EIN MIKROPROZESSORGESTÜTZTES INTELLIGENTES DIALOG-DIGITALISIER-SYSTEM FÜR DIE GRAPHISCHE ANWENDUNG DURCH NAIVE BENUTZER

F.-J. Kauffels, V. Grossmann, D. Mayer, M. Schuh,
A. Schwarzwald, M. Weller, Bonn

Zusammenfassung: Das System PICOLO, welches hier vorgestellt wird, ist ein graphisches Display-System auf Mikroprozessor-Basis. Aufgaben sind die autonome Erstellung von Zeichnungen einer gewissen maximalen Komplexität und der Einsatz als hochintelligenter Arbeitsplatz für große CAD-Systeme. Das System ist ein Beispiel für eine kostengünstige, Mikroprozessor-unterstützte, Ergonomie-integrale Lösung, die zeigt, daß die Software-Ergonomie keinen negativen Einfluß auf die Gesamtkosten hat, wenn sie von Anfang an in die Planung als wesentliches Ziel einbezogen wird. Außerdem wird eine Kausalität zwischen Software-Ergonomie und objektorientierter Programmierung deutlich.

1 Einführung

Das Klein-CAD-System PICOLO ist als Single-User-Arbeitsplatzsystem konzipiert. Es soll den Arbeitsplatz eines technischen Zeichners, Architekten etc. nachbilden. Die Berücksichtigung software-ergonomischer Aspekte in der Startphase des Entwurfsprozesses des Systems war deshalb zwingend, weil die Benutzer-Zielgruppe eindeutig als DV-naiv zu kennzeichnen ist. Wie wir sehen werden, hat gerade die frühzeitige Einbeziehung dieser Ziele zum gewünschten Erfolg geführt.

Die Dialogschnittstelle soll eine möglichst genaue Nachbildung des natürlichen Arbeitsplatzes eines technischen Zeichners erzielen. Weiterhin ist gewünscht, daß der Benutzer in der Lage ist, mit zunehmender Vertrautheit mit dem System immer intelligentere Transaktionen durchzuführen.

2 Zielsetzung des Systems

Bevor wir auf Einzelheiten der Realisierung eingehen, wollen wir die Zielspezifikation des Systems informell besprechen. Wir untergliedern dabei in Vorgaben, Aspekte der Software-Ergonomie und Aspekte der Software-Strukturierung.

2.1 Vorgaben

Entsprechend der arbeitstechnischen Zielsetzung des Systems spielt der Digitizer eine zentrale Rolle. An Peripheriegeräten stehen weiterhin ein Raster-Graphik-Display mit hoher Auflösung sowie ein alphanumerisches Display mit Tastatur zur Verfügung.

Als Rechner wird ein Z-80 mit Arithmetikprozessor und 1MB Halbleiterspeicher sowie zwei Double-sided-double-density Floppys in einer zweistufigen Speicherhierarchie verwendet. Ein solcher Ausbau des Rechners ist nicht unbedingt notwendig, wie die Entwicklung des Systems gezeigt hat. Wir werden später noch darauf eingehen. Weitere Peripheriegeräte, wie z.B. Plotter, sind anschließbar. Wir wollen uns jedoch auf die genannten beschränken.

Das System soll Zeichnungen bis zu einer gewissen Komplexität erstellen, verwalten und speichern. Das System ist ausgelegt für zweidimensionale Darstellungen, deren Größe i.w. vom verfügbaren Speicher abhängt. Es ist so ausgelegt, daß farbige Darstellungen durch eine entsprechende Nachrüstung der Hardware verfügbar sein werden.

2.2 Aspekte der Software-Ergonomie

Der erste wesentliche Aspekt ist eine einfache modulare Modellbildung des Gesamtsystems. Der Benutzer sollte einerseits in der Lage sein, das System dynamisch während der Benutzung kennenzulernen. Dies muß schrittweise geschehen können, damit er nicht mit Information überladen wird. Um die einzelnen Informationen richtig einzuordnen, sollte es für den Benutzer andererseits rasch möglich sein, das System als Ganzes in grober Körnung zu überblicken; entsprechend seinem Informationsstand kann er dann die ihm offenstehenden Möglichkeiten intelligent wählen.

