Spezifikation von Benutzer-Oberflächen durch Bildschirm-Sequenzen

K. Pommerening
Fachbereich Mathematik
der Johannes-Gutenberg-Universität
Saarstraße 21
6500 Mainz

Juli 1987

****

Inhalt.

Einleitung

Erfahrungen mit einer Reihe von Software-Projekten zeigen, daß Entwurf und Implementation einer sauberen Benutzer-Oberfläche immer noch erhebliche Probleme mit sich bringen, obwohl sie scheinbar trivial sind. Diese Schwierigkeiten beruhen auf Mängeln in der Phase der Anforderungsdefinition: Es fehlt immer noch an exakten, systematischen, aber bequem zu handhabenden Methoden zur Spezifikation von Benutzer-Oberflächen, die den Methoden zur Spezifikation von Daten und Funktionen (wie SADT und Funktionshandbuch) gleichwertig sind.

Erste Erfahrungen mit der hier vorgestellten Methode sind sehr positiv: Man gelangt fast wie von selbst zu einem einwandfreien Entwurf. Die Methode ist anwendbar für Programme, die mit einer Bildschirm-orientierten Benutzer-Interaktion auf einem Bildschirm ablaufen, dessen Positionen aus dem Programm heraus beliebig ansteuerbar sind. Um nicht zu allgemein zu bleiben, habe ich die Darstellung jedoch auf Textbildschirme mit grafischer Ausgabe-Möglichkeit eingeschränkt, so daß grafische Benutzer-Oberflächen (Mäuse, Icons, ...), Grafik-Systeme (CAD, ...) und Bildschirmspiele nicht erfaßt werden. Die Methode läßt sich jedoch leicht entsprechend erweitern.

Werden die Ergebnisse der Methode in den folgenden Entwicklungsschritten Entwurf, Implementation und Test konsequent verwendet, so ist garantiert, daß der Benutzer das fertige Programm nicht durch Bedienungsfehler in unerwünschte oder undefinierte Situationen bringen kann (außer durch physische Gewalt oder vorsätzliche Mißachtung von Warnhinweisen).

Die Methode kann halbformal (mit Bleistift und Papier) angewendet werden; das Design passender Formulare oder der Einbau in ein Programm-Entwicklungssystem dürfte aber auch keine besonderen Schwierigkeiten verursachen.

1 Prinzipien.

Die Benutzer-Oberfläche ist neben Daten (oder zu verarbeitenden Informationen) und Funktionen (oder verarbeitenden Aktivitäten) der dritte Teilbereich, der in der Anforderungsdefinition exakt zu spezifizieren ist. Es müssen Konsistenzprüfungen zwischen den drei Bereichen durchführbar sein: Stimmen Bezeichnungen überein? Werden Daten und Funktionen konsistent behandelt? ....

Die Anforderungen an die Benutzer-Oberfläche werden ausgehend davon spezifiziert, wie sich das Programm dem Benutzer darstellt: als eine Folge von Bildschirmen, die verschiedene Informationen, auch gleichzeitig, enthalten können:

- Mitteilungen und Systemmeldungen,
- Aufforderungen und Möglichkeiten zur Kommando-Eingabe,

- Eingabefelder zur Dateneingabe,
- Datenausgabe in Form von Text oder Grafik.

Der Benutzer gibt dann ein Kommando ein, das eine definierte Aktion auslöst; auf bestimmte Kommandos ändert sich außerdem der Bildschirm in definierter Weise. Dieser Teil der Spezifikation erfaßt also nicht nur die statische Bildschirm-Struktur, sondern auch den dynamischen Programmablauf und ist somit auch Grundlage für den Entwurf der Steuerung des Programms. Dies leistet die Spezifikation von Daten und Funktionen nicht.

