierten und mangelhaften Programmierstil der Programmierer gesehen. Auf der NATO-Konferenz zum Thema ‘Software-Engineering’ von 1968 wurde der Begriff der *Software-Krise* geprägt und ausdrücklich hervorgehoben, *"*[...] *daß es sich bei der Erstellung von Software nicht um eine geheimnisumwobene Kunst, sondern um eine ingenieurmäßige Disziplin handelt."* Die Tätigkeit der Programmerstellung wurde damit zu einer technischen Entwicklungstätigkeit erklärt, die sich an ingenieurmäßigen Grundsätzen zu orientieren hat und in standardisierter und normierter Form vorgenommen werden muß.

1. Konzept der Strukturierten Programmierung

Zur Bewältigung der Software-Krise wurde zunächst das Konzept der strukturierten Programmierung verfolgt, bei dem vor allem durch die konsequente Verwendung von Programmblöcken einer generelle Vereinheitlichung der Programmstruktur erreicht werden soll. Spätestens Ende der 70er Jahre hat sich dann aber herausgestellt, daß mit dem Konzept der strukturierten Programmierung die entwickelten

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Qualitätsanforderungen an die erstellte Software nicht gewährleistet werden konnten. Eine allgemeine Durchsetzung in der Praxis konnte nicht gewährleistet werden, da die Anwendung des Konzeptes der strukturierten Programmierung ausschließlich im Ermessen eines jeden einzelnen Programmierers lag.

2. Die Diskussion unterschiedlicher Programmiersprachenkonzepte

Mit der Erkenntnis, daß die Sicherung von Qualität bei der Erstellung von Software nicht allein durch Disziplin und Selbstbeschränkung seitens der Programmierer erzielt werden kann, und daß diese letztendlich auch nicht die Verursacher der bestehenden Probleme sind, hat Ende der 70er Jahre eine tiefergehende Ursachenforschung über den bestehenden Programmierstil eingesetzt.

Dabei wurde erkannt, daß die am weitesten verbreiteten Programmiersprachen, d.h. vor allem ALGOL, FORTRAN und COBOL, demselben Sprachkonzept angehören, welches speziell auf dem allgemein verbreiteten Konzept der sogenannten 'von Neumann-Architektur' von Rechenanlagen basiert. Die dadurch verursachten Unzulänglichkeiten der höheren Programmiersprachen implizieren einen ganz bestimmten, die Möglichkeiten der Entwicklung effizienter Programme stark beschränkenden Programmierstil.

Ein sinnvoller Ausweg aus dieser Problematik schien in der vollständigen Abkehr von dem Sprachkonzept dieser 'von Neumann-Sprachen' zu liegen. Die entsprechende Diskussion kam zunächst in den USA auf. Als Alternative wurde z.B. die konsequente Verwendung eines funktionalen Programmierstils diskutiert. Dieser Ansatz konnte sich aber nicht gegen die von Neumann-Sprachen durchsetzen.

Um letztendlich die durch die Software-Krise aufgezeigten Anforderungen an die Bewertung und Verbesserung der Qualität von erstellter Software doch noch zu erfüllen, wurde in jüngster Zeit das Ende der 60er Jahre mit der Programmiersprache SIMULA erstmals entwickelte Konzept der *objektorientierten Programmierung* wieder aufgenommen, welches konsequent auf dem Konzept der strukturierten Programmierung aufbaut. Dabei werden die entsprechenden Konventionen nicht mehr der Disziplin der Programmierer überlassen. Vielmehr wird deren konsequente Umsetzung durch die gegebenen Sprachkonstrukte zwingend vorgeschrieben.

2. *Die Einordnung des Plankalküls in bestehende Programmiersprachenkonzepte in Deutschland*

Anfang der 70er Jahre, d.h. zu dem Zeitpunkt, zu dem die Rezeption des Plankalküls statfand, wurde in Deutschland noch nicht die Veränderung von Programmiersprachenkonzepten als Lösung der Software-Krise diskutiert. Der Lösungsansatz wurde vielmehr in einer veränderten Programmiermethodik gesehen, welche z.B. durch die Konzepte der Strukturierten und Modularisierten Programmierung erreicht werden sollte. Von daher erscheint es nicht verwunderlich, daß Argumente über die Abkehr vom imperativen Sprachkonzept und en von Neumann-Sprachen in der Diskussion um den Plankalkül nicht angeführt werden. Mit der üblichen Zeitverzögerung bei der Übernahme neuer Ansätze aus den USA konnte eine entsprechende Diskussion in Deutschland frühestens Anfang der 80er Jahre aufkommen. Zu diesem Zeitpunkt war die Diskussion um den Plankalkül aber bereits wieder beendet.

Eine Einordnung des Plankalküls in bestehende Programmiersprachenkonzepte wird demzufolge nicht vorgenommen. Vielmehr verwenden die vorgestellten Rezeptionen des Plankalküls ausnahmslos ALGOL 68 als Bezugssprache. Eine entsprechende Diskussion, ob dies der einzig mögliche Weg ist, sich dem Plankalkül zu nähern, wurde in den deutschprachigen Rezeptionen nie geführt. Die formale Notation von ALGOL 68 wurde aber als optimal angesehen, wodurch sie hervorragend geeignet erschien, die im Plankalkül enthaltenen Konzepte zu bewerten.

1. Die ‚ALGOL-Verschwörung'

Bei der Beurteilung der deutschsprachigen Rezeptionen ist zu berücksichtigen, daß diese nicht unabhängig voneinander entstanden sind. Es erscheint wenig verwunderlich, daß F. L. Bauer ganz allgemein die Verbindung zu ALGOL hergestellt hat, da er seit den Anfängen Ende der 50er Jahre an dessen Entwicklung beteiligt gewesen ist. Konrad Zuse hat in seinen Arbeiten über den Plankalkül ebenfalls den Bezug zu ALGOL 68 hergestellt, obwohl er von den Möglichkeiten dieser

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Programmiersprache nicht sonderlich überzeugt war.