Der nächste Aspekt ist die Minimierung artfremden Handelns. Die innere Akzeptanz eines DV-Systems durch einen zwangsweise herangeführten DV-naiven Benutzer mittleren Bildungsgrades wird unabhängig von seiner Aufgeschlossenheit immer umgekehrt proportional sein zu der Anzahl der durch ihn auszuführenden artfremden Aktionen, d.h. der Handlungen, die System- und nicht Zielhandlungs-bezogen sind. Da jedoch ein direkter Zusammenhang zwischen innerer Akzeptanz und Produktivität offenkundig ist, ist dieser Aspekt sehr wichtig. Weiterhin sind wir der Überzeugung, daß diese Minimierung artfremden Handelns ein kleiner aber notwendiger Beitrag zur Verringerung der allgemeinen Technik- speziell DV-Feindlichkeit in den betreffenden Gruppen der Gesellschaft ist.

Die Forderung nach einer einfachen, verständlichen Benutzerführung

ist ein offenkundiges Ziel. In diesem Zusammenhang halten wir die Verwendung der Landessprache des Benutzers für ein wesentliches, bei vernünftiger Software-Planung leicht realisierbares, Hilfsmittel.

Der letzte zu integrierende Aspekt ist die Berücksichtigung der Benutzerqualifikation. Es muß möglich sein, die Benutzerführung im System in Stufen so variabel zu halten, daß ein Benutzer, der durch Vorwissen oder Einarbeitung höherqualifiziert ist, nicht alle Schritte der Benutzerführung durchlaufen muß und so auf einer höheren Stufe intelligenter und flexibler mit dem System kommunizieren und arbeiten kann. Die so entstehende Redundanz an Basisinformation muß durch eine geeignete Konzeption kostenirrelevant gehalten werden.

2.3 Aspekte der Software-Strukturierung

Klassische CAD-Programmpakete sind nach den einzelnen Aufgaben des Systems strukturiert: Zeichnungs-Ein- und-Ausgabe, Ändern, Löschen, Transformieren, Drehen, Zooming usf.. Diese einzelnen Module sind dann sekundär evtl. noch einmal nach den möglichen Sorten von Zeichnungen, die im System vorkommen, unterteilt. Dies führt dazu, daß diese Pakete schlecht an neue Aufgaben anpaßbar sind. Wenn einmal die möglichen Zeichnungselemente festgelegt sind, ist das Systempaket starr. Nachträgliche Änderungen sind mit hohen Kosten und Risiken verbunden. Näheres darüber in [1].

Diese Inflexibilität ist ein schweres Hindernis für die Einbeziehung software-ergonomischer Erkenntnisse in das Gesamtsystem.

Im Gegensatz zu diesem Weg soll bei PICOLO ein anderes Strukturierungskonzept verfolgt werden: die Strukturierung nach Art der Zeichnungselemente, jener Grundelemente also, aus denen eine ganze Zeichnung zusammengesetzt ist. Diese Strukturierung orientiert sich nicht primär an den verschiedenen Aufgaben innerhalb des Systems sondern macht die einzelnen Sorten oder Typen von Elementen innerhalb einer Zeichnung zu ihrer primären Grundlage. In der modernen Software-Technologie ist diese Vorgehensweise, die nicht von den Aufgaben, sondern von den Objekten, auf denen diese Aufgaben zu erfüllen sind, geleitet wird, als objektorientierte Programmierung bekannt [2].

In unserem Fall sind die Objekte die sogenannten "Graphischen Typen" also die kleinsten Einheiten innerhalb einer Zeichnung (wie Linie, Kreisbogen, Text, Kurve, Polygon). Dabei soll ein graphischer Typ eine vollständige Spezifikation einer solchen kleinsten Einheit sein, das heißt Definition durch Eingabe, Transformation und Modifikation sowie Darstellung durch Ausgabe werden für ein Element des Typs genau festgelegt.

Man stellt also für den Umgang mit Objekten eines Typs folgendes zur Verfügung: ein typdefinierendes Objekt und Prozeduren für Operationen auf Inkarnationen des Typs. Für jede Inkarnation müssen Parameter gesetzt werden, z.B. die Lage in der gesamten Zeichnung. Die Erstellung/ Modifikation einer ganzen Zeichnung geschieht durch die Anwendung der Prozeduren auf den einzelnen Elementen der Zeichnung.