Dieser Ansatz zur Spezifikation der Benutzer-Oberfläche leistet aber noch mehr: Er führt bequem zu einem brauchbaren Benutzer-Handbuch. Er ist für den Auftraggeber des Projekts nachvollziehbar, so daß Abstimmung und erster Akzeptanztest wesentlich erleichtert werden – mit formalisierten Spezifikationsmethoden auf hohem Abstraktionsniveau, die für den Auftraggeber oft unverständlich bleiben, ist das nicht zu erreichen.

Nach [Balzert, S. 99 ff.] sollen Anforderungsdefinitionen „die Benutzermaschine beschreiben, d. h., die gewünschten Systemfunktionen und das erwartete Systemverhalten.“ Sie sollen

- vollständig,
- konsistent,
- testbar,
- verständlich,
- realisierbar,
- robust gegen Änderungen

sein und

- die Entwurfsfreiheit nicht einschränken.

Die Erfüllung dieser Kriterien wird durch die Methode verbessert, bis auf die letzten beiden. Die Robustheit gegen Änderungen ist natürlich etwas geringer, da mehr Konsistenzprüfungen auszuführen sind. Beim letzten Kriterium, der Entwurfsfreiheit, ist die engere Einschränkung eher positiv zu bewerten, im Sinne einer Entwurfserleichterung.

Die Beschreibung der Benutzer-Oberfläche umfaßt drei Hierarchie-Stufen:

- Bildschirme,
- Fenster,
- (Ein-/Ausgabe-) Felder.

Die Fenster bilden also nach den Bildschirmen nur die zweite Stufe in der Hierarchie. Gerade die Einengung des Blicks auf das zu verwendende Fenstersystem hat oft zu einem unerfreulichen Bildschirmaufbau und einem unnötig komplizierten Entwurf geführt.

2 Definitionen und Erläuterungen

Die Benutzer-Oberfläche (im Sinne der Anforderungsdefinition) ist eine Menge von Bildschirmen.

Ein Bildschirm besteht aus einer Menge von Fenstern und einer Menge von Kommandos. Er besitzt zusätzlich die Attribute

- Modus (z. B. 40 oder 80 Zeichen),
- Zahl der Farben,
- Farbpalette.

Erläuterungen: Ein Bildschirm ist eine Abstraktion des physikalischen, zu einem bestimmten Zeitpunkt für den Benutzer sichtbaren Bildschirms. Die Fenster überlappen sich nicht. Bei der Umsetzung in ein konkretes Programm können auf dem physikalischen Bildschirm auch verdeckte Fenster teilweise sichtbar sein; diese gelten als nicht aktuell und gehören nicht zum momentanen Bildschirm, sondern zum undefinierten Untergrund. Fenster sind rechteckig. Daher wird definiert:

Ein Fenster besteht aus den (Text-) Koordinaten des linken oberen und des rechten unteren Eckpunktes, dem Typ und weiteren typabhängigen Bestandteilen. Der Typ eines Fensters kann sein:

- Überschriftfenster,
- Mitteilungsfenster,
- Menüfenster,
- (Ein-/Ausgabe-) Maske,
- Grafikfenster.

Erläuterungen: Überschriftfenster helfen dem Benutzer bei der Orientierung, in welchem Teil des Programms er sich befindet. Sie enthalten nur eine Textausgabe. In Mitteilungsfenstern wird dem Benutzer gesagt, was das Programm momentan von ihm erwartet, welche Möglichkeiten er zur Programmfortsetzung hat oder welche Fehler aufgetreten sind. (Streng genommen sind auch Überschriftfenster nur Mitteilungsfenster mit einer speziellen Aufgabe. Der Unterschied liegt auf der semantischen Ebene.) Menüfenster präsentieren dem Benutzer eine Liste von Kommandos, aus denen er eines auswählen

