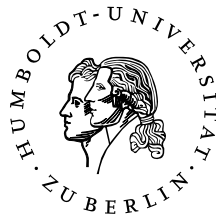


Schnittstellen statistischer Software – Analyse und Implementierung

Diplomarbeit

zur Erlangung des Grades
eines Diplom-Kaufmanns

an der Wirtschaftswissenschaftlichen Fakultät
der Humboldt-Universität zu Berlin



vorgelegt von

Uwe Ziegenhagen

(Matrikel-Nr. 156445)

1. Prüfer: Prof. Dr. Wolfgang Härdle

2. Prüfer: Prof. Dr. Bernd Rönz

Berlin, 12. September 2003

Inhaltsverzeichnis

Abkürzungsverzeichnis	V
Tabellenverzeichnis	VI
Abbildungsverzeichnis	VII
1 Einleitung	1
2 Analyse	3
2.1 S-Plus 2000	3
2.1.1 Die Integrierte Skriptsprache	4
2.1.2 Import von Dateien	5
2.1.3 Der S-Plus Dateneditor	6
2.1.4 Der S-Plus Objekteditor	8
2.1.5 Export von Daten und Grafiken	9
2.1.6 Hilfesystem	10
2.2 R	10
2.2.1 Das R - Userinterface	11
2.2.2 Der integrierte Dateneditor	11
2.2.3 Grafik-Export	12
2.2.4 Die Linux-Version von R	13
2.2.5 Hilfesystem	13
2.3 Minitab 13	15
2.3.1 Datenimport	15
2.3.2 Der Minitab-Dateneditor	16
2.3.3 Grafikexport	18
2.3.4 Hilfesystem	18
2.4 SPSS 11	18
2.4.1 Dateneditor	19
2.4.2 Datenimport	21
2.4.3 Export von Daten und Grafiken	23
2.4.4 Hilfesystem	23
2.5 Gauss 5	25
2.5.1 Import und Export	26

2.5.2	Dateneditor	26
2.5.3	Grafikexport	27
2.5.4	Hilfesystem	27
2.6	Statistica 6	28
2.6.1	Datenerstellung	28
2.6.2	Datenimport	29
2.6.3	Der Tabelleneditor in Statistica 6	32
2.6.4	Der Workbook-Manager	37
2.6.5	Export-Optionen	37
2.6.6	Hilfesystem	38
2.7	Stata 8/SE	38
2.7.1	Datenimport	39
2.7.2	Datenmanipulation	40
2.7.3	Grafikexport	41
2.7.4	Export	41
2.7.5	Hilfesystem	42
2.8	Tabellarische Übersichten	43
2.9	Fazit	44
3	Implementierung	46
3.1	Die grafische Benutzeroberfläche	46
3.2	Der Dateneditor	46
3.2.1	Laden von Datensätzen	46
3.2.2	Das Popup-Menü	48
3.3	Die xplora.ini	49
3.4	Wechsel zwischen Zeilen- und Spaltenmodus	50
4	Ausblick	51
4.1	Integrierter Hilfebrowser	51
4.2	Internationalisierung	52
4.3	Parser	52
4.4	Grafik	52
4.5	UML	53
5	Anhang	55
5.1	Microsoft Hilfedateien	55
5.2	Kopien zitierter Literatur	55
5.3	Quellcodes	63
5.3.1	table_edit.tex	63
5.3.2	unit1.cpp	66
5.3.3	Output.cpp	67
5.3.4	openform.cpp	68
5.3.5	mdiapp.cpp	71
5.3.6	main.cpp	72

Inhaltsverzeichnis

5.3.7 Input.cpp 74
5.3.8 Childwin.cpp 74
5.3.9 About.cpp 75
Literaturverzeichnis **76**

Abkürzungs- und Begriffsverzeichnis

APSS	Auto Pilot Support System
BMP	Windows Bitmap Format
CHM	Compiled HTML Helpformat
CSV	Comma Separated Values
DASL	Data and Story Library
DDE	Direct Data Exchange
DPI	Dots per Inch
DSN	Data Name Source
EPS	Enhanced Postscript
GNU	rekursives Akronym „GNU IS NOT UNIX“
HPGL	Hewlett Packard Graphics Language
ISO	International Organization for Standardization
JPEG	Grafikformat Joint Picture Experts Group;
MD	Missing Data Code in Statistica 6
MDI	Multiple Document Interface
n.v.	nicht verfügbar
ODBC	Open DataBase Connectivity
OLE	Object Linking and Embedding
PDF	Potable Document Format
PNG	Portable Network Graphics; Grafikformat
PS	Postscript
SQL	Structured Query Language
SVG	Scalable Vector Graphics
UML	Unified Modelling Language