Dies bedeutet, daß der Anwender sich die Grundelemente, mit denen er arbeiten möchte, in einem großen Freiheitsgrad wählen kann. Aus diesen setzt er dann (Teil-) Zeichnungen zusammen, wobei sich die Prozeduren für den Umgang mit Zeichnungen wie besprochen ergeben. Die Zeichnungen sind also wieder Objekte zusammengesetzter Art mit den entsprechenden Operationen auf ihnen. Daher können sie wieder zum Aufbau von neuen Zeichnungen benutzt werden. Die Schachtelungstiefe dabei stößt nur an technische, nicht jedoch an konzeptionelle Grenzen.

Wir sehen schon hier, daß objektorientierte Programmierung eine wichtige Voraussetzung für Software-Ergonomie ist.

3 Realisierung der Konzepte

3.1 Überblick

Das System arbeitet in der angegebenen Konfiguration. Die für einen Z-80 relativ große Speicherkapazität wurde durch einen speziellen schnellen Halbleiterspeicher erzielt, der ein Floppy- Laufwerk ersetzt. Die Geschwindigkeit des Systems bzgl. der Antwortzeiten konnte gegenüber früheren Versionen um den Faktor 10 gesteigert werden. Auch wuchs die Komplexität der möglichen Zeichnungen. Als Betriebssystem wurde CP/M verwandt, als Programmiersprache PASCAL MT+, das eine Modulkonzeption und Overlay-Strukturen unterstützt. Die für die Durchsetzung der in 2.3 genannten Ziele besser prädestinierte Sprache ADA stand leider nicht zur Verfügung.

Die PICOLO-Software ist primär nach Art der Zeichnungselemente strukturiert, wie oben ausgeführt. Das System ist in drei logisch und physikalisch getrennte Modul-Schichten untergliedert:

Monitor-Ebene
Graphische-Typen-Ebene
System-Ebene

Wir werden sie i.f. im Hinblick auf die Erreichung der angegebenen Ziele besprechen.

3.2 System-Ebene

In der System-Ebene werden alle Typen und Operationen in Form von Modulen zur Verfügung gestellt, die

- Schnittstellen zu peripheren Geräten darstellen, z.B. E/A-Module mit Initialisierungs- und Dialogprozeduren für Bildschirm, Digitizer, Platte usf.
- vom Anwendungsprogrammierer nicht unmittelbar verwendbare Strukturen definieren und realisieren, wie z.B. Suchbaum zum Suchen von Ecken und Punkten in der Zeichnung oder Zeichnungsdatei, in der die bestehende Zeichnung auf Platte gebracht wird.

Die Körper der Prozeduren dieser Ebene sind für den Anwendungsprogrammierer i.w. unsichtbar. Jede Prozedur des restlichen Systems muß sich auf die wohldefinierten Schnittstellenspezifikationen der Prozeduren der System-Ebene berufen, um durch diese Ebene abgedeckte Aktionen durchführen zu können.

3.3 Graphische-Typen-Ebene

PICOLO arbeitet wie angestrebt auf einer Menge von graphischen Typen. Ein graphischer Typ ist definiert als 5-Tupel (N,E,A,T,M) mit

N = Name des Typs
E = Eingabeprozedur
A = Ausgabeprozedur
T = Transformationsprozedur
M = Modifikationsprozedur .

Die Eingabeprozedur legt fest, in welcher Form ein Objekt des Typs einzugeben ist. Sie erfaßt im Dialog mit dem Benutzer alle Daten eines betreffenden Objektes und speichert es anschließend in der Zeichnung auf Platte.

Die Ausgabeprozedur legt die graphische Darstellung eines Objekts des jeweiligen Typs auf den Ausgabegeräten fest.

Die Transformationsprozedur transformiert ein Objekt des jeweiligen Typs im Sinne von drehen, verschieben, kopieren, zoomen als Ganzes.

Die Modifikationsprozedur verändert Teile eines Objektes nach Bedarf oder verändert seine Darstellung (z.B. punktierte statt durchgezogene Linie).

Die Prozeduren dieser Ebene greifen auf die der System-Ebene zurück. Alle graphischen Typen des Systems befinden sich in dieser Ebene. Der Anwendungsprogrammierer kann diese Ebene nach den Wünschen des Benutzers ausgestalten. Der Benutzer selbst kann keine Eingriffe in diese Ebene vornehmen. Dies gilt auch für die anderen beiden Ebenen. Eine spätere Änderung durch den Anwendungsprogrammierer ist selbstverständlich immer möglich.