kann (und muß), und zwar durch direkte Anwahl in diesem Fenster. (Daher sind auch Menüfenster nur Mitteilungsfenster mit einer speziellen Aufgabe. Das besondere an ihnen ist die Aktion des Benutzers im Fenster selbst. Ein Mitteilungsfenster, das etwa die Belegung von Funktionstasten beschreibt, hat zwar im Grunde auch die Funktion eines Menüfensters, ist aber keines im Sinne der Definition.) Masken sind die Fenster, die dem Benutzer den Blick auf eine Datenstruktur oder einen Teil davon erlauben. Sie bestehen aus Erläuterungen und Feldern, in denen Daten ausgegeben oder eingegeben und ediert werden können; Eingabefelder sind hervorgehoben (invers oder farbig). Grafikfenster dienen für grafische Ausgaben; sie besitzen ein Grafik-Koordinatensystem. Zusätzlich können sie auch Ausgabefelder enthalten. Also ist die *Definition*:

Ein **Überschriftfenster** ist ein Fenster mit einem Textausgabefeld. Ein **Mitteilungsfenster** ist ein Fenster mit einem oder mehreren Textausgabefeldern. Ein **Menüfenster** ist ein Fenster mit mehreren Kommandofeldern, von denen eines als aktuell gekennzeichnet ist. Eine **Maske** ist ein Fenster mit einer Menge von Textausgabefeldern, variablen Ausgabefeldern und Eingabefeldern. Ein **Grafikfenster** ist ein Fenster mit einer Menge von Ausgabefeldern und einer gitterförmigen Überdeckung durch eine Menge von Punkten (Pixeln).

Erläuterungen: Textausgabefelder sind während der gesamten Lebensdauer des Bildschirms konstant. Ausgabefelder geben den aktuellen Wert einer Variablen eines einfachen Typs wieder und können sich auf dem Bildschirm ändern. Eingabefelder dienen zur Aus- und Eingabe des Wertes einer Variablen eines einfachen Typs; sie können auch leer sein, wenn der Variablen noch kein Wert zugewiesen ist. Sie können vom Benutzer angesteuert und mit einem neuen Wert besetzt werden, der dann auch als Wert der Variablen übernommen wird. Der linke untere Punkt des Grafik-Fensters trägt die Koordinaten (0,0).

Ein Feld besteht aus einer Menge von Positionen (des Textbildschirms), einem Typ und einem Attribut. Der Typ kann sein:

- Textausgabefeld,
- (variables) Ausgabefeld,
- Eingabefeld.

Das Attribut kann sein:

- Hervorhebung,
- Inversdarstellung,
- Unterlegung mit einer Farbe,
- nichts von alledem.

Jeder Bildschirm wird durch ein Benutzerkommando aktualisiert und durch ein Benutzerkommando geschlossen. Ausnahme: Der erste Bildschirm, in der Regel ein Titelbild, wird durch den Programmstart aktualisiert.

Wie Bildschirme, Fenster und Felder für die Anforderungsdefinition zu spezifizieren sind, wird im folgenden Abschnitt beschrieben.

3 Die Spezifikation eines Bildschirms

Jede Bildschirmsituation im gesamten Programmablauf wird eindeutig (pixelgenau) festgelegt. Da sich Bildschirme und deren Ansteuerung unterscheiden, erzwingt diese Forderung eine gewisse (eigentlich unerwünschte) Geräteabhängigkeit, die einerseits unvermeidbar ist, andererseits aber durch die Verwendung von Konstantennamen ziemlich klein gehalten werden kann.

Ein Bildschirm wird durch folgende Angaben spezifiziert:

- Belegung der Attribute Modus, Zahl der Farben, Farbpalette.
- Ein Abbild mit der genauen Position von Fenstern und Feldern.
- Position der Schreibmarke (Cursor).
- Erzeugende Kommandos.
- Angaben der Reaktionen auf alle möglichen Benutzeraktionen, insbesondere Kommandos.
- Angaben der Reaktionen auf Bedienungsfehler und Fehlersituationen.
- Spezifikation der Fenster.
- Nötige Erläuterungen.

Ein Fenster wird durch folgende Angaben spezifiziert:

- Seinen Typ.
- Linke obere und rechte untere Ecke.
- Hintergrundfarbe, Schrift- und Grafikfarben.
- Spezifikation der Felder.
- Nötige Erläuterungen.