Diese Kontroverse zwischen dem Plankalkül und ALGOL, oder besser: zwischen Konrad Zuse und F. L. Bauer, hatte gravierende Auswirkungen auf die Bewertung des Plankalküls. Sowohl bei Konrad Zuse als auch bei F. L. Bauer fällt die Überzeugtheit von der Qualität der eigenen Sprache auf, die demzufolge eine unabhängige und unvoreingenommene Bewertung eines Vergleichs zwischen diesen beiden Programmiersprachen nicht zuläßt. Das wird u.a. in folgenden Aussagen deutlich:

- Konrad Zuse: *"Nach Ansicht des Verfassers stellt* [ALGOL 68; ...] *nicht das Optimum dar"*
- F. L. Bauer/H. Wössner: *"Selbstverständlich enthält ALGOL 68 Möglichkeiten für eine effizientere Formulierung."*

Dabei sah sich allerdings vor allem Konrad Zuse fortwährend veranlaßt den Beweis anzutreten, daß der Plankalkül ‚besser' oder zumindest ‚gleich gut' ist wie ALGOL 68. ie Ursache für diesen Konflikt ist sicherlich darin zu suchen, daß gerade in den Anfängen der Entwicklung von ALGOL Konzepte des Plankalküls darin eingeflossen sind, dieser Einfluß aber kontinuierlich geleugnet wurde.

Rückblickend erscheint dies allerdings unverständlich, da gerade Heinz Rutishauser von der ETH Zürich, der maßgeblich an der Entwicklung von ALGOL 58 beteiligt gewesen ist, in den 50er Jahren engen Kontakt zu Konrad Zuse hatte, und damit auch zu dessen Ideen des Plankalküls. Von 1950 bis 1955 war die von Konrad Zuse während des Krieges gebaute programmgesteuerte Rechenanlage Z4 an das Institut für angewandte Mathematik der ETH Zürich ausgeliehen. Heinz Rutishauser war u.a. mit der Wartung und Programmierung der Z4 beschäftigt. Aufgrund der darauf beruhenden Erkenntnisse veröffentlichte er 1952 eine Abhandlung über die automatische Programmierung von Rechenanlagen. Diese Erkenntnisse sind in die Entwicklung von ALGOL 58 eingeflossen. Da es während dieser Zeit wiederholt zu Diskussionen mit Konrad Zuse über Programmierung und damit auch über dessen Plankalkül gekommen ist, erscheint es unwahrscheinlich, daß die im Plankalkül umgesetzten Ideen zur Programmierung von Konrad Zuse keinen Einfluß auf die von Heinz Rutishauser entwickelten Vorstellungen ausgeübt haben.

Zwar wurden bereits in der Rezeption des Plankalküls durch F. L. Bauer und H. Wössner indirekte Einflüsse des Plankalküls auf die Entwicklung von ALGOL durch die Übernahme einiger Sprachkonzepte eingeräumt. Es scheint aber durchaus möglich, daß gerade Konrad Zuse selbst weitaus mehr Ähnlichkeiten zwischen den beiden Sprachen gesehen hat. Zumindest würde dadurch sein beständiges Beharren auf dem Vergleich zwischen dem Plankalkül und ALGOL erklärlich, als der Versuch diese Einflüsse eindeutig nachzuweisen.

1. Die verschenkte Möglichkeit eines Vergleichs des Plankalküls mit SIMULA

Unter dem Eindruck dieser Kontroverse ist die Studie von J. Hohmann besonders interessant, da dieser sowohl mit Konrad Zuse als auch mit F. L. Bauer Diskussionen über den Plankalkül geführt hat, deren Ergebnisse in die Untersuchung eingeflossen sind. Von daher erscheint es naheliegend, daß auch J. Hohmann den Vergleich zwischen dem Plankalkül und ALGOL 68 übernommen hat.

Immerhin wurden auch gelegentliche Bezüge zu anderen Programmiersprachen, wie z.B. PASCAL, PL/I und SIMULA hergestellt. Es ist bedauerlich, daß dieser Ansatz nicht weiterverfolgt wurde. Ein Vergleich der Konzepte des Plankalküls mit SIMULA, welche dem objekt-orientierten Sprachkonzept zuzuordnen ist, wäre der wiederholten Gegenüberstellung zu ALGOL 68 vorzuziehen gewesen. In Bezug auf eine Bewertung des Plankalküls wären damit vermutlich spannendere Ergebnisse zu erwarten gewesen.

Die Ansätze eines Vergleiches zwischen dem Plankalkül und SIMULA, die J. Hohmann in seine Arbeit einfließen läßt, klingen sehr vielversprechend:

- Die direkte Zugriffsmöglichkeit auf beliebige Komponenten einer komplexen Datenstruktur im Plankalkül weist Ähnlichkeiten zur Selektion von Klassenattributen in SIMULA auf.
- Die Möglichkeit des Plankalküls, Programme zu Gruppen zusammenzufassen, könnte dahingehend erweitert werden, daß mehrere Programmgruppen ebenfalls zusammengefaßt werden können. Eine Erweiterung der Programmbezeichnung wäre ausreichend, um eine derartige Hierarchiebildung einführen zu können. Damit wäre das

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Konzept der Klassenbildung aus SIMULA abgebildet.