3.4 Monitor-Ebene

In den Modulen dieser Ebene werden alle Hauptfunktionen (wie Zeichnungen ausgeben, löschen, Objekte definieren, kopieren usf.) zur Verfügung gestellt. Diese Schicht stellt die Schnittstelle zum Benutzer dar.

Für jede Monitor-Prozedur steht dem Benutzer am Digitizer ein Menuefeld zur Verfügung, durch welches die entsprechende Prozedur angesprochen wird.

Die Monitorprozeduren greifen auf die Prozeduren der unteren Ebenen in der besprochenen Weise zurück.

Die Befehlsausführung geht dann z.B. wie folgt vonstatten: durch Aktivieren einer Menuebox auf dem Digitizer wird in der Monitor-Ebene eine Funktion ausgewählt. Diese Funktion überschreitet die ganze Zeichnung und wählt anhand des graphischen Typs für jedes Objekt in der Zeichnung die entsprechende Ausgabeprozedur in der Graphische-Typen-Ebene, die ihrerseits die Objekte mit Zugriffen auf die Graphik-Betriebssystem-Schnittstelle in der System-Ebene ausgeben.

3.5 Individualität, Transparenz und Modularität

Wie wir gesehen haben, ist das System im höchsten Maße modular aufgebaut. Dies hat zum einen den Vorteil der Transparenz für den Benutzer: er gewinnt schnell einen groben Oberblick über das Gesamtsystem; er kann einzelne Module in ihrer Funktionalität betrachten ohne mit Information überlastet zu werden; die für ihn nicht zum Arbeiten relevanten Teile des Systems bleiben verdeckt; letztere machen ca. 80 % der Software aus.

Weiterhin hat die Modularität noch den Vorteil der Flexibilität: wenn z.B. ein neues E/A-Gerät angeschlossen werden soll, so braucht lediglich das entsprechende Modul in der System-Ebene ausgetauscht zu werden. Der Benutzer hat somit die Möglichkeit, ihm evtl. schon vertraute Geräte an das System anschließen zu lassen. Dies ist ein weiterer akzeptanzerhöhender Faktor.

Eine weitere sehr wesentliche Eigenschaft des Systems ist die Individualität: die Typen und Prozeduren der Graphische-Typen-Ebene sind für jeden Anwender individuell gestaltbar. Das bedeutet, daß jeder Benutzer für sich spezifizieren kann, mit welchen Elementen er arbeiten möchte. Diese Elemente kennt er also schon vor der Installation des Systems und kann abgesehen von einer kurzen Einarbeitung sofort mit ihnen arbeiten. Die Erfahrung zeigt, daß die Gesamtzahl der möglichen verschiedenen graphischen Typen nicht so groß ist, wie man zunächst vermuten möchte.

Weiterhin kann man für jeden Benutzer das Menuefeld individuell definieren. Dies erstreckt sich nicht nur auf die wählbaren Funktionen, sondern auch auf die physikalische Lokalisierung und Form des Feldes auf dem Digitizer. Auch dies verringert die Mensch-Rechner-Distanz.

3.6 Minimierung artfremden Handelns

Ein Rechensystem wird nur dann als Hilfe empfunden, wenn der Aufwand für Bedienung und Handhabung incl. Lernphase geringer ist als der Aufwand für die manuelle Lösung des Problems.

In unserem Falle muß der technische Zeichner das Graphiksystem genauso als echtes Hilfsmittel akzeptieren wie Papier und Bleistift. Von der Nützlichkeit wird er erst überzeugt sein, wenn er über längere Zeit positive Erfahrungen mit dem System hat machen können. Es ist also insbesondere die Aufgabe des Systemdesigners, diese Systemerfahrungen positiv zu gestalten. Die Maschine darf den Neuling an ihr nicht überfordern. Die Schnittstelle muß beliebig flexibel ausgelegt werden können, um sich den (wachsenden) Kenntnissen des Benutzers anpassen zu können. Wir werden später noch darauf zurückkommen.

Durch den angeschlossenen Digitizer wird schon allein durch die Handhabung eines Stiftes als Button der Eindruck erzeugt, der Zeichner würde auf einem normalen Blatt arbeiten. Beispielsweise ist es zur Definition einer Linie nur nötig, Anfangs- und Endpunkt auf dem Digitizer anzutippen; es braucht keine umständliche Koordinatenliste eingegeben zu werden. Beim heutigen Stand der Technik ist das Paar Digitizer-"Cursor-Stift" eine gute Approximation für das Paar Papier-Stift. Die Tastatur muß normalerweise nie verwandt werden. Dies wird als sehr angenehm empfunden, weil die Arbeit eines technischen Zeichners sonst auch ohne Tastatur vonstatten geht.