Tabellenverzeichnis

2.1	S-Plus: Importformate	6
2.2	Statistica: Importformate	30
2.3	Statistica: Exportformate	37
2.4	Stat/Transfer: unterstützte Dateiformate	39
2.5	Stata: Dateiformate für Grafikexport	41
2.6	Übersicht Statistikprogramme	43
2.7	eingebaute Datenimportformate	44
2.8	Formatübersicht Grafikexport	44
5.1	„Companies“ Datensatz	57

Abbildungsverzeichnis

1.1	Anzahl der Transistoren pro Prozessor (Intel (2003)	2
2.1	S-Plus: Auswahlfenster	4
2.2	S-Plus: Dateneditor	6
2.3	S-Plus: Benennung von Zeilen/Spalten im Editor	7
2.4	S-Plus: Object Explorer	8
2.5	S-Plus: Ausschnitt aus der Workspace-Datei	8
2.6	R: Grafische Benutzeroberfläche (Windows)	11
2.7	R: Matrixeditor (Windows)	12
2.8	R: Editor unter Linux	13
2.9	R: eingebautes Hilfesystem (Windows)	14
2.10	Minitab: Grafische Benutzeroberfläche	15
2.11	Minitab: Importoptionen	16
2.12	Minitab: Dateneditor	17
2.13	SPSS: Benutzeroberfläche	19
2.15	SPSS: ODBC-Importoptionen	21
2.14	SPSS: Import von Textdateien	22
2.16	SPSS: HTML Tutorial	23
2.17	SPSS: Statistics Coach	24
2.18	Gauss: grafische Oberfläche	25
2.19	Gauss: Matrix Editor	26
2.20	Statistica: Erstellen neuer Dokumente	28
2.21	Statistica: ODBC-Importdialog	29
2.22	Statistica: ODBC-Importauswahl	30
2.23	Statistica: Import von Excel-Tabellen	31
2.24	Statistica: ASCII Text Import „Free“	31
2.25	Statistica: Text Import „Auto“	32
2.26	Statistica: Importierter Companies-Datensatz	32
2.27	Statistica: Variablen-Editor	33
2.28	Statistica: Tabelleneditor für Variablentypen	34
2.29	Statistica: Values/Stats	34
2.30	Statistica: Kontextmenü der Tabellen	35
2.31	Statistica: Fill/Standardize Menü	35
2.32	Statistica: blockweise Anwendung von Methoden	36

Abbildungsverzeichnis

2.33	Statistica: blockweise Anwendung von Grafiken	36
2.34	Statistica: ActiveX Komponenten im Workbook-Manager . . .	37
2.35	Stata: Benutzeroberfläche	38
2.36	Stata: Stat/Transfer	39
2.37	Stata: Stat/Transfer Optionen	40
2.38	Stata: Variableneditor	41
2.39	Stata: Export-Dialog	42
2.40	Stata: Hilfesystem	43
3.1	Yxilon: grafische Benutzeroberfläche	47
3.2	Yxilon: Dateneditor	47
3.3	Yxilon: Open Data-Dialog	48
4.1	Yxilon: Übersicht	51
4.2	UML: Diagrammbeispiele	54

Danksagung

Besonderer Dank gilt denjenigen, die mich bei meiner Arbeit unterstützt haben. An erster Stelle möchte ich meinen Eltern danken, die mir stets hilfreich zur Seite gestanden haben.

Bedanken möchte ich mich auch bei meinem Betreuer, Prof. Dr. Wolfgang Härdle, für die Unterstützung bei der Entstehung dieser Arbeit.

Weiterhin danke ich meinen Freunden und Kollegen, die mir wertvolle Ratschläge gaben: Simon Borak, Marko Frank, Dr. Sigbert Klinke, Martin Schröter und Rodrigo Witzel.

1 Einleitung

Die Geschichte der computergestützten Statistik ist eng mit der Weiterentwicklung der Computertechnologie verknüpft. Benötigte man in den 60er und 70er Jahren des vergangenen Jahrhunderts noch millionenteure Rechenanlagen, die Lochkarten gesteuert, Tage und Wochen für die Lösung einer Problemstellung benötigten, hat sich dies mit der Entwicklung des Personalcomputers dramatisch geändert. Jeder Computerbesitzer verfügt heute mit seinem heimischen PC über eine Rechenleistung, die jene der NASA zu Zeiten des Apollo-Programms um ein Vielfaches übertrifft; viele Auswertungen und Verfahren (Genom-Analyse, Neuronale Netze) wurden überhaupt erst durch die enorme Rechenleistung, die heute zur Verfügung steht, möglich gemacht. Bild 1.1 zeigt recht anschaulich, wie sich die Zahl der Transistoren in den letzten Jahrzehnten entwickelt hat.

Da unzureichende Rechenkraft bei der Auswertung von statistischen Daten in den meisten Fällen keine Probleme bereitet, gewinnen andere Aspekte an Bedeutung. *Usability* ist eines der Schlagwörter, die immer häufiger im Zusammenhang mit Software und Softwareentwicklung fallen.