1. ***Zur Einordnung des Plankalküls in bestehende Sprachkonzepte durch Konrad Zuse***

   Konrad Zuse selbst hat nie den Versuch unternommen, den Plankalkül außer mit ALGOL 68 auch noch mit anderen Sprachen zu vergleichen. Wenn man als Hintergrund die bereits dargestellte ‚ALGOL-Verschwörung' berücksichtigt, dann scheint Zuses Hauptinteresse darin gelegen zu haben, den Nachweis zu erbringen, daß der Plankalkül ‚besser' ist als ALGOL bzw. welche ALGOL-Konzepte direkt oder indirekt vom Plankalkül abstammen. Nur so läßt sich die Detailbesessenheit erklären, mit denen Konrad Zuse in den verschiedenen Arbeiten, einschließlich der von J. Hohmann, versucht hat, den Plankalkül und ALGOL 68 zu vergleichen.

   Dabei wäre es für Zuses erklärtes Ziel, die Anregung einer Diskussion um die Aktualität des Plankalküls, sicherlich vorteilhafter gewesen, bei den Vergleichsstudien weit weniger ins Detail zu gehen. Statt dessen hätten z.B. die Möglichkeiten untersucht werden können, die eine grundlegende konzeptionelle Überarbeitung des Plankalküls hätte bringen können. Schließlich sollten Ansätze gefunden werden, die zur Überwindung der Software-Krise hätten beitragen können. Die Erkenntnis von J. Hohmann, daß der Plankalkül die seinerzeit populären Anforderungen an Strukturiertheit und Modularität erfüllt, erscheint zumindest rückblickend kein sonderlich zufriedenstellendes Ergebnis.

   1. Das Fehlen einer formalen Sprachbeschreibung des Plankalküls

      Das bis heute bestehende Problem im Umgang mit dem Plankalkül liegt hauptsächlich darin, daß dieser über keine formale Sprachbeschreibung verfügt. Zum Zeitpunkt seiner Erstellung war dies auch noch nicht üblich. Vielmehr orientierte sich Konrad Zuse auch dabei an der Mathematik, in der mit Beispielen und verbalen Erläuterungen gearbeitet wird. Die Notwendigkeit einer formalen Syntaxbeschreibung wurde erst im Zusammenhang mit der Entwicklung von ALGOL 60 erkannt und in der Folge mit der Backus-Naur-Form (BNF) ein entsprechendes Beschreibungsmittel zur Verfügung gestellt.

      Die Erstellung einer solchen eindeutigen Sprachbeschreibung für den Plankalkül wäre eigentlich eine Voraussetzung gewesen, um den angestrebten Vergleich mit ALGOL 68 korrekt durchführen zu können. Die kontinuierlich gewählte Form der verbalen Beschreibung hat einen großen Interpretationsspielraum gelassen, der sicherlich mit zu den unterschiedlichen Interpretationsergebnissen beigetragen hat. Es erscheint daher erstaunlich, daß eine solche formale Sprachbeschreibung nicht realisiert wurde. Immerhin wurde gerade der enge Bezug zu ALGOL gesucht. Der Gedanke an die Erstellung einer formalen Sprachbeschreibung für den Plankalkül hätte also nahegelegen.

   2. Der Einfluß von Konrad Zuse auf die Interpretation des Plankalküls

   Andererseits hätte die Überführung des Plankalküls in eine BNF eine gründliche Änderung im Aussehen des Plankalküls zur Folge gehabt. Die Analyse der Rezeptionen hat aber den Eindruck vermittelt, daß gerade dies von Konrad Zuse nicht gewollt wurde. Zwar hat er immer wieder betont, daß für die praktische Relevanz des Plankalküls dessen ‚compilergerechter Zuschnitt' Voraussetzung wäre. Andererseits fällt das Beharren auf den in der Originalfassung enthaltenen Beispielen auf: gerade J. Hohmann betont in seiner Studie immer wieder, daß im Plankalkül zwar ein durch die aktuellen Programmiersprachen etabliertes Sprachkonstrukt enthalten ist, dieses aber nicht explizit durch ein Beispiel in der Originalfassung hervorgehoben wird.

   Eine derartige Vorsicht bei der Beschreibung und Interpretation der Sprachkonstrukte des Plankalküls ist weder aus der in der Untersuchung von J. Hohmann angestrebten historischen Betrachtungsebene, noch im Hinblick auf dessen Analyse bezüglich alternativer Sprachkonstrukte erklärlich. Wenn man allerdings berücksichtigt, daß diese Arbeit in enger Zusammenarbeit mit Konrad Zuse entstanden ist, dann wird dessen Einfluß unverkennbar. Da sich Zuse selbst in allen seinen Arbeiten zum Plankalkül ebenfalls streng an den Aufbau und Inhalt der Originalfassung orientiert hat, erscheint die Vorgehensweise von J. Hohmann allerdings plausibel.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Das Beharren von Konrad Zuse auf der Originalfassung des Plankalküls erscheint nur dadurch erklärlich, daß er vor allem daran interessiert war, Umfang und Inhalt der Übernahme von Elementen des Plankalküls in die ALGOL-Entwicklung nachzuweisen.

2. *Zur Einordnung des Plankalküls in bestehende Sprachkonzepte in den USA*

Die vorgestellte Arbeit von D. E. Knuth und L. Trabb Pardo ist vermutlich die einzige, die sich im englischsprachigen Raum inhaltlich mit dem Plankalkül auseinander gesetzt hat. Die Zielsetzung dieser Untersuchung liegt ausschließlich in der historischen Auseinandersetzung mit frühen, in der Regel weitgehend unbekannten Programmiersprachen. Demzufolge wird auch keine Beziehung zwischen den betrachteten Programmiersprachen und möglichen aktuellen Problemen, z.B. in Verbindung mit der Software-Krise, hergestellt. Die Frage nach der Aktualität des Plankalküls wurde ausschließlich in Deutschland diskutiert.