Die Modularität ermöglicht, daß PICOLO für ein und dasselbe graphische Objekt verschiedene Definitionsroutinen verwenden kann. Der Benutzer muß daher nicht umdenken, sondern kann seine bisherige Arbeitsweise beibehalten.

Während der Konstruktion von graphischen Objekten werden dem Benutzer auf dem Graphik-Schirm bereits getätigte Eingaben durch Kreuze angezeigt. Parallel dazu zeigt der alphanumerische Bildschirm die entsprechenden Koordinaten und die Angaben, die noch zur vollständigen Definition benötigt werden. Der Benutzer ist also zu jeder Zeit darüber informiert, welche Eingaben aktuell notwendig sind, wozu sie angefordert werden und wie PICOLO sie interpretiert. Auch hier herrscht also Transparenz.

3.7 Benutzerführung

Eine gute Benutzerführung ist ein fundamentaler software-ergonomischer Aspekt. Wir wollen sie nun für unseren Fall besprechen.

Die Verwendung zweier Monitore (Bild und Text) ermöglicht die konsequente Trennung von Benutzerführung und Informationen über die eigentliche Arbeit. Dies ermöglicht dem Benutzer, entsprechend seiner Qualifikation nur bei Bedarf Notiz von den Systemmitteilungen zu nehmen, und garantiert eine effiziente Arbeit. Zudem wird der Alphabildschirm als "stiller Partner" akzeptiert, der gegebenenfalls weiterhilft, sich aber sonst im Hintergrund hält. Die von uns gewählte Implementierung dieses Features erscheint uns die kostengünstigste zu sein. An die Qualität des CRT werden keine hohen Anforderungen gestellt. Daher ist ein ganzes CRT mit den besprochenen Vorteilen billiger als einige Textzeilen auf dem teuren hochauflösenden Graphik-Bildschirm, so daß dieser voll genutzt werden kann.

Bei der Benutzerführung ist die Verständlichkeit der Dialogsprache für den Benutzer wesentlich. Die Anpassung an eine Landessprache ist bei PICOLO nur mit einem geringen Aufwand verbunden, da alle Textausgaben in leicht austauschbaren Programmodulen zusammengefaßt sind.

Als Beispiel für einen Dialog sei nun kurz eine Liniendefinition ausgeführt. Wir beschreiben die Aktivitäten auf den Bildschirmen. Zunächst bringt der Benutzer sich durch "Ankreuzen" auf dem Menuefeld in den Modus der Liniendefinition. Der Alphabildschirm zeigt dann an:

```
LINIENDEF:  FREI  RECHTWINKLIG  SENKRECHTE  TANGENTE
```

Der Benutzer kreuzt das Gewünschte mit dem Button an. Er entscheidet sich unter dem Angebot der Menuezeile für eine freie Liniendefinition. Der Schirm zeigt dann:

```
LINIENDEF: *FREI  RECHTWINKLIG  SENKRECHTE  TANGENTE
BASIS    :  XY-ACHSEN  BELIEBIG
```

Die zweite Zeile tritt für die Auswahl der Art der Definition der Basis für die Liniendefinition auf. Wir wollen "BELIEBIG" wählen. Wir erhalten:

```
LINIENDEF: *FREI  RECHTWINKLIG  SENKRECHTE  TANGENTE
BASIS    :  XY-ACHSEN *BELIEBIG
BASISDEF :  TASTATUR  DIGITIZER
```

Wie wir jetzt sehen, verlangt das System eine Angabe darüber, ob wir die Basisdefinition über die Tastatur oder den Digitizer vornehmen wollen. Wir entscheiden uns für die letzte Alternative. Durch Betätigung der blauen Taste