Die ISO 9241, zitiert in Eichinger (2001), definiert *Usability* eines Produkts als

„[. . .] das Ausmaß, in dem es von einem bestimmten Benutzer verwendet werden kann, um bestimmte Ziele in einem bestimmten Kontext effektiv, effizient und zufriedenstellend zu erreichen.“

Eine andere Definition von Jacob Nielsen, einem der führenden Usability-Gurus, lautet (ebenfalls entnommen Eichinger (2001)):

„Usability is the measure of the quality of the user experience when interacting with something – whether a Web site, a traditional software application, or any other device the user can operate in some way or another.“

Zur *Usability* von Statistik-Software zähle ich insbesondere den Umgang mit Daten, daher möchte ich untersuchen, wie verschiedene Statistik-Programme den Nutzer beim Import, bei der Verwaltung und beim Export von Daten und Grafiken unterstützen.

1 Einleitung

Dazu wird im ersten Kapitel eine Reihe von Programmen betrachtet und die Art und Weise, wie sie mit Daten umgehen, d.h. welche Möglichkeiten des Imports und Exports verfügbar sind und wie komfortabel diese gestaltet sind. Der zweite Teil beschäftigt sich dann mit der Umsetzung der gewonnenen Erkenntnisse in C++. Dazu entwickelt und implementiert der Autor den Date-Editor für die grafische Benutzeroberfläche, welche sich an das bestehende XploRe anlehnt und eine der GUIs des Yxilon-Projekts sein wird.

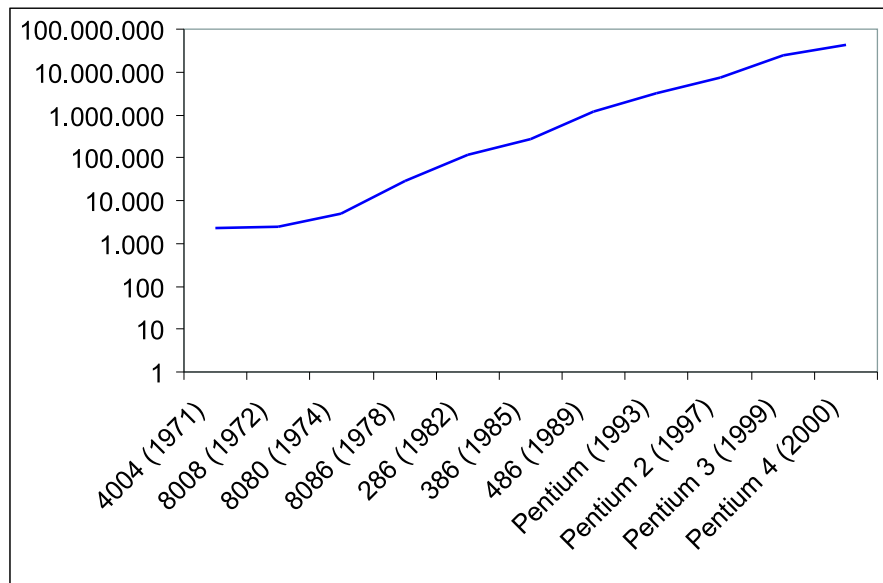


Abbildung 1.1. Anzahl der Transistoren pro Prozessor (Intel (2003))

2 Analyse

Folgende Statistik-Programme sollen in diesem Abschnitt näher untersucht werden:

- *S-Plus 2000* (Studentenversion)
- *R* (Windows/Linux)
- *Minitab 13* (Demoversion)
- *Gauss 5.02* (Demoversion)
- *SPSS 11.0*
- *Stata 8/SE*
- *Statistica 6*

Bei jeder Anwendung sollen folgende Fragestellungen berücksichtigt werden:

Datenimport: Wie komfortabel ist der Import aus anderen Anwendungen angelegt, gibt es Assistenten für den Datenimport? Welche Formate können importiert werden?

Datenmodifikationen: Wie lassen sich die eingegebenen Daten verwalten? Welche Möglichkeiten hat der Nutzer, die Daten zu ändern und wie bequem sind diese? Welche Hilfsmittel stehen ihm dabei zur Verfügung?

Daten- und Grafikexport: Welche Exportformate werden unterstützt? Besteht auch die Möglichkeit, in vektorbasierte Formate wie Postscript und SVG zu exportieren?

2.1 S-Plus 2000

Die amerikanische Firma Insightful ist der exklusive Lizenznehmer der S-Sprache, die an den berühmten Bell Labs der US-Telefongesellschaft AT&T im Jahr 1976 entwickelt wurde.

2 Analyse

Ziel des Teams um John Chambers, der für das Design von S 1999 den Software System Award der „Association for Computing Machinery“ erhielt, war die Unterstützung von grundlegenden Datenanalysen der Bell Statistics Group mit einem einheitlichen Set an Modulen.