Eine Einordnung des Plankalküls in bestehende Sprachkonzepte kann auch nicht indirekt über die zum Vergleich herangezogenen Programmiersprachen vorgenommen werden, da diese ausschließlich nach dem Zeitpunkt ihrer Entstehung ausgewählt wurden. Eine Unterscheidung zwischen einzelnen Sprachkonzepten wurde für diese frühen Spezialsprachen noch nicht vorgenommen. Nur durch den Umstand, daß FORTRAN I in die Untersuchung mit einbezogen wurde, konnte der Vergleich des Plankalküls mit einer etablierten Programmiersprache vorgenommen werden.

Hinzu kommt, daß sich die Untersuchung von D. E. Knuth und L. Trabb Pardo nur auf die jeweiligen Basiskonzepte der Programmiersprachen bezieht. Damit kann eine direkte Zuordnung der einzelnen Programmiersprachen zu bestehende Sprachkonzepten nicht abgeleitet werden.

1. Zusammenfassung

Der Plankalkül wurde 1945 von Konrad Zuse in ausschließlich textueller Form entwickelt und niemals implementiert. Die vollständige Veröffentlichung des Plankalküls erfolgte erst 1972. Zu diesem Zeitpunkt war die Entwicklung von Programmiersprachen schon weit vorangeschritten, so daß eine entsprechende Arbeit aus den Anfängen der Computer-Entwicklung nur noch von historischem Wert erschien. Trotzdem sind in den 70er Jahren eine Reihe von Arbeiten entstanden, die sich inhaltlich mit den Konstrukten des Plankalküls auseinander gesetzt haben. Dabei wurde der Plankalkül bereits etablierten Programmiersprachen gegenübergestellt, um daraus eine Bewertung der Potentiale des Plankalküls abzuleiten.

Zielsetzung der vorliegenden Arbeit war es, anhand der Analyse dieser in den 70er Jahren veröffentlichten Rezeptionen zum Plankalkül, die damit jeweils vorgenommene Einordnung des Plankalküls in bestehende Sprachkonzepte zu untersuchen. Der Grundgedanke ist dabei, daß die Aussage darüber, ob der Plankalkül den imperativen von Neumann-Sprachen, den funktionalen, den logischen oder den objekt-orientierten Programmiersprachen zuzurechnen ist, die Voraussetzung für eine abschließende Bewertung des Plankalküls darstellt.

Bedauerlicherweise konnten die in der vorliegenden Arbeit analysierten Rezeptionen des Plankalküls aus den 70er Jahren in dieser Hinsicht keine Hilfestellung geben. Sie zeichneten sich vielmehr durch eine überwiegend subjektive Analyse und Einschätzung des Plankalküls aus, welche auf den persönlichen Vorlieben des jeweiligen Verfassers beruht haben. Demzufolge verfügen auch die vorgenommenen Bewertungen des Plankalküls über wenig Aussagekraft. Sie spiegeln vielmehr hauptsächlich die persönlichen Vorlieben von Friedrich L. Bauer für ALGOL 68 und Konrad Zuse für den Plankalkül wider.

Eine Ausnahme bildet die Untersuchung von Donald E. Knuth und Luis Trabb Pardo, die an die Analyse des Plankalküls unvoreingenommen herangetreten sind. Allerdings haben sie den Plankalkül ausschließlich aus historischen Gesichtspunkten betrachtet, wodurch der Bezug zu den etablierten Programmiersprachen bedeutungslos war.

Für eine realistische Einschätzung der im Plankalkül enthaltenen Sprachkonzepte und einer abschließende Bewertung der Potentiale des Plankalküls erscheint dessen Einordnung in die bestehenden Programmiersprachenkonzepte notwendig, selbst wenn eine eindeutige Zuordnung nicht möglich sein sollte. In diesem Fall kann zumindest eine Abgrenzung gegenüber den einzelnen Sprachkonzepten erfolgen, indem festgestellt wird, welche signifikanten Konstrukte des jeweiligen Sprachkonzeptes bereits im Plankalkül vorweggenommen wurden.

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Wie in der vorliegenden Arbeit gezeigt wurde, ist der Vergleich des Plankalküls mit der *imperativen Programmiersprache* ALGOL bisher nicht zufriedenstellend gelöst worden. Es wäre wünschenswert, den noch ausstehenden systematischen Vergleich der beiden Programmiersprachen nachzuholen.

Ein Vergleich mit der *funktionalen Programmiersprache* LISP könnte aus folgenden Gründen interessant sein:

- Der geschachtelte Aufbau in Form einer Baumstruktur von Datenstrukturen im Plankalkül erinnert an den Aufbau von Funktionsausdrücken in LISP.
- Der Plankalkül kennt die Rekursion, auch wenn sie in der Originalfassung nicht vorgesehen ist.

Ein Zusammenhang zwischen dem Plankalkül und dem *Konzept der logischen Programmierung* wurde in den untersuchten Rezeptionen nicht hergestellt. Da aber Konrad Zuse selbst den Plankalkül als logische Programmiersprache bezeichnet hat, sollte dieses Sprachkonzept bei dem Versuch einer Einordnung des Plankalkül auf jeden Fall berücksichtigt werden.