des Buttons wird der Anfangs- durch Betätigung der grünen Taste der Endpunkt der Linie bei entsprechender Placierung auf dem Feld eingegeben. Diese Punkte werden dann auf dem Graphik-Schirm angezeigt. Nach Bestätigung erfolgt die Eingabe der Linie in die Zeichnung. Dabei sind noch zwei Punkte erwähnenswert: die Punkteingabe mit dem Digitizer ist nicht immer so genau zu handhaben, wie dies wünschenswert wäre. D.h. es wird nicht die gewünschte Genauigkeit erreicht. Dem hilft das System wie folgt ab: durch ein spezielles Kommando kann erreicht werden, daß das System einen Bezugspunkt für die Liniendefinition in der Zeichnung sucht, der dem eingegebenen Punkt nahekommt. Das System schlägt dann einen oder mehrere dieser Bezugspunkte vor. Man kann dann den gewünschten aussuchen und die Linie verläuft dementsprechend. Analoges gibt es auch z.B. bei der Erstellung von Tangenten. Das System verlangt also keine besonders hohe Genauigkeit vom Benutzer, was dieser positiv empfindet. Weiterhin sind natürlich z.B. die Informationen, welcher Button-Taste welche Funktion zugeordnet ist und ähnliches, vom System erhältlich.

Wie wir gesehen haben, ist die Benutzerführung sehr ausführlich. Für einen Benutzer, der gelernt hat und eine gewisse Erfahrung aufweist, ist sie zu ausführlich. Wir haben vorgesehen, daß der Benutzer einen Teil der Informationen unterdrücken kann. Dadurch sinkt die Anzahl der redundanten Information und die Arbeitsgeschwindigkeit steigt.

Es gibt fünf Stufen der Benutzerführung, von denen die umfangreichste enthält: sämtliche Konstruktionsmöglichkeiten mit Erklärung, Bedeutung der verschiedenen Alternativen eines graphischen Typs, Erklärung der Button-Betätigung, Status des Graphikbildschirmes, mögliche Eingaben, aktuelle Softmenuezeile. Die abstrakteste Stufe enthält nur noch die Softmenuezeile. Der Benutzer hat Wahlfreiheit bezüglich der Stufen.

Die Benutzerführung enthält noch einige weitere Features, auf die wir hier nicht näher eingehen können.

3.8 Anpassung an höherqualifizierte Benutzer

Es stehen in PICOLO für höherqualifizierte Benutzer nicht nur die oben genannten informationsdämmenden Maßnahmen zur Verfügung. Sie alleine reichen nämlich nicht aus für eine wirkliche Steigerung der Arbeitsgeschwindigkeit des erfahrenen Benutzers.

In einem besonderen Modus ("PICOLO-QUICK") können komplexere Eingaberoutinen zur Verfügung gestellt werden, die mehrere einfache Routinen zusammenfassen. Die Unterscheidung in die einzelnen Unterfunktionen wird anhand der möglichen Knopfkombinationen des Buttons des Digitizers getroffen.

Wir wollen dies am Beispiel der Liniendefinition verdeutlichen: Wir seien im QUICK-Modus. Anfangspunkte werden weiterhin mit der blauen Taste definiert. Gleichzeitig wird ein Objekt identifiziert. Der Endpunkt wird bei freien Linien weiterhin mit grün definiert. Soll dagegen rechtwinklig gearbeitet werden, so wird die gelbe Taste für den Endpunkt benutzt. Die Eingabe der Basisrichtung kann dabei über Tastatur (Winkel) oder Digitizer (Vektor vom Anfangspunkt zu einem blaugelb definierten Endpunkt) erfolgen. Wird für den Endpunkt die Kombination blaugrün benutzt, so wird von diesem Punkt je nach dem mit blau identifizierten Element die Tangente an oder die Senkrechte auf dieses Element gezogen.

Mit dieser Definitionsroutine QUICK ist somit also ein sehr schnelles Arbeiten mit dem System möglich, wobei allerdings einige Übung vorausgesetzt wird. Sollten bei einer solchen schnellen Definition Fehler unterlaufen, so können sie in der üblichen Weise leicht korrigiert werden.

Bei diesem Modus geht ein Teil der Übersichtlichkeit und ein Teil des Variantenreichtums verloren, insbesondere, wenn man ihn auf komplexere Objekte (Zeichnungen) ausdehnt, was prinzipiell möglich ist. Diese Routinen müssen deshalb für jeden Anwender nach Ermittlung seines Bedarfes erstellt werden, was jedoch für den Anwendungsprogrammierer nicht erheblich schwierig ist. So könnten z.B. beim Entwurf von Printplatten die IC-Darstellungen auf Knopfdruck abrufbar gemacht werden.