S-Plus 2000 ist eine MDI-Anwendung (Multiple Document Interface), einzelne Komponenten zur Daten- und Programmmanipulation werden daher innerhalb des MDI-Hauptfensters dargestellt.

Die Arbeit mit *S-Plus* wie auch mit anderen Paketen beginnt in der Regel mit dem Einladen oder Erstellen der Daten. Dazu hat der Nutzer von *S-Plus* drei Möglichkeiten:

- Erstellen eines neuen Datenobjekts mit der integrierten Skriptsprache
- Importieren einer Datendatei
- Erstellen einer Datenmatrix mit dem Dateneditor



Abbildung 2.1. S-Plus: Auswahlfenster

2.1.1 Die Integrierte Skriptsprache

Legt man über den Befehl `File→New→Script File` eine neue Datei an, kann beispielsweise mit `new <- array(c(1:25,1:25), dim = c(25,25))` eine neue 25×25 Matrix erstellt werden, die dann über den Namen `new` angesprochen werden kann. Gleichzeitig erscheint die Matrix im Object Explorer (siehe Bild 2.1.4) unter dem Menüpunkt `Data`.

2.1.2 Import von Dateien

Zum skriptgesteuerten Import von Daten stehen zwei Befehle zur Verfügung:

- scan
- read.table

read.table ist die einfachste Möglichkeit, tabellenähnliche Daten zu importieren. Der Befehl ist dabei eine Kapselung von scan. Im Gegensatz zu scan wird das Format der Daten automatisch erfaßt, was bei großen Dateien zu Leistungseinbußen führen kann.

So liest der folgende Befehl den *Companies*-Datensatz aus einer tabulatorgetrennten Textdatei und berücksichtigt dabei, daß in der ersten Zeile bzw. Spalte die Namen der Variablen und Fälle stehen.

```
companies<-read.table("G:/uwe/Datensaetze/dataset.txt",  
header = TRUE, row.names = 1)
```

scan ist ein allgemeineres Kommando zum Lesen von Textdaten und verfügt über eine ganze Reihe von Optionen, mit denen die Anzahl der zu lesenden Datensätze, der Separator usw. spezifiziert werden können.

Der Befehl `File→New→Import Data` enthält die grafischen Gegenstücke zu den Skriptbefehlen.

From File: stellt die Funktionen des scan-Befehls unter der grafischen Oberfläche zur Verfügung und bietet über den Options- und Filterdialog eine einfachere Möglichkeit als die Kommandozeile zur Auswahl der gewünschten Daten. Zusätzlich lassen sich über diesen Dialog auch andere Dateiformate importieren (siehe Bild 2.1).

From ODBC Connection: Über die ODBC-Schnittstelle kann jedes Programm, das dieses Protokoll unterstützt, aus ODBC-Quellen lesen. Nahezu alle Datenbanksysteme und Tabellenkalkulationen unterstützen dieses Format. Dadurch kann S-Plus direkt diese Quellen lesen, ohne Kenntnisse über die interne Dateistruktur der Wirtssysteme haben zu müssen.

Bloomberg, Fame, Mim: Über die Bloomberg-Schnittstelle erhält der Benutzer Zugriff auf eine der größten Finanzdatenbanken der Welt. Mit FAME (www.fame.com) und MIM (www.lim.com) werden noch zwei weitere Marktinformationssysteme unterstützt, die Finanzmarktdaten über das Internet ihren Nutzern zur Verfügung stellen.

Insbesondere für Nutzer aus dem Finanzwesen sind diese letzten drei Schnittstellen interessant, da so jederzeit auf Echtzeit-Informationen von den wichtigsten Börsenplätzen zurückgegriffen werden kann.