Der Bezug zur *objekt-orientierten Programmiersprache* SIMULA wurde bereits dargestellt (vgl. Kapitel 6.2.2). Einen weiteren Hinweis darauf, daß der allgemeine Vergleich des Plankalküls mit dem objekt-orientierten Sprachkonzept interessant sein könnte, gibt Joachim Hohmann. Bei seiner Bewertung des Plankalküls führt er aus, daß in diesem das Konzept der Strukturierten Programmierung durch die Programmstrukturen erzwungen werden. Dies entspricht aber genau dem Ansatz der objekt-orientierten Programmierung.

Der Durchführung eines entsprechenden systematischen Vergleiches des Plankalküls mit ausgewählten Programmiersprachen der einzelnen Sprachkonzepte, sollte die Erstellung einer formalen Sprachbeschreibung für den Plankalkül vorangehen. Dies erscheint notwendig, um die einer jeden verbalen Beschreibung innewohnenden Mehrdeutigkeiten zu eliminieren und damit Fehlinterpretationen zu vermeiden.

1. ## Tabellenverzeichnis

2. ## Beispielverzeichnis

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Beispiel 8: Standardaufruf eines Unterprogrammes *

Beispiel 9: Sonderform des Aufrufes eines Unterprogrammes *

Beispiel 10: Programm PΔ 19.2 zur rekursiven Lösung von n! *

Beispiel 11: Programm zur Syntaxanalyse für aussagenlogische Ausdrücke in der korrigierten Fassung von F. L. Bauer/H. Wössner (1972) *

Beispiel 12: Programm zur Syntaxanalyse für aussagenlogische Ausdrücke in

ALGOL 68-Notation *

Beispiel 13: Programm zur Bewertung des Plankalküls von

D. E. Knuth/L. Trabb Pardo (1977) *

3. Literaturverzeichnis

Ahrens/Fischer1992
Ahrens, Klaus/Fischer, Joachim, Objektorientierte Programmierung; Berlin, München: Verlag Technik 1992

Backus1983
Backus, John, Can Programming be Liberated from the von Neumann Style? A Functional Style and its Algebra of Programs, in: Horowitz, Ellis (Ed.), Programming Languages. A Grand Tour; Berlin, Heidelberg, New York: Springer 1983,
S. 146-173; Reprinted from Communications of the ACM, 21(1978)8, S. 613-641

Bauer1980
Bauer, Friedrich Ludwig, Between Zuse and Rutishauser - The Early Development of Digital Computing in Central Europe, in: Metropolis, Nicholas/Howlett, Jack/Rota, Gian-Carlo (Eds.), A History of Computing in the 20th Century. A Collection of Essays; New York, London: Academic Press 1980, S. 505-524

Bauer1980b
Bauer, Friedrich Ludwig, Laudatio auf Konrad Zuse, in: Schmidt, Reimer (Hrsg.), Computer 1980 - Rückblick und Ausblick. Gedanken aus Anlaß der Verleihung des Aachner und Münchener Preises für Technik und angewandte Naturwissenschaften an Konrad Zuse; Aachen 1980, S. 27-32

Bauer/Wössner1972
Bauer, Friedrich Ludwig/Wössner, H., Zuses 'Plankalkül', ein Vorläufer der Programmiersprachen - gesehen vom Jahr 1972, in: Elektronische Rechenanlagen, 14(1972)3, S. 111-118

Bauer/Wössner1972[engl.]
Bauer, Friedrich Ludwig/Wössner, The ‚Plankalkül' of Konrad Zuse: A Forerunner of Today's Programming Languages, in: Communications ACM, 15(1972)7,
S. 678-685

Dorsch1989
Dorsch, Hadwig, Der 1. Computer. Konrad Zuses Z1 - Berlin 1936. Beginn und Entwicklung einer technischen Revolution; Berlin: Museum für Verkehr und Technik 1989

Giloi1984
Giloi, Wolfgang K., Die Entwicklung der Rechnerarchitektur von der von-Neumann-Maschine bis zu den Rechnern der 'fünften Generation', in: Elektronische Rechenanlagen, 26(1984)2, S. 55-70

Giloi1997
Giloi, Wolfgang K., Konrad Zuse's Plankalkül: The First High-Level, 'non von Neumann' Programming Language,

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

in: Annals of the History of Computing, 19(1997)2, S. 17-24

Hohmann1975
Hohmann, Joachim, Eine Untersuchung des Plankalküls im Vergleich mit algorithmischen Sprachen, GMD-Bericht Nr. 104; St. Augustin: GMD 1975

Knuth/Pardo1977
Knuth, Donald E./Trabb Pardo, Luis, The Early Development of Programming Languages, in: Metropolis, Nicholas/Howlett, Jack/Rota, Gian-Carlo (Eds.), A History of Computing in the 20th Century. A Collection of Essays; New York, London: Academic Press 1980, S. 197-273; Reprinted from Belzer, Jack (Ed.), Encyclopedia of Computer Science and Technology, Vol. 7; New York: Dekker 1977, S. 419-493

Lindner/etal1984
Lindner, R./Wohak, B./Zeltwanger, H., Planen, Entscheiden, Herrschen. Vom Rechnen zur elektronischen Datenverarbeitung; Reinbek bei Hamburg: Rowohlt 1984

Petzold1985
Petzold, Hartmut, Rechnende Maschinen. Eine historische Untersuchung ihrer Herstellung und Anwendung vom Kaiserreich bis zur Bundesrepublik, Technikgeschichte in Einzeldarstellungen Bd. 41; Düsseldorf: VDI-Verlag 1985

Petzold1992
Petzold, Hartmut, Moderne Rechenkünstler - Die Industrialisierung der Rechentechnik in Deutschland; München: Beck, 2. überarb. Aufl. [von Petzold1985] 1992

