EIN PRAGMATISCH ORIENTIERTES KONTEXTMODELL FÜR DIE MENSCH-MASCHINE-KOMMUNIKATION

P. Zoller, Universität der Bundeswehr München

Die vorliegende Arbeit behandelt ein Teilgebiet eines insbesondere in den Jahren 1981 bis 1984 betriebenen Forschungsvorhabens (SAM), das teilweise mit Mitteln des Bundesministeriums für Forschung und Technologie gefördert wurde.

Zusammenfassung: Die auf Grund der zahlreichen Programme vorgegebene Bezeichnervielfalt an der Benutzerschnittstelle wird noch verstärkt durch das freie Bezeichnen der vom Benutzer erstellten Programme und Methoden. Dies gilt in gleichem Maße für die Unmengen an Bezeichnern, die in den privaten Dateisystemen zum Ausdruck kommen. Für diesen mit der erdrückenden Bezeichnervielfalt konfrontierten Benutzer macht es nun Sinn, ein möglichst vollständiges Modell zu entwerfen, das durch Berücksichtigung des persönlichen, fachlichen und organisatorischen Umfeldes des Benutzers eine sinnvolle Beschränkung dieser Namensflut erlaubt, die es ihm gestattet, sich mit vertretbarem Aufwand in riesigen Arsenalen von Programmen, Datenobjekten oder anderen informatiktechnischen Größen zurechtzufinden.

1 Motivation

Die konkrete Situation der Benutzerschnittstelle an heutigen Rechensystemen ist besonders durch eine weitgehend unsystematisch gewachsene Flut von Bezeichnern geprägt, die zudem institutionsspezifisch unterschiedliche Terminologien aufweisen. Dabei vergrößert sich die auf Grund zahlreicher, fertig angebotener Programme vorgefundene Bezeichnervielfalt an der Benutzerschnittstelle außerdem durch das freie Bezeichnen der vom Benutzer erstellten Programme und Methoden, das als grundsätzliche Möglichkeit zwar erhalten bleiben, durch die Vorgabe umfangreicher Bezeichnersysteme aber zurückgedrängt werden sollte (dabei wird unter "Benutzer" hauptsächlich derjenige verstanden, für den die Rechensysteme

nur ein Werkzeug zur Bewerkstelligung seiner beruflichen, inhaltlich nicht der Informatik zugehörigen Aufgaben sind [2]). Diese Namensflut wird noch verstärkt durch die in den Datenbanken und Dateien enthaltenen Datenobjekte, wie beispielsweise im Falle des computer integrated manufacturing (CIM) die Stücklisten, Arbeitspläne und Prüfkonstanten, wie auch durch das in [3] vorgeschlagene, vergrößerte Angebot an vorgegebenen Datentypen (z.B. Werkstückmodelle, Fertigungsauftragsformulare etc.).

Für diesen mit der erdrückenden Bezeichnervielfalt konfrontierten Benutzer macht es nun Sinn, ein möglichst vollständiges Modell zu entwerfen, das durch Berücksichtigung des persönlichen, fachlichen und organisatorischen Umfeldes des Benutzers eine sinnvolle Beschränkung dieser Namensflut erlaubt, die es ihm gestattet, sich mit vertretbarem Aufwand in riesigen Arsenalen von Programmen, Datenobjekten oder anderen informatiktechnischen Größen zurechtzufinden, ohne sich vorher in die, durch die Datenverarbeitung geprägte, Begriffswelt (repräsentiert durch das Beispiel der DV-technischen Datei) einarbeiten zu müssen [7]. 'Sinnvoll' bezieht sich auf den Umfang (gedacht ist hierbei an ca. 100 - 1000 Größen) wie auch auf die Bedeutung, die die Größen für den Benutzer besitzen.

Ein weiterer wichtiger Gesichtspunkt dieses Umfeldes ist die leichte Handhabbarkeit. Sämtliche Möglichkeiten der Namensbildung, wie Synonyme, Abkürzungen und Namensbeschränkungen auf gewisse Bereiche, sollten offengehalten werden, um terminologisch korrekte, eindeutige und verständliche Namen schaffen zu können. Sollte innerhalb eines gewissen "Kontextes" eine vollständige Namenskollisionsfreiheit nicht erreicht werden, so sollten die verbleibenden Namenskollisionen nicht dadurch aufgelöst werden, daß versucht wird, möglichst verschiedene (und der eigentlichen Bedeutung nicht adäquate bzw. sehr lange) Namen zu vergeben, sondern durch entsprechende, den 'Kontext der Begriffe' bestimmende Eigenschaften, die die einzelnen, mit kurzen Bezeichnungen behafteten Größen

voneinander unterscheiden (Einbeziehung beispielsweise ihrer Entstehungsgeschichte und des aktuellen Dialogs).