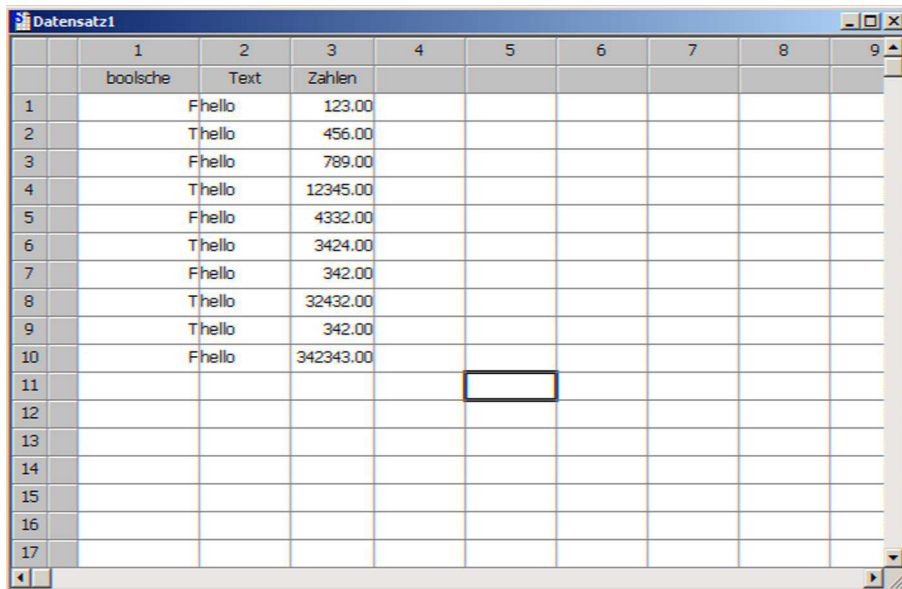
2 Analyse

ASCII	Borland Paradox
DBase	Quattro Pro
Gauss	SAS
Lotus 1-2-3	SPSS
Matlab	STATA
Microsoft Access	SYSTAT
Microsoft Excel	Sigmaplot

Tabelle 2.1. S-Plus: Importformate

2.1.3 Der S-Plus Dateneditor

Der Dateneditor bietet dem Benutzer eine Tabellenumgebung zur Definition von Variablen und Eingabe von Werten.



The screenshot shows a window titled 'Datensatz1' containing a data editor table. The table has 9 columns and 17 rows. The first two columns are labeled 'boolsche' and 'Text', and the third is 'Zahlen'. The data is as follows:

	1	2	3	4	5	6	7	8	9
	boolsche	Text	Zahlen						
1		Fhello	123.00						
2		Thello	456.00						
3		Fhello	789.00						
4		Thello	12345.00						
5		Fhello	4332.00						
6		Thello	3424.00						
7		Fhello	342.00						
8		Thello	32432.00						
9		Thello	342.00						
10		Fhello	342343.00						
11									
12									
13									
14									
15									
16									
17									

Abbildung 2.2. S-Plus: Dateneditor

Die beiden ersten Spalten sowie die obersten zwei Zeilen stehen nicht für die Dateneingabe zur Verfügung, sie dienen vielmehr dem Hinzufügen von Metadaten. Geht der Nutzer mit dem Mauscursor über eine der beiden ersten Zeilen jeder Spalte, verändert sich der Mauscursor zu einem Pfeil und durch einen Mausklick kann dann die gesamte Spalte markiert werden.

Die beiden ersten Spalten jeder Zeile dienen analog dazu, per Mausklick die gesamte Zeile zu markieren.

Erste Spalte und Zeile dienen der Anzeige der aktuellen Zelle, die zweite Spalte enthält die Zeilenbeschriftungen, die der Nutzer optional per Doppelklick

2 Analyse

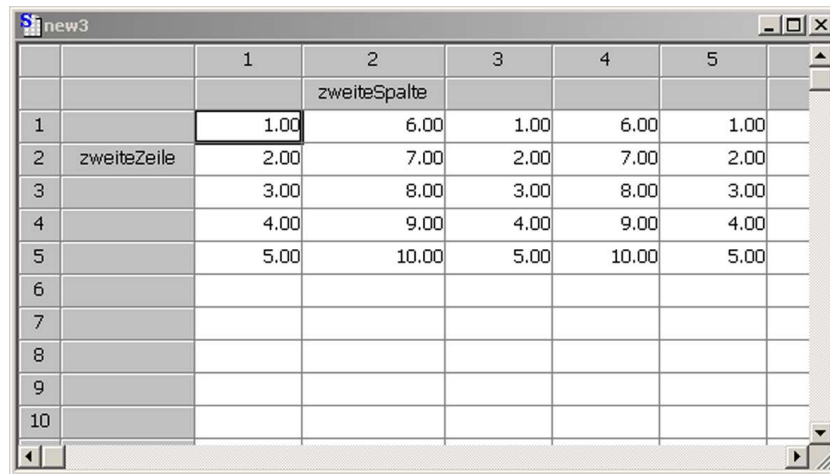
hinzufügen kann. Beim Einfügen von Daten wird automatisch in die zweite Zeile ein Variablenname eingefügt, der ebenfalls per Doppelklick verändert werden kann.

S-Plus verfügt auch über einen intelligenten Einfügemodus: Trägt der Nutzer in die erste Datenzeile einer Spalte eine Zahl ein, wird der Typ der Spalte automatisch auf *Double* gesetzt. Sollten in die weiteren Zeilen der Spalte Zeichenketten eingetragen werden, wird „NA“ für einen fehlenden Wert eingetragen.

Beginnt der Benutzer hingegen mit einer Zeichenkette (String) in der ersten Zeile, wird der Spaltentyp auf *Factor* (Faktorvariable) gesetzt. In den weiteren Zellen kann dann über eine Dropdown-Liste auf die bisher eingegebenen Werte zurückgegriffen werden, was sich in der Praxis als sehr komfortabel erweist.