Ein Feld wird spezifiziert durch die Angaben:

- Typ des Feldes.
- Lage des Feldes, d. h., Angabe der zugehörigen Bildschirmpositionen.

- Konstanter Text bzw. Typ der entsprechenden Ein-Ausgabe-Variablen.
- Eines der Attribute: Hervorhebung, Inversdarstellung, Unterlegung mit einer Farbe.

4 Empfehlungen

Einige zusätzliche Empfehlungen für die Praxis, die nicht notwendig aus den aufgestellten Grundsätzen folgen, sollen als Ansätze zu einer guten Software-Ergonomie dienen; natürlich wird dieses Thema damit bei weitem nicht vollständig abgehandelt.

Um einen klaren, ruhigen, übersichtlichen Bildschirmaufbau zu erzielen, wird ergänzend empfohlen, daß gleichzeitig aktuell höchstens sein können:

- 1 Überschriftfenster,
- 1 Grafikfenster,
- entweder 1 Menüfenster oder 1 Maske,
- mehrere Mitteilungsfenster (aber nur in Ausnahmefällen mehr als 1, etwa ein Fenster mit Definitionen von Funktionstasten und eines für Fehlermeldungen).

Falls der Bildschirm eine Ein-/Ausgabemaske enthält, werden zulässige Kommandos in einem Mitteilungsfenster angezeigt und durch Betätigung von Funktions- oder Zeichentasten eingegeben. Andernfalls enthält der Bildschirm ein Menüfenster, und in diesem stehen alle Kommandos, die zum Bildschirm gehören. Die Kommandos werden dann durch Wahl mit Hilfe der Pfeiltasten und Abschluß mit der ENTER- oder RETURN-Taste eingegeben. Eine andere Form der Kommandoeingabe ist für keinen Bildschirm zugelassen. Benutzeraktionen sind nur in einem Fenster möglich; insbesondere kann die Schreibmarke, sofern sie überhaupt vorhanden ist, Fenstergrenzen nicht überspringen.

Eine Hilfefunktion ist mindestens dann anzubieten (und zwar sichtbar!), wenn nicht alle Kommandos angezeigt sind, was etwa bei Funktionstasten-Belegungen denkbar ist – aber nur in Ausnahmefällen akzeptabel. Weiter soll das Thema ‚Hilfefunktion' hier nicht diskutiert werden.

Bei Zugriffen auf externe Speichermedien (etwa Disketten) soll vor der Ausführung stets das Verzeichnis der auf der aktuellen Directory vorhandenen relevanten Dateien ausgegeben werden. Gleichzeitig ist dem Benutzer die Möglichkeit zur Directory-Änderung zu geben. Bei Lesezugriffen wird die gewünschte Datei wie ein Menüpunkt angewählt, das Fenster mit dem Verzeichnis also als Menüfenster behandelt. Bei Schreibzugriffen dagegen gilt das Fenster als Ein-/Ausgabemaske; es gibt ein einziges Eingabefeld, in

das der Name der Datei eingetragen wird. Gibt der Benutzer einen schon vorhandenen Namen ein, so darf angenommen werden, daß er wünscht, die alte Datei zu überschreiben.

Daß ein Bildschirm kein völlig statisches Gebilde ist, wird schon aus der Definition einer Maske klar. In der Praxis darf man sich als Erleichterung gestatten, auch gewisse Textausgabefelder als variabel zu betrachten. Z. B. kann es sinnvoll sein, eine Meldung erscheinen zu lassen, die mit dem nächsten Tastendruck wieder verschwindet. Im Sinne der Definition wäre das ein neuer Bildschirm, vor allem weil sich die Menge der Kommandos geändert hat. Größere Abweichungen von der Definition sollte man aber nicht durchgehen lassen. Tritt ein gleich aussehender Bildschirm in einem anderen Zusammenhang auf, so haben seine Kommandos teilweise unterschiedliche Wirkungen, und sei es nur, daß der auf ein Kommando folgende Bildschirm ein anderer ist. In diesem Fall müssen die gleich aussehenden Bildschirme streng auseinander gehalten werden. Auf diese Weise wird ein übersichtlicher baumartiger Zusammenhang der verschiedenen Bildschirme unterstützt, wenn auch nicht erzwungen.