4 Erfahrungen

Das System wurde in etwa 2,5 Mannjahren erstellt. Bereits nach 1,5 Mannjahren konnten die meisten Features benutzt werden. Naive Benutzer haben sich nach einer sehr kurzen Einarbeitung gut zurechtgefunden. Man braucht nur etwa eine Woche für das Manual. Diese Zeit soll noch verkürzt werden, ebenso ist ein effizientes Trainingsprogramm in Arbeit. Nach etwa 10 Minuten mündlicher Einweisung ist man bereits in der Lage, eine Zeichnung zu erstellen. Danach kann der Benutzer das System i.w. alleine dynamisch erschließen.

Durch eine aktive Beteiligung des Benutzers an der Installationsplanung wird die Vertrautheit noch weiter gefördert werden. In der besprochenen Konfiguration ist das System für etwa 50.000 DM erhältlich, wobei der Löwenanteil bei der Hardware liegt. Diese Kosten sind nur ein Bruchteil der Kosten für ein großes CAD-System. Es sinkt also so die Schwelle für die Einführung eines solchen Systems erheblich.

5 Zusammenfassung

Das Dialog-Digitalisiersystem PICOLO wurde vorgestellt. Wir haben gesehen, daß sich auch bei einem kleineren System Software-ergonomische Aspekte und Forderungen integrieren und durchsetzen lassen, wenn die Planung geeignet vorgenommen wird.

Wir haben Forderungen an das System gestellt, die wir für wesentlich halten. Sicher ist die Liste dieser Forderungen nicht vollständig und bedarf einer weiteren Diskussion.

Wir haben gesehen, daß sich für die Durchsetzung dieser Forderungen eine Kausalität zwischen Software-Ergonomie und Objekt-orientierter Programmierung ergibt. Dieser sind wir nachgegangen.

Wir konnten nicht alle Einzelheiten der Realisierung diskutieren. Wesentlich ist aber in diesem Zusammenhang, daß ein Großteil der Ergonomie auf leicht erstellbaren und auswechselbaren Textmodulen beruht, die jederzeit weiter verbessert werden können und keine besonderen Kosten verursachen.

Wir haben die Partizipation bei der Installationsplanung als nützliches Instrument zur Akzeptanzerhöhung erkannt und erfolgreich eingesetzt.

Es konnten hier nur die Software-ergonomischen Aspekte des Systems bezüglich der Abgrenzung zu anderen CAD-Systemen besprochen werden.

6 Literaturverzeichnis

[1] Weller, M.: Standard oder Spezial-CAD-Software? Markt & Technik. Wochenzeitung für Elektronik. 39 (1982)

[2] Wirth, N.: Systematisches Programmieren. Stuttgart: Teubner 1975

Dr.rer.nat. Franz-Joachim Kauffels: Institut für Informatik der Universität Bonn, Abt. II, Wegelerstr. 6, 5300 Bonn 1

V. Grossmann, D. Mayer, M. Schuh, A. Schwarzwald, M. Weller: LOGOTEC Hardware GmbH, Niedenstr. 58-60, 4010 Hilden

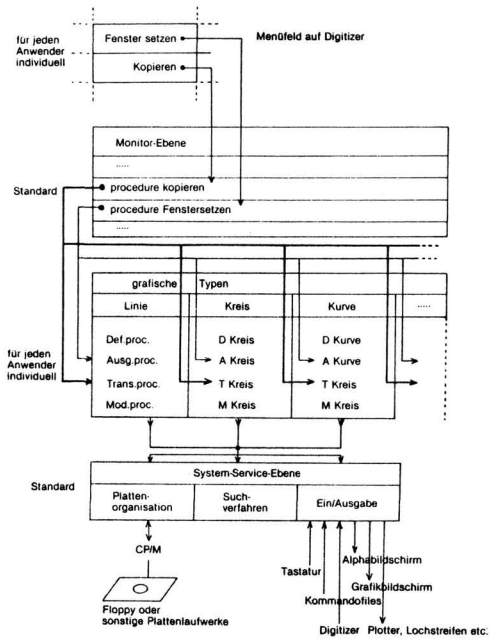

Abb. 1

Das PICOLO Software-Paket:

Standardelemente und individuelle Anwenderprogramme bilden das Gesamtpaket. Deutlich sind die drei Ebenen zu erkennen.

Abb. 2

Die Monitorebene mit der zentralen Monitorprozedur und deren Arbeitsweise.