Über den Unterpunkt *Properties* im Kontextmenü (das auch die Befehle für die Zwischenablage enthält), kann der Spaltenname sowie die Spaltenbreite und Ausrichtung der Spalte verändert werden.

Die Typen der einzelnen Spalten lassen sich über das Untermenü *Change Data Type* aufrufen, zu den verfügbaren Datentypen gehören: character, complex, dates, double, factor, integer, logical und single.



	1	2	3	4	5
		zweiteSpalte			
1	1.00	6.00	1.00	6.00	1.00
2	zweiteZeile	2.00	7.00	2.00	7.00
3	3.00	8.00	3.00	8.00	3.00
4	4.00	9.00	4.00	9.00	4.00
5	5.00	10.00	5.00	10.00	5.00
6					
7					
8					
9					
10					

Abbildung 2.3. S-Plus: Benennung von Zeilen/Spalten im Editor

2.1.4 Der S-Plus Objekteditor

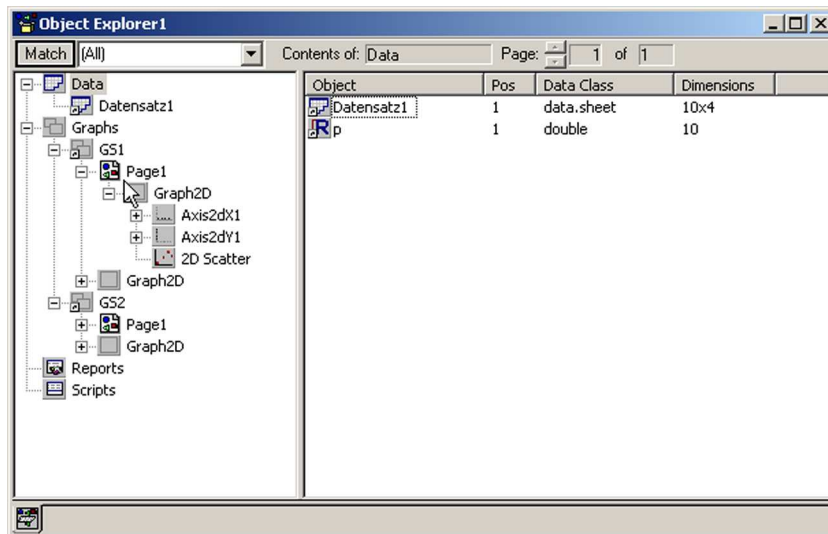


Abbildung 2.4. S-Plus: Object Explorer

S-Plus benutzt eine sogenannte Workspace-Datei, in der die geöffneten Objekte mit ihren Fensterpositionen aufgelistet sind. Diese Datei enthält jedoch nur die Referenzen auf die geöffneten Objekte, d.h. die Objekte selbst sind einzeln in Dateien gespeichert.

```
guiCloseAllDocuments();  
guiRefreshMemory();  
.attach("C:\\SP2000SE\\USERS\\ADMINISTRATOR\\_DATA", pos=1)  
guiOpenView("data.sheet", Name = "DS2",  
            Hide = F,  
            Show = "Normal",  
            Top = "61",  
            Left = "175",  
            Width = "1846",  
            Height = "454",  
            Duration = "0")
```

Abbildung 2.5. S-Plus: Ausschnitt aus der Workspace-Datei

Das zentrale Hilfsmittel zur Verwaltung der einzelnen Objekte in *S-Plus* ist der *Object Explorer*, den man mit `File→New→Object Explorer` aufruft.

Im *Object Explorer* lassen sich einzelne Seiten erstellen, die mit eigenen Bildern versehen werden können. In jeder dieser Seiten kann der Benutzer dann einzelne Ordner oder Objekte wie Datensätze, Grafiken und Skripte einfügen.

2 Analyse

Auf diese Weise lassen sich die in einem Projekt benutzten Dateien bequem verwalten.

Analog zum Windows Explorer bietet der *Object Explorer* auch die Möglichkeit, zwischen verschiedenen Ansichten wie großen oder kleinen Icons oder der Detailansicht zu wählen.

Ähnlich wie der *S-Plus* Workspace werden die einzelnen Objekte wie Datensätze, Reports und Skripte jedoch nicht im *Object Explorer* selbst gespeichert.

2.1.5 Export von Daten und Grafiken

S-Plus bietet eine Vielzahl von Exportfiltern an, um die benutzten Daten und erstellten Ergebnisse in andere Formate zu überführen.