Rutishauser1952
Rutishauser, Heinz, Automatische Rechenplanfertigung bei programmgesteuerten Rechenmaschinen, Mitteilungen aus dem Institut für angewandte Mathematik Nr. 3, ETH Zürich; Basel: Birkhäuser 1952

Zuse1944
Zuse, Konrad, Ansätze einer Theorie des allgemeinen Rechnens unter besonderer Berücksichtigung des Aussagenkalküls und dessen Anwendung auf Relaisschaltungen (1944), Kurzfassung in: Zuse, Konrad, Der Plankalkül, GMD-Bericht Nr. 63; St. Augustin: GMD 1972, getr. Pag

Zuse1945
Zuse, Konrad, Der Planklalkül (in der Fassung von 1945), in: Zuse, Konrad, Der Plankalkül, GMD-Bericht Nr. 63; St. Augustin: GMD 1972, getr. Pag

Zuse1948
Zuse, Konrad, Über den Plankalkül als Mittel zur Formulierung schematisch kombinativer Aufgaben, in: Archiv der Mathematik, 1(1948/1949)6, S. 441-449

Zuse1959
Zuse, Konrad, Über den Plankalkül, in: Elektronische Rechenanlagen, 1(1959)2,
S. 68-71

Zuse1968
Zuse, Konrad, Gesichtspunkte zur sprachlichen Formulierung in Vielfachzugriffssystemen unter Berücksichtigung des Plankalküls, in: Händler, Wolfgang (Hrsg.), Teilnehmerrechensysteme, NTG-Tagung vom 21.-23. Sept. 1966 in Bad Hersfeld; München, Wien: Oldenbourg 1968, S. 223-230

Zuse1970
Zuse, Konrad, Der Computer - Mein Lebenswerk; München: Verlag Moderne Industrie 1970

Zuse1972
Zuse, Konrad, Kommentar zum Plankalkül, in: Der Plankalkül, GMD-Bericht Nr. 63; St. Augustin: GMD 1972, getr. Pag

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Zuse1972b
Zuse, Konrad, Der Plankalkül, GMD-Bericht Nr. 63; St. Augustin: GMD 1972

Zuse1975
Zuse, Konrad, Gesichtspunkte zur Beurteilung algorithmischer Sprachen, GMD-Bericht Nr. 105; St. Augustin: GMD 1975

Zuse1977
Zuse, Konrad, Beschreibung des Plankalküls, GMD-Bericht Nr. 112; München, Wien: Oldenbourg 1977

Zuse1978
Zuse, Konrad, Vorwort (vom Oktober 1978), in: Hohmann, Joachim, Der Plankalkül im Vergleich mit algorithmischen Sprachen; Darmstadt: stmv, 2. Aufl. 1979,
S. 5

Zuse1980
Zuse, Konrad, Installation of the German Computer Z4 in Zurich in 1950, in: Annals of the History of Computing, 2(1980)3, S. 239-241

Zuse1984
Zuse, Konrad, Der Computer: mein Lebenswerk; Berlin, Heidelberg, New York: Springer, 2. vollst. überarb. Aufl. 1984

4. Anhang A: Lebenslauf von Konrad Zuse (1910 - 1995)

Dieser tabellarische Lebenslauf von Konrad Zuse enthält neben einigen biographischen Daten vor allem die Entstehungsdaten der wichtigsten Entwicklungen von Konrad Zuse.

22.06.1910	geboren in Berlin
1934	Beginn der Entwicklungsarbeiten für programmgesteuerte Rechenmaschinen
1927 - 1935	Studium Bauingenieurwesen, Abt. konstruktiver Ingenieurbau, Technische Hochschule Berlin-Charlottenburg
1935 - 1936	Anstellung als Statiker bei den Henschel-Flugzeugwerken
1936 - 1938	Bau der Z1 (mechanisches Speicherwerk)
1937 - 1945	Studien zum Bau einer elektronischen Rechenmaschine in Röhrenrelaistechnik, zusammen mit Helmut Schreyer
1938 - 1939	Bau der Z2 (Versuchsmodell in elektromagnetischer Relaistechnik)
1939 - 1945	Anstellung als Statiker bei den Henschel-Flugzeugwerken
1939 - 1941	Bau der Z3 (elektromechanische Relaistechnik)
1940 - 1945	Bau eines Versuchsmodells in Röhrenrelaistechnik durch Helmut Schreyer
1941	Gründung der Firma ZUSE Ingenieurbüro und Apparatebau, Berlin
1941 - 1945	Bau der Z4 (elektromechanische Relaistechnik) Bau der Spezialgeräte S1 und S2 zur Flügelvermessung
1944	Bau des Versuchsmodells eines ‘logistischen Rechengerätes’
1945	Flucht aus Berlin nach Hinterstein, Allgäu Entwicklung der universellen Programmiersprache ‘Der Plankalkül’
1949	Gründung der Firma ZUSE KG

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

seit 1949	Entwicklung weiterer programmgesteuerter Rechenanlagen in elektromechanischer, Röhren- und Transistortechnik Modelle: Z9, Z11, Z22, Z23, Z25, Z31
1950-1955	Einsatz der Z4 an der ETH Zürich
1957	Verleihung der Ehrendoktorwürde der Technischen Universität Berlin
1959	Entwicklung des automatischen Zeichentisches Graphomat G64
seit 1964	diverse Ehrungen und Ehrendoktortitel
1966	Ausscheiden aus der ZUSE KG Honorarprofessur für Computertechnologie, Universität Göttingen
18.12.1995	gestorben in Hünfeld

5. Anhang B: Veröffentlichungen von Konrad Zuse

Die vorliegende Zusammenstellung der Veröffentlichungen erhebt keinen Anspruch auf Vollständigkeit. Konrad Zuse hat eine Vielzahl von Arbeiten verfaßt, welche überwiegend in Zeitschriften und Sammelbänden erschienen sind.