Eingabefelder in einer Maske sind linear geordnet (z. B. durchlaufend numeriert) nach der Regel „zeilenweise links vor rechts, oben vor unten“. Die Schreibmarke steht zu Beginn am Anfang des ersten Feldes; sinnvolle Ausnahmen sind zugelassen. Die Schreibmarke springt nach Betätigung der Pfeiltasten in das nächste Feld der gewünschten Richtung. Ist dort kein Feld vorhanden, so geschieht nichts. (Es wird also neben der linearen Ordnung der Eingabefelder auch die zweidimensionale Anordnung auf dem Bildschirm berücksichtigt.) Bei Betätigung der ENTER-Taste wird der im aktuellen Feld stehende Wert übernommen und das nächste Feld in der linearen Ordnung angesprungen. Wird das Feld (auch nach Beginn des Edierens) ohne Betätigung der ENTER-Taste verlassen, wird der alte Wert restauriert. Fehleingaben werden nicht angenommen; in diesem Fall wird in einem (vorhandenen oder extra reservierten) Fenster eine Meldung der Art „Ungültige Eingabe. Bitte drücken Sie eine Taste.“ ausgegeben; für einen angemessenen Moment sind alle Tasten gesperrt, damit ein schnellschreibender Benutzer nicht schon aus Versehen die Löschung der Fehleingabe ausgelöst hat und seinen Fehler dann nicht mehr erkennen kann. Danach wird auf beliebigen Tastendruck der vor der Fehleingabe bestehende Bildschirm restauriert. Dies ist ein Beispiel für die weiter oben erörterte Frage, wie weit durch eine solche Meldung ein neuer Bildschirm definiert wird.

In einem Menüfenster gilt sinngemäß das gleiche über die Betätigung von Pfeil- und ENTER-Tasten wie in einer Maske; allerdings sollte man nur in Ausnahmefällen von der linearen Anordnung der Menüpunkte auf dem physikalischen Bildschirm abweichen.

Ein vorzügliches Spezifikationshilfsmittel ist übrigens kariertes Papier mit einem Karo pro Bildschirmposition. Es erleichtert die präzise Beschreibung, ist anschaulich und übersichtlich und erspart viele Worte.

Für die Arbeitsschritte bei der Anforderungsdefinition hat sich folgende zeitliche Reihenfolge bewährt:

- Pflichtenheft,
- Spezifikation der Daten,
- Benutzer-Oberfläche,
- Funktionshandbuch,
- Benutzer-Handbuch.

Natürlich können je nach Projekt auch andere Reihenfolgen, vor allem aber Überschneidungen sinnvoll sein.

5 Folgerungen für den Entwurf

Der Entwurf beginnt mit der Grobzerlegung des Gesamtsystems in Teilsysteme. Die Spezifikation der Benutzer-Oberfläche nach der vorgeschlagenen Methode hat Auswirkungen auf zwei Teilsysteme: dasjenige, das die Steuerung des Programmablaufs verkapselt, und dasjenige, das die Bildschirmgestaltung oder Benutzeroberfläche im engeren Sinne (den unintelligenten Teil davon) verkapselt, vergleiche [Engels/Perl].