Ein irgendwie zusammengestellter Ausschnitt ist aber noch keine sinnvolle Beschränkung; Ziel ist es somit, eine für typische Aufgabenstellungen sinnvolle Arbeitsmenge zu finden, die für einen gewissen Zeitraum, ohne groβe Änderungen vornehmen zu müssen, dem Benutzer als Schnittstelle am Datensichtgerät angeboten werden kann; es kann hierbei auch von einem "Arbeitskontext" gesprochen werden. Dies stellt dann eine weitgehende Integration der Fach-Terminologie, sei es anwendungsbezogen, fach- oder institutionsspezifisch, in die Benutzerschnittstelle dar, die dadurch die Verständigkeit eines "eingearbeiteten Kollegen" gewinnt.

2 Namensgebung

Bei der Arbeit mit Moduln, Datenobjekten und Datentypen (wie auch bei den noch später zu behandelnden elementaren Kontextbausteinen und Kontexten) macht es Sinn, mit den eigentlichen Werten nicht unmittelbar zu operieren, sondern diese, wie auch die damit zusammenhängenden Komponenten (z.B. formale Parameter, Teilobjekte etc.), mit einem Namen zu versehen, um so eine leichte Handhabung und eindeutige Identifizierung zu ermöglichen. Ein Bezeichner ist meist kürzer und sinnfälliger als die evtl. vorhandene Standardnotation; auβerdem erlaubt er erst, da er einen zusätzlichen Bestandteil einer Gröβe darstellt, eine für die dialogbezogene Arbeit notwendige Flexibilität, die aus der Beibehaltung des Namens und der Veränderung des entsprechenden Wertes herrührt.

Ein Name bezeichnet gemäβ Meyers Enzyklopädischem Lexikon eine einzelne als Individuum oder individuelles Kollektiv gedachte Person oder Sache zum Zwecke der eindeutigen Identifizierung und Benennung. In der Programmierung ist der Name ein zentraler Begriff und geht meist auf eine Definition der folgenden Art zurück:

a) Der Name eines Elementes besteht aus den Großbuchstaben (A-Z) und den Ziffern (0-9), wobei das erste Zeichen ein Buchstabe sein muß.
b) Ein Name besteht aus 1 bis 8 Zeichen
c) Die Namen innerhalb einer Datei/eines Blockes müssen eindeutig sein.

In der menschlichen Kommunikation ist dem aber i.a. nicht so, wie die zahlreichen kritischen Bemerkungen zu diesem Thema auch zeigen. Ein weiteres Problem besteht darin, daß die Namensgebung meist rein syntaktisch oder anwendungsorientiert geprägt ist. Dies ist aber für die an der Benutzerschnittstelle zur Verfügung stehenden Arsenale nicht ausreichend; sie induzieren eine gewaltige Menge an Bezeichnern: die Namen der Moduln und der mit ihnen gegebenen Formalparameter, die Namen der Datentypen und des mit ihnen gegebenen Zubehörs, beispielsweise die Selektoren, sowie die Namen der im System vorhandenen Datenobjekte. Es gibt daher zahlreiche Verbesserungsvorschläge auf dem Gebiet der Namensgebung, wobei einige der interessantesten den Kommandosprachen zu entnehmen sind.

Wie schon Saltzer [6] erkannte ("A name is always interpreted relative to some context") ist in Normungsvorschlägen zur eindeutigen Charakterisierung eines Objektes die Kenntnis von Kontext, Name und Typ erforderlich, wobei Kontext und Typ implizit gegeben sein können. Die in der Datenverarbeitung wie insbesondere in menschlichen Dialogen auftretenden Mehrdeutigkeiten, Namenskollisionen genannt, könnten durch diese Einbeziehung von Kontexten aufgelöst werden. Daneben könnte auch die Handhabung verbessert, kontextspezifische Namen und diverse Abkürzungen (Abkürzungsregeln, Kommandotabellen, Namensvariablen) vom Benutzer verwendet und die Namensräume (Menge aller Bezeichnungen eines Benutzers in einer bestimmten Situation) auf überschaubare Gebilde begrenzt werden.