Zuse1936
Zuse, Konrad, Verfahren zur selbsttätigen Durchführung von Rechnungen mit Hilfe von Rechenmaschinen, Deutsche Patenschrift Z23624 vom 11. April 1936

Zuse1936[engl.]
Zuse, Konrad, Method for Automatic Execution of Calculations with the Aid of Computers (1936), in: Randell, Brian (Ed.), The Origins of Digital Computers. Texts and Monographs in Computer Science; Berlin, Heidelberg, New York: Springer, 3. Aufl. 1982, S. 159-166

Zuse1944
Zuse, Konrad, Ansätze einer Theorie des allgemeinen Rechnens unter besonderer Berücksichtigung des Aussagenkalküls und dessen Anwendung auf Relaisschaltungen (1944), Kurzfassung in: Zuse, Konrad, Der Plankalkül, GMD-Bericht Nr. 63; St. Augustin: GMD 1972, getr. Pag.

Zuse1945
Zuse, Konrad, Der Plankalkül (in der Fassung von 1945), in: Zuse, Konrad, Der Plankalkül, GMD-Bericht Nr. 63; St. Augustin: GMD 1972, getr. Pag.

Zuse1947
Zuse-Ingenieurbüro Hopferau (Hrsg.), Zuse-Rechengeräte. Informationsblatt des Zuse-Ingenieurbüros, Oktober 1947, in: Beauclair, Wilfried de, Rechnen mit Maschinen. Eine Bildgeschichte der Rechentechnik; Braunschweig: Vieweg 1968,
S. 77-80

Zuse1948
Zuse, Konrad, Über den Plankalkül als Mittel zur Formulierung schematisch kombinativer Aufgaben, in: Archiv der Mathematik, 1(1948/1949)6, S. 441-449

Zuse1949
Zuse, Konrad, Die mathematischen Voraussetzungen für die Entwicklung logistisch kombinativer Rechenmaschinen, in: ZAMM - Zeitschrift für angewandte Mathematik und Mechanik, 29(1949)1/2, S. 36-37

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Zuse1950a
Zuse, Konrad, Deutsche programmgesteuerte Rechenanlagen für wissenschaftliche Berechnungen, in: Vermessungstechnische Rundschau, 12(1950)12, S. 249-253

Zuse1950b
Zuse, Konrad, Programmgesteuerte Rechenmaschinen in Deutschland. Vortrag, gehalten auf der GAMM-Tagung vom 16. - 19. April 1950 in Darmstadt, in: ZAMM - Zeitschrift für angewandte Mathematik und Mechanik, 30(1950)8/9, S. 292-293

Zuse1952
Zuse, Konrad, Der Programmator, in: ZAMM - Zeitschrift für angewandte Mathematik und Mechanik, 32(1952)8/9, S. 246

Zuse1953a
Zuse, Konrad, Erfahrungen mit dem programmgesteuerten Rechengerät Z5, in: Biermann, Ludwig (Hrsg.), Vorträge über Rechenanlagen, gehalten in Göttingen, 19. - 21. März 1953, im Auftrag der Kommission 'Rechenanlagen' der DFG; Göttingen: Max-Planck-Institut für Physik 1953, S. 42-46

Zuse1953b
Zuse, Konrad, Über programmgesteuerte Rechengeräte für industrielle Verwendung, in: Cremer, Hubert (Hrsg.), Probleme der Entwicklung programmgesteuerter Rechengeräte und Integrieranlagen; Aachen: RWTH Aachen 1953, S. 57-75

Zuse1956a
Zuse, Konrad, Gedanken zur Automation und zum Problem der technischen Keimzelle, in: Unternehmensforschung, 1(1956/1957)1 [Nachdruck von 1963],
S. 160-165

Zuse1956b
Zuse, Konrad, Große oder kleine programmgesteuerte Rechengeräte, in: Unternehmensforschung, 1(1956/1957)4 [Nachdruck: 1963], S. 117-120

Zuse1958a
Zuse, Konrad, Die Feldrechenmaschine, in: MTW-Mitteilungen Nr. V/4 1958,
S. 213-220

Zuse1958b
Zuse, Konrad, Die programmgesteuerte elektronische Rechenmaschine Z22 und die programmgesteuerte Relais-Rechenmaschine Z11, in: Kneißl, Max (Hrsg.), Geodätische Streckenmessung, Teil III: Verwendung automatischer Rechenmaschinen in der Geodäsie, Veröffentlichung der Deutschen Geodätischen Kommission bei der Bayerischen Akademie der Wissenschaften, Reihe A: Höhere Geodäsie, Nr. 28/3; München: Deutsche Geodätische Kommission 1958, S. 55-62

Zuse1959a
Zuse, Konrad, Über den Plankalkül, in: Elektronische Rechenanlagen, 1(1959)2,
S. 68-71

Zuse1959b
Zuse, Konrad, Einige Gesichtspunkte der Entwicklung programmgesteuerter Rechenanlagen in den letzten 20 Jahren, in: Allgemeines Statistisches Archiv, 43(1959), S. 334-340

Zuse1959c
Zuse, Konrad, Elektromechanische und elektronische Rechenmaschinen, in: VDI-Bericht Nr. 37; Düsseldorf: VDI-Verlag 1959, S. 37-40

Zuse1962
Zuse, Konrad, Entwicklungslinien einer Rechengeräte-Entwicklung von der Mechanik zur Elektronik, in:

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Hoffmann, Walter (Hrsg.), Digitale Informationswandler; Braunschweig: Vieweg 1962, S. 508-532