Nativ ist *S-Plus* in der Lage, folgende Formate zu schreiben:

ASCII	Borland Paradox
DBase	Quattro Pro
Gauss	SAS
Lotus 1-2-3	SPSS
Matlab	STATA
Microsoft Access	SYSTAT
Microsoft Excel	

Aus einer Vielzahl von Grafikformaten sind die wichtigsten:

Adobe Photoshop	JPEG
GIF	Windows BMP
PNG	TIFF
EPS	Windows Metafile
EXIF	Corel Wordperfect

Zusätzlich verfügt *S-Plus* über ein Exportmodul für Microsoft Powerpoint, das eine Liste von Grafiken zu einer Powerpoint-Präsentation zusammenstellt, die dann in Powerpoint selbst weiterbearbeitet werden kann.

2.1.6 Hilfesystem

Zusätzlich zum Hilfesystem (mit Sprachreferenz) im Windows HLP-Format beinhaltet *S-Plus* die *Mathsoft S-PLUS 2000 Object Hierarchy* im HTML-Format sowie eine Reihe von PDF-Dateien:

- Getting Started
- User's Guide
- Programmer's Guide
- Guide to Statistics Vol. 1
- Guide to Statistics Vol. 2

2.2 R

Bei *R* handelt es sich nach Leisch (2000) um die „Neuimplementierung der Statistiksprache *S*, bestehend aus einem Laufzeitsystem (Interpreter), Grafik und vielen Toolboxes.“

R hat seinen Ursprung in der Arbeit von Robert Gentleman und Ross Ihaka (Statistik Abteilung der Universität Auckland), die die zu *S* erschienenen Handbücher (BLAU, WEISS und GRÜN genannt) als Basis genommen haben, um die Sprache neu zu implementieren.

Was im Jahr 1993 in Auckland begann, hat sich innerhalb weniger Jahre in eine ernsthafte Alternative zu den Paketen der etablierten Anbieter wie *SAS* und *S-PLUS* entwickelt. Durch die kostenlose Verfügbarkeit von *R* für eine Vielzahl an Plattformen hat sich eine weltweite Nutzergemeinde zusammengeschlossen, die *R* mit Funktionen und Algorithmen versorgt. Seit 1995 steht *R* unter der GNU PUBLIC LICENSE (www.gnu.org), die die freie Verfügbarkeit der Programmquellen garantiert.

Da Sprachumfang und -syntax von *R* und *S-Plus* nahezu identisch sind, funktionieren die meisten *S*-Kommandos unverändert auch in *R*.

Durch die vielen Extrapakete, die über das CRAN (Central *R* Archive Network) verfügbar sind, besitzt *R* Importmodule (entnommen R Development Core Team (2003)) für XML, *Minitab*, *EpiInfo*, *S*, *SAS*, *SPSS*, *STATA* und Binärformate der NASA und UCAR (University Corporation for Atmospheric Research).

Zum Import aus relationalen Datenbanksystemen steht eine Reihe von Treibern zur Verfügung, die Daten aus *mSQL*, *MySQL*, *Oracle* und *Postgres* importieren können. Zusätzlich stehen mit dem *RODBC*-Paket alle Datenquellen zur Verfügung, für die ein ODBC-Treiber existiert.

2.2.1 Das R - Userinterface

R ist zwar unter Windows eine MDI-Anwendung, der Benutzer gibt jedoch nahezu alle Kommandos über die Tastatur ein. Die wenigen Befehle der Menüleiste und dem Kontextmenü dienen nur zum Aufruf von Dateioperationen, zur Installation von zusätzlichen Paketen vom CRAN (Comprehensive R Network Archive) und als Zugang zum Hilfesystem.

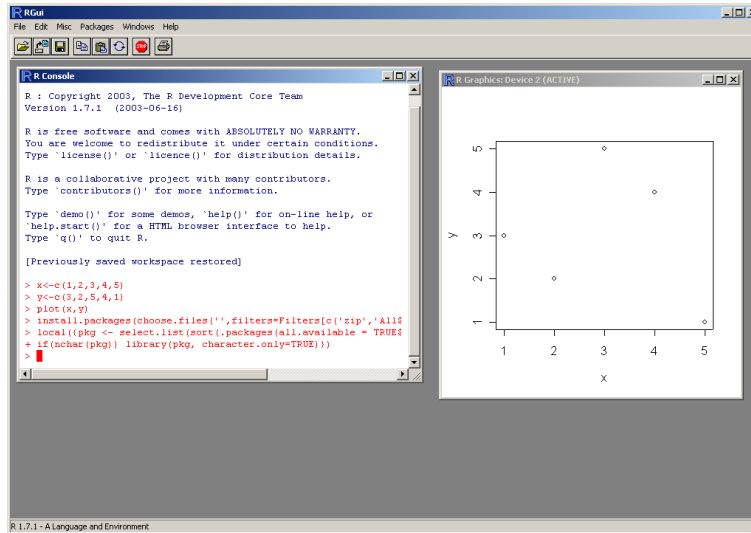


Abbildung 2.6. R: Grafische Benutzeroberfläche (Windows)