3 Das Architekturmodell der Kontexte im Überblick

3.1 Die Kontexte

Jedes Element nach dem gesucht werden kann, seien es also Moduln, Datenobjekte oder andere informatiktechnische Größen, wird genau einer Umgebung zugeordnet, elementarer Kontextbaustein (EKB) genannt, dessen "Eigentümer" (meist die entsprechende Normungsgruppe oder auch - im Falle der privaten Komponenten - der Benutzer selbst) für die Betreuung dieses Elementes und seine Einordnung in den EKB zuständig sind. Es kann daher bei den elementaren Kontextbausteinen von sog. "Pflegschaftseinheiten" gesprochen werden. Um Pflegschaftsüberschneidungen zu vermeiden, wird ausgeschlossen, daß ein Element mehreren elementaren Kontextbausteinen 'aufgebürdet' wird; die EKBe sind somit disjunkt.

Einem Benutzer können/sollen i.a. nicht sämtliche in einem Rechensystem vorhandenen EKBe zur Verfügung gestellt werden; es muß eine geeignete Auswahl getroffen werden. Diese so gebildeten Verwendbarkeitskontexte (VK) werden dem Benutzer meist schon vorgeprägt angeboten, so daß dieser sich, ohne großen Aufwand, darauf beziehen kann. Da zahlreiche VKe in einem Rechensystem vorhanden sein können, kann dies dazu führen, daß einem Benutzer mehrere VKe angeboten werden; einer dieser VKe wird von dem Benutzer ausgezeichnet und steht somit implizit mit dem LOGON ('Anmeldung an ein Rechensystem') zur Verfügung, in die anderen VKe kann von dem Benutzer bei Bedarf explizit umgeschaltet werden.

Der Benutzer muß sich nun aus den ihm im VK zur Verfügung gestellten EKBe die entsprechend seiner Aufgabe benötigten Bausteine aussuchen, diese bilden zusammengenommen dann seinen Benutzeraufgabenkontext (BAK). Dank zahlreicher, einfach zu realisierender Implizitäten (z.B. Berücksichtigung der Arbeitsumgebung des Benutzers oder Fortsetzung einer unterbrochenen Aufgabe), Vorgabe von BAK-Schablonen und Vorprägung benutzerspezifischer BAKe kann dieser Vorgang sehr stark abgekürzt werden.

Der BAK ist für eine einfache Handhabung noch zu umfangreich, so daß durch explizite Einschränkungen und implizite Abhängigkeitsbeziehungen versucht werden muß, diesen BAK weiter einzugrenzen: der so entstehende Arbeitskontext (AK) stellt einen aktuellen Ausschnitt des BAKes dar.

Die dialogbezogene Arbeit eines Benutzers beschränkt sich aber nicht nur auf die im Arbeitskontext enthaltenen Elemente, so kann diese beispielsweise bei Nicht-Auffinden eines Elementes auf den BAK ausgedehnt oder sogar mittels expliziter Angabe seitens des Benutzers, auf die außerhalb des BAKes liegenden Größen erweitert werden.

Es kann somit von einem sog. Verwendungsabstand gesprochen werden, durch den gewissermaßen "Schalen" bzgl. der Mensch-Maschine-Schnittstelle gebildet werden (AK≪BAK≪VK≪SK, d.h. AK ist Vorgängerschale von BAK etc.):

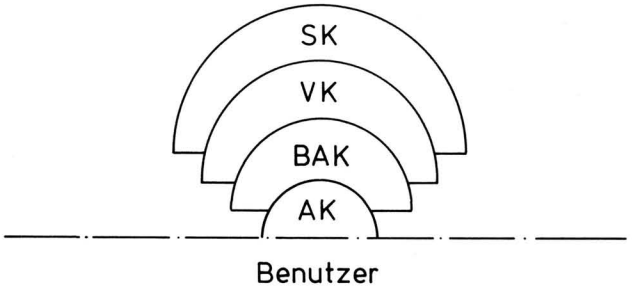

Fig. 1: Der Verwendungsabstand

Abschließend folgt nun die Definition eines auf die dialogbezogene Arbeit zugeschnittenen Arbeitskontextes:

Der Kontext ist eine bzgl. einer Arbeitssituation vollständige und geeignete, wohlstrukturierte Menge von DV-bezogenen Größen, die einem Benutzer bei der Bearbeitung einer bestimmten Aufgabe, in Bezug auf ein Rechensystem, während eines Dialoges an einem Datensichtgerät unmittelbar (d.h. mit minimalem Verwendungsabstand) zur Verfügung stehen.