Zuse1962[engl.]
Zuse, Konrad, The Outline of a Computer Development from Mechanics to Electronics, in: Randell, Brian (Ed.), The Origins of Digital Computers. Texts and Monographs in Computer Science; Berlin, Heidelberg, New York: Springer, 3. Aufl. 1982, S. 171-186

Zuse1967a
Zuse, Konrad, Über sich selbst reproduzierende Systeme, in: Elektronische Rechenanlagen, 9(1967)2, S. 57-64

Zuse1968a
Zuse, Konrad, Gesichtspunkte zur sprachlichen Formulierung in Vielfachzugriffssystemen unter Berücksichtigung des Plankalküls, in: Händler, Wolfgang (Hrsg.), Teilnehmerrechensysteme, NTG-Tagung vom 21.-23. Sept. 1966 in Bad Hersfeld; München, Wien: Oldenbourg 1968, S. 223-230

Zuse1968b
Zuse, Konrad, Die Entwicklungen der programmgesteuerten Rechenanlagen, in: Busse von Colbe, Walther/Mattessich, Richard (Hrsg.), Der Computer im Dienste der Unternehmensführung; Bielefeld: Bertelsmann 1968, S. 13-35

Zuse1968c
Zuse, Konrad, Wesen und Bedeutung der Elektronik in Gegenwart und Zukunft, in: Der Langfristige Kredit, 19(1968)2, S. 84-87

Zuse1969
Zuse, Konrad, Rechnender Raum; Braunschweig: Vieweg 1969

Zuse1970a
Zuse, Konrad, Der Computer - Mein Lebenswerk; München: Verlag Moderne Industrie 1970

Zuse1970b
Zuse, Konrad, Entwicklungstendenzen der Informationsverarbeitung, in: ADL-Nachrichten. Zeitschrift für Informationsverarbeitung, 15(1970)60, S. 28-33

Zuse1970c
Zuse, Konrad, Wissenschaft und Rechenmaschine, in: Bild der Wissenschaft, 7(1970)3, S. 232-241

Zuse1972a
Zuse, Konrad, Der Plankalkül, GMD-Bericht Nr. 63, St. Augustin: GMD 1972

Zuse1972b
Zuse, Konrad, Computer und Bürokratie, in: DSWR - Datenverarbeitung in Steuer, Wirtschaft und Recht, 1(1972)9, S. 257-262

Zuse1972c
Zuse, Konrad, Zur Problematik der Rechenautomaten, in: Meschkowski, Herbert (Hrsg.), Grundlagen der modernen Mathematik; Darmstadt: Wiss. Buchges. 1972, S. 253-309

Zuse1973
Zuse, Konrad, Die ersten programmgesteuerten Relais-Rechenmaschinen, in: Graef, Martin (Hrsg.), 350 Jahre Rechenmaschinen. Vorträge eines Festkolloquiums; München: Hanser 1973, S. 51-56

Zuse1975
Zuse, Konrad, Gesichtspunkte zur Beurteilung algorithmischer Sprachen, GMD-Bericht Nr. 105; St. Augustin: GMD 1975

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA

Zuse1976
Zuse, Konrad, The Plankalkül, GMD-Bericht Nr. 106; St. Augustin: GMD 1976

Zuse1977
Zuse, Konrad, Beschreibung des Plankalküls, GMD-Bericht Nr. 112; München, Wien: Oldenbourg 1977

Zuse1979
Zuse, Konrad, Vom Bauingenieurstudium zum Computer, in: Schwarz, Karl (Hrsg.), 100 Jahre Technische Universität Berlin 1879-1979. Katalog zur Ausstellung; Berlin: TU Berlin 1979, S. 358-361

Zuse1980a
Zuse, Konrad, Petri-Netze aus der Sicht des Ingenieurs; Braunschweig, Wiesbaden: Vieweg 1980

Zuse1980b
Zuse, Konrad, Some Remarks on the History of Computing in Germany, in: Metropolis, Nicholas/Howlett, Jack/Rota, Gian-Carlo (Eds.), A History of Computing in the 20th Century. A Collection of Essays; New York, London: Academic Press 1980, S. 611-627

Zuse1980c
Zuse, Konrad, Installation of the German Computer Z4 in Zurich in 1950, in: Annals of the History of Computing, 2(1980)3, S. 239-241

Zuse1980d
Zuse, Konrad, Einige Gedanken zur vergangenen und zukünftigen Entwicklung des Computers, in: Schmidt, Reimer (Hrsg.), Computer 1980 - Rückblick und Ausblick. Gedanken aus Anlaß der Verleihung des Aachener und Münchener Preises für Technik und angewandte Naturwissenschaften an Konrad Zuse; Aachen 1980, S. 33-40

Zuse1982a
Zuse, Konrad, Anwendungen von Petri-Netzen; Braunschweig, Wiesbaden: Vieweg 1982

Zuse1983a
Zuse, Konrad, Der Computer und die Evolution menschlichen Denkens, in: Elektronische Rechenanlagen, 25(1983)6, S. 2-4

Zuse1983b
Zuse, Konrad, Computerentwicklung und allgemeine Informationsverarbeitung - grundsätzliche Tendenzen aus persönlicher Sicht, in: Gebhardt, Friedrich (Hrsg.), Skizzen aus den Anfängen der Datenverarbeitung, GMD-Bericht Nr. 143; München, Wien: Oldenbourg 1983, S. 9-22

Zuse1984
Zuse, Konrad, Der Computer: mein Lebenswerk; Berlin, Heidelberg, New York: Springer, 2. vollst. überarb. Aufl. 1984

Konrad Zuse Internet Archive http://zuse.zib.de
License: CC-BY-NC-SA