Da sich aus der vorgeschlagenen Spezifikation eine klare Gliederung des Programmablaufs in eine Folge Bildschirm – Befehl – Bildschirm – Befehl – ... zwingend ergibt, wird das Programm durch eine einfache Hauptschleife gesteuert, die in Pseudo-Code so aussieht:

- Wiederhole:
  - Gib den aktuellen Bildschirm aus.
  - Nimm Benutzerkommando entgegen.
  - Übersetze das Benutzerkommando in die entsprechende Aktion und führe sie aus.
  - Bestimme den neuen aktuellen Bildschirm.

  solange bis der aktuelle Bildschirm das Endbild und das Benutzerkommando das Endkommando ist.

Der Fall, daß während der Ausgabe eines Bildschirms selbständig Aktionen ablaufen, die vom Benutzer nach Belieben gestoppt werden können, wird erfaßt, wenn man auch das leere Kommando zuläßt; wird dieses entgegengenommen, so läuft die selbständige Aktion einen Schritt weiter. (Diese Idee verdanke ich meinen Studenten im Software-Praktikum.)

Natürlich enthält die Steuerung noch Teilsysteme, die die nötigen Prozeduren zur Ausführung der in der Hauptschleife gestellten Aufgaben verkapseln, soweit dazu Intelligenz, d. h., Kenntnis der Semantik nötig ist:

- semantisch korrekte Besetzung des auszugebenden Bildschirms mit den aus dem Problembereich des Gesamtsystems stammenden Daten,
- semantische Analyse der Benutzerkommandos (die ja vom Teilsystem ‚Benutzer-Oberfläche' nur in der Form „Benutzer hat Menüpunkt x angewählt" übergeben werden),
- semantisch korrekte Datenmanipulation.

Die Details sind in [Engels/Perl] genauer beschrieben. Diese Teilsysteme sind sehr stark problemabhängig, so daß man kaum allgemeine Prinzipien für sie aufstellen kann. Die Spezifikationsmethode liefert aber auch hier sehr brauchbare Anhaltspunkte wie etwa Kommando-Umsetzungstabellen.

Für den Entwurf des Teilsystems Benutzeroberfläche/Bildschirmgestaltung liefert die Spezifikation klare Vorgaben über die benötigten Fenster, System- und Fehlermeldungen, Ein- und Ausgabeprozeduren für Daten und Kommandos. Insbesondere wird die Entscheidung für ein bestimmtes Fenstersystem erleichtert, das mit möglichst wenig Aufwand die Anforderungen erfüllt.

Die nicht zu vermeidende Geräteabhängigkeit bei der Spezifikation der Bildschirme schadet natürlich der Portabilität des Systems. Man kann aber hier doch hinreichend flexibel bleiben, wenn man die physikalischen Bildschirmkonstanten, wie Zahl der Zeilen und Spalten im Textmodus, Zahl der Bildpunkte im Grafikmodus, Farbpaletten, in einen eigenen Modul packt.

Die genaue Beschreibung immer wiederkehrender Standardsituationen, wie etwa ‚Hauptmenü', und ihre problemunabhängige Formulierung erlaubt den Aufbau von Werkzeug-Sammlungen und erleichtert so die Implementation. Es ist eine Binsenweisheit, die wie auf die anderen Teilsysteme auch auf die Benutzer-Oberfläche zutrifft, daß die problemabhängigen Teile in ein eigenes, möglichst kleines Teilsystem eingesperrt werden sollen. Bei der Benutzer-Oberfläche gilt das auch für die firmenspezifischen Teile, die geschmacksabhängige Design-Fragen betreffen und das typische wiedererkennbare Erscheinungsbild des Softwareprodukts definieren.

6 Anhang: Ein Beispiel

Der im folgenden abgebildete und spezifizierte Bildschirm gehört zum Software-Projekt „Simulation dynamischer Systeme" am Fachbereich Mathematik in Mainz, Sommersemester 1987.

Modus: 80-Zeichen.

Farben: 2.

Farbpalette: schwarz, weiß.

Abbild: siehe nächste Seite.

Schreibmarke: unsichtbar.

Erzeugende Kommandos: „Start der Berechnung“ auf Bildschirm 1 (Hauptmenü).