3.2 Der Benutzeraufgabenkontext

Bisher fehlt es in der Informatik an einem Architekturmodell der Kontexte, das es erlaubt sämtliche möglichen Arbeitsumgebungen des Benutzers einordnen zu können; meist beschränken sich einzelne, unabhängig voneinander geschaffene Konstrukte auf einige wenige Teilbereiche. Durch

- Untersuchung und Weiterentwicklung des in Abschnitt 1 vorgestellten Ansatzes unter Berücksichtigung beispielsweise des Arbeitsablaufes und der Ablagenorganisation eines am Schreibtisch arbeitenden Menschen,
- wie auch mit Hilfe zahlreicher, der Informatikliteratur entnommener Entwicklungen (beispielsweise Methaplan (MEBLOD 1 - 4), UNIX (Suchpfade) oder den seit ca. 1981 vorliegenden Definitionen des environment ([5]) und
- mit Hilfe der Ergebnisse einer Fragebogenaktion (156 Fragebögen, [7])

kristallisierte sich daher ein pragmatisches Schema, bestehend aus den folgenden (sieben) hierarchisch angeordneten Schichten, heraus:

(1) allgemeines Faktenwissen
(2) Fachwissen
(3) Institutionswissen
(4) Projektwissen
(5) Benutzerwissen
(6) Aufgabenwissen
(7) aktuelles Wissen.

Die Anzahl der Ebenen war ursprünglich nicht auf sieben festgelegt, doch zeigte sich durch Studien der verschiedenen Beispiele, daß sich (wenigstens bisher) die gesamte für einen Dialog mit dem Computer benötigte Informationsmenge eines Benutzers in diese 7 Schichten einordnen läßt, so auch beispielsweise die ehemals vorgeschlagenen 13 Ebenen [1]. Die Ergebnisse der Fragebogenaktion bestätigen dies, und zudem liegt die Zahl der Ebenen innerhalb der von der Benutzerforschung vorgegebenen Grenzen [4].

Der BAK eines Benutzers, der an einer bestimmten Aufgabe zur Zeit arbeitet, zerfällt nun in sieben (disjunkte) Teilbereiche, deren Inhalte (= EKBe) zusammengenommen den aktuellen Kontext des Benutzers (bzgl. der fraglichen Aufgabe) repräsentieren. Daraus folgt nun, daß ein Benutzer, entsprechend seinen Aufgaben, mehrere BAKe besitzen kann.

Dieser so entstandene BAK unterstützt nun den - aufgrund des Computers sich in gewissen Grenzen bewegenden - Diskurs, paßt die Fehlermeldungen eines Benutzers an dessen Vorbildung an, ermöglicht einen Rückblick auf die Dialoggeschichte und löst evtl. auftretende Namenskollisionen. Dies geschieht im wesentlichen durch die Vorgabe eines bestimmten **Suchalgorithmus,** der auf dem BAK definiert wird. Einen Überblick bietet abschließend die folgende Abbildung.