Reaktionen auf Benutzeraktionen:

- Keine Aktion: Die Berechnung wird eine Zeiteinheit weitergeführt.
- Pfeiltasten, ENTER: Die Berechnung wird (nach Abschluß des aktuellen Berechnungsschritts) gestoppt und erst nach Abschluß der Wahl durch ENTER und den daran eventuell anschließenden Programm-Aktionen mit den möglicherweise geänderten Parametern oder Darstellungsoptionen fortgeführt.
- Andere Tasten: keine Reaktion.

Bedeutung der Menüpunkte: ....

Fehlersituationen: Bei Wahl von „Drucker“ ist der Drucker nicht bereit. Dann ....

Spezifikation der Fenster:

- 1 Überschriftfenster.

  Ecken: (1,61) und (3,80).

  Hintergrundfarbe: schwarz.

  Schriftfarbe: weiß.

  1 Textausgabefeld: siehe Abbildung, kein Attribut.
- 1 Menüfenster.

  Ecken: (4,61) und (20,80).

  Hintergrundfarbe: schwarz.

  Schriftfarbe: weiß.

  15 Textausgabefelder (für Menüpunkte).

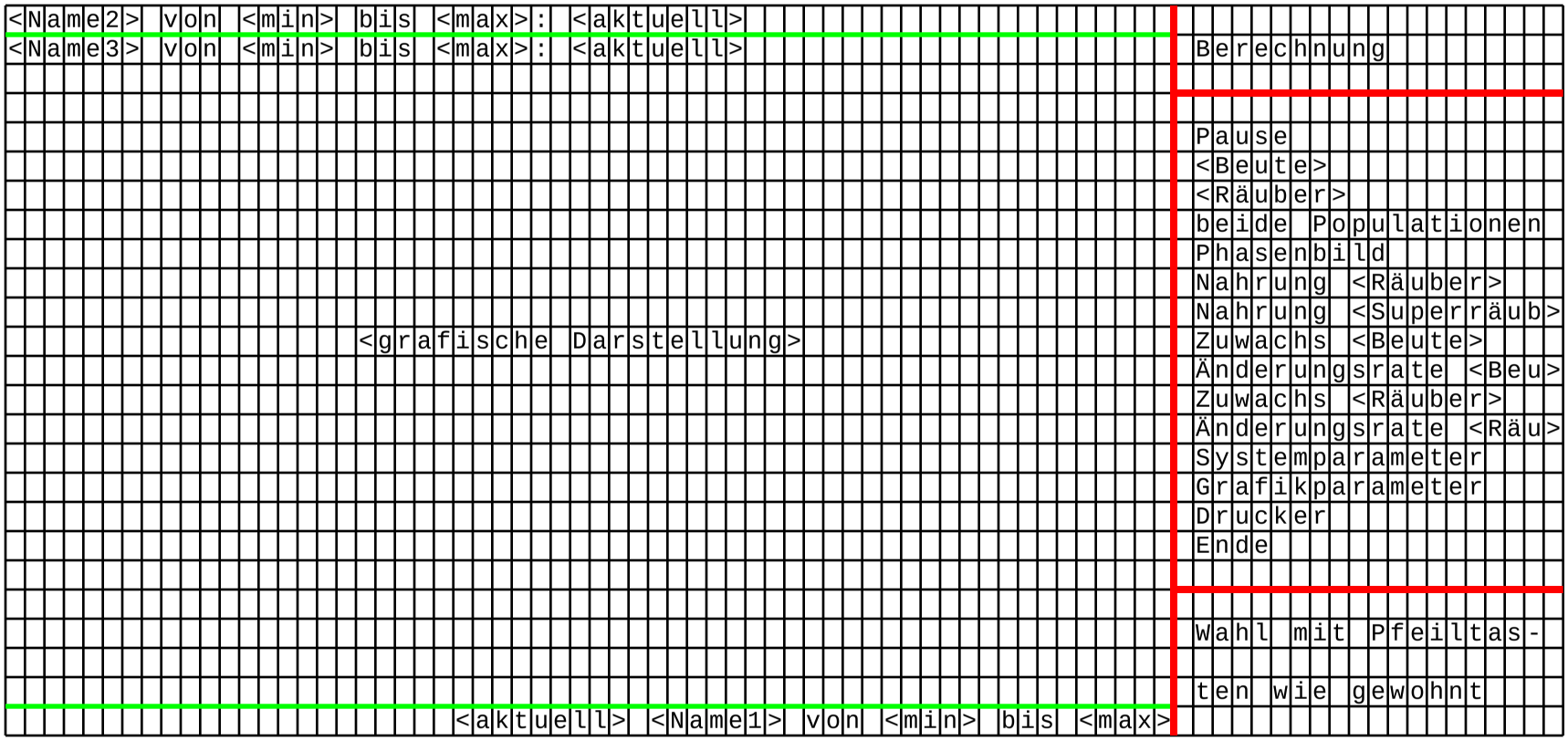
  Lage und Inhalt: siehe Abbildung; für `Beute` usw. werden die aktuellen Namen eingesetzt, auf den vorhandenen Raum passend verkürzt.