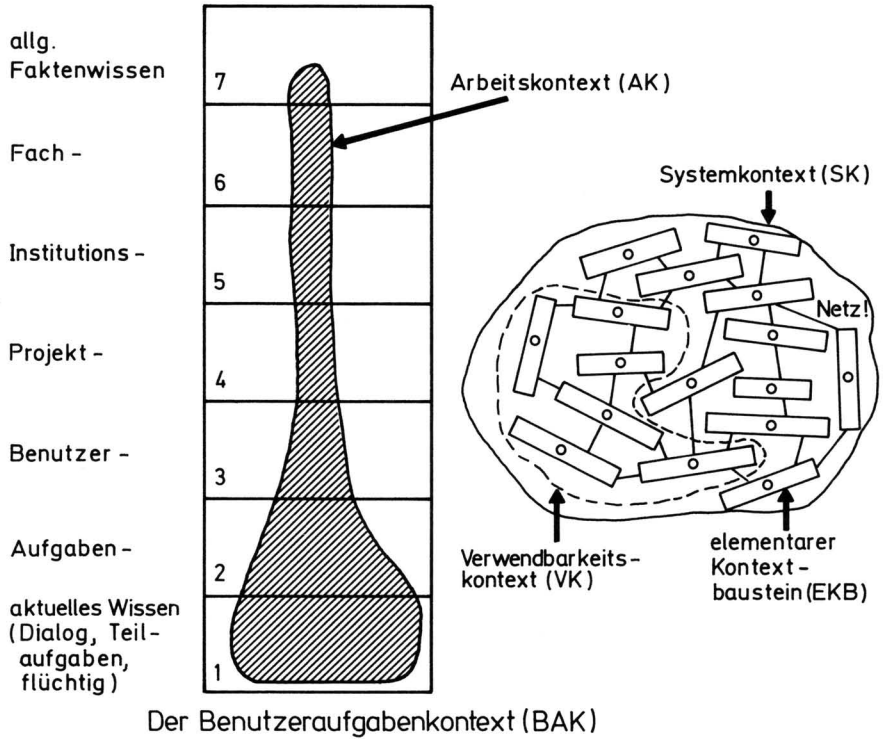

Fig. 2: Das Architekturmodell der Kontexte

4 Verbesserung der Mensch-Maschine-Schnittstelle gezeigt am Projekt MAUS

Um die Nützlichkeit und Durchführbarkeit des obigen Modelles zeigen zu können, werden anhand des Beispiels der "Erstellung einer Einkommensteuererklärung" (und dem hierbei anfallenden Nebenaspekt der Adressabfrage) einige wesentliche Teile des BAK-Konzeptes und diverse Suchstrategien implementiert, so daß dann beispielsweise Fragen der folgenden Art mit möglichst wenig Aufwand (Klärungsdialoge) beantwortet werden können.

Benutzer: Arbeitet Herr Meier noch bei der Firma Comp AG ?
System : Ja, immer noch.
Benutzer: Seine Adresse ?
System : ...
:
:
Benutzer: Wie war nochmals die letzte Adresse ?

Die Bearbeitung des eigentlichen EST-Formulars baut dann auf einem vom System geführten Dialog auf, dem die im BAK enthaltenen Informationen zugrunde liegen, so daß das Gespräch ähnlich einer zwischenmenschlichen Kommunikation geführt werden kann: unveränderliche Daten der letzten EST werden übernommen, im Verlauf des vergangenen Jahres eingetretene Änderungen (z.B. Umzug) werden soweit als möglich implizit berücksichtigt und die damit zusammenhängenden bzw. aus Steuergesetzänderungen abzuleitenden Auswirkungen auf andere Teilbereiche der EST (z.B. Angabe bei Werbungskosten) dem Benutzer angezeigt. Reichen die im BAK enthaltenen Daten für das Ausfüllen einer Komponente nicht mehr aus oder muß aus Sicherheitsgründen auf Richtigkeit nachgefragt werden, so muß ähnlich der zuletzt angegebenen Anfrage auf einen Klärungsdialog zurückgegriffen werden. Die Fertigstellung einer ersten Version des Systems ist für Ende September 1985 geplant (unter dem Namen MAUS: Münchner Adressen- und Steuernsystem).

5 Literaturverzeichnis

[1] Ertingshausen, T.: Die Strukturierung des Gedankenguts eines Rechensystems durch Kontexte, Diplomarbeit, HSBw-ID 06/82, Neubiberg, 1982

[2] Kaufmann, S.; Paul, D.W.; Wiehle, H.R.; Zoller, P.: SAM, TB 1: Ein Ansatz für ein Gesamtkonzept der Rechner-/Anwenderschnittstelle, # 8305, HSBw, München, 1983

[3] Kaufmann, S.; Paul, D.W.; Wiehle, H.R.; Zoller, P.: SAM, TB 2: Eine (sehr) primitive Kern-Schnittstelle: SAM_0, # 8306, HSBw, München, 1983

[4] Miller, G.A.: The magical number seven, plus or minus two: Some limits on our capacity for processing information, Psych. Review 63, 2, 1956, 81 - 96

[5] NI 5.3.2: Diverse Arbeiten des Normungsausschusses NI 5.3.2 in den Jahren 1981 bis 1984

[6] Saltzer, J.H.: Naming and binding of objects, in: Lecture Notes, Advanced course on operating systems, TU, München, 1977, 1-104

[7] Zoller, P.: SAM, TB 6: Die Strukturierung der Benutzerschnittstelle nach inhaltlichen Gesichtspunkten, Vorstudie, # 8401, UniBw, München, 1985

P. Zoller
Fachbereich Informatik
Universität der Bundeswehr München
Werner-Heisenberg-Weg 39

D-8014 Neubiberg