  Attribute: Invers dargestellt (und somit aktuell) ist beim ersten Erscheinen des Bildschirms der Punkt „beide Populationen“, sonst der zuletzt angewählte Punkt.
- 1 Mitteilungsfenster.

  Ecken: (21,61) und (25,80) usw.

- 1 Grafikfenster.
  Ecken: (1,1) und (25,60).
  Hintergrundfarbe: weiß.
  Schriftfarbe: schwarz.
  3 Textausgabefelder: siehe Abbildung, keine Attribute.

Erläuterungen. Die Ausgabefelder im Grafikfenster dienen zum Beschriften der Achsen. Die beiden oberen bezeichnen die Ordinate (y-Achse) und werden linksbündig beschrieben. Das untere gilt für die Abszisse (x-Achse) und wird rechtsbündig beschrieben. `Name1` bis `Name3` werden durch die Namen der dargestellten Variablen ersetzt; das Feld für `Name3` usw. wird nur benützt, wenn beide Populationen dargestellt werden. Die Namen werden auf 24 Zeichen verkürzt, falls sie länger sind, `min` und `max` werden jeweils durch den minimalen und maximalen darstellbaren Wert ersetzt, `aktuell` durch den Wert der Variablen zum aktuellen Zeitpunkt, den die Berechnung gerade erreicht hat. Dieser Wert wird auch in die grafische Darstellung übernommen und mit dem entsprechenden Vorgängerpunkt (falls es einen gibt) durch eine gerade Linie verbunden. Die Koordinatenachsen sind so weit sichtbar, wie sie innerhalb des dargestellten Teils der Koordinatenebene liegen. Falls beim Erscheinen des Bildschirms schon ein Teil der Berechnung abgelaufen ist, wird die grafische Darstellung bis zum aktuellen Zeitpunkt aufgebaut, bevor eine Benutzeraktion möglich ist.


<Name2> von <min> bis <max>: <aktuell>
<Name3> von <min> bis <max>: <aktuell>
Berechnung
Pause
<Beute>
<Räuber>
beide Populationen
Phasenbild
Nahrung <Räuber>
Nahrung <Superräub>
<grafische Darstellung>
Zuwachs <Beute>
Änderungsrate <Beu>
Zuwachs <Räuber>
Änderungsrate <Räu>
Systemparameter
Grafikparameter
Drucker
Ende
Wahl mit Pfeiltas-
ten wie gewohnt
<aktuell> <Name1> von <min> bis <max>

Literatur

H. Balzert: *Die Entwicklung von Software-Systemen.* B.I., Mannheim 1982.

G. Engels/ J. Perl: *Überlegungen zum Programmieren im Großen.* Preprint, Mainz 1987.

G. Pomberger: *Softwaretechnik und Modula-2.* Hanser, München 1984.