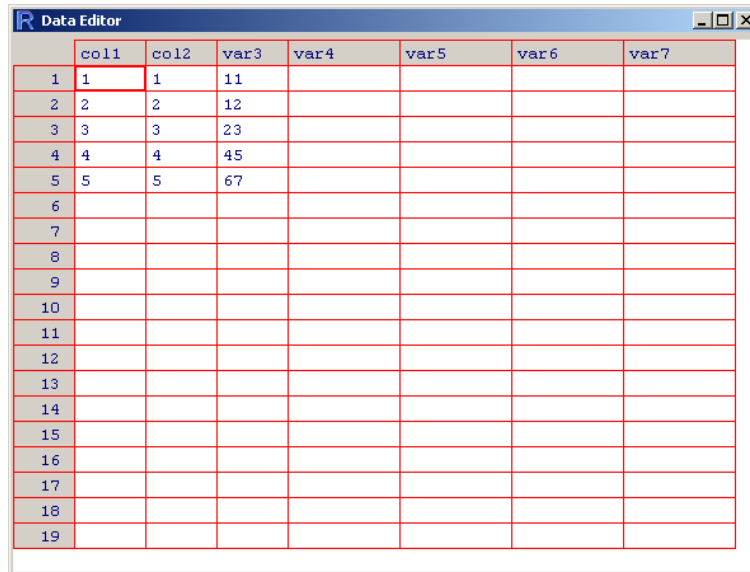
2.2.2 Der integrierte Dateneditor

Unter Windows bringt R einen einfachen Dateneditor mit, mit dessen Hilfe der Benutzer die Matrizen verändern kann. Wird der Editor über den Menüpunkt `Edit`→`Data Editor` gestartet, öffnet sich ein Eingabefeld für den Namen des Objekts, das editiert werden soll.

Der Editor läßt sich auch über die Befehle `fix(Objektname)` und `edit(Objektname)` aufrufen. Der Unterschied zwischen beiden Kommandos liegt darin, daß ein Aufruf von `edit()` nach Beendigung des Editors die Matrix nur zurückgibt (der Nutzer muß diese Rückgabe selbst einer Variablen zuweisen) und `fix` die aufgerufene Matrix mit den neuen Werten überschreibt.

Klickt man auf die Kopfzeile einer Spalte, kann im sich öffnenden Menü Spaltenname und Variablentyp (numeric oder character) verändert werden.

2 Analyse



The screenshot shows the R Data Editor window with a table containing 7 columns and 19 rows. The columns are labeled 'col1', 'col2', 'var3', 'var4', 'var5', 'var6', and 'var7'. The rows are numbered 1 through 19. The data in the first five rows is as follows:

	col1	col2	var3	var4	var5	var6	var7
1	1	1	11				
2	2	2	12				
3	3	3	23				
4	4	4	45				
5	5	5	67				
6							
7							
8							
9							
10							
11							
12							
13							
14							
15							
16							
17							
18							
19							

Abbildung 2.7. R: Matrixeditor (Windows)

2.2.3 Grafik-Export

Erzeugt der Nutzer grafischen Output innerhalb von *R*, erscheint bei einem Klick mit der rechten Maustaste auf die Grafik ein Popup-Menü, das die folgenden Exportfunktionen bereithält:

- Copy as Metafile
- Copy as Bitmap
- Save as Metafile
- Save as Postscript
- Print

Eine sehr interessante Fähigkeit von *R* ist die Umleitung (Redirection) grafischen Outputs in eine Datei.

R stellt dazu eine Reihe von Treibern (Postscript, picTeX , PDF) bereit, die die *R*-Ausgabe in das entsprechende Format umwandeln und als Parameter den Dateinamen entgegennehmen. Weitere Treiber sind als Download über das Internet verfügbar.

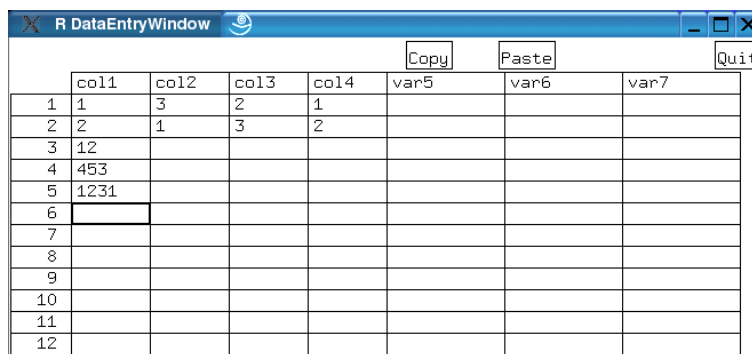
2 Analyse

Eine Beispielsitzung, die eine Postscript-Datei erzeugt, sieht dann so:

```
postscript("c:/beispiel.ps")
x<-rnorm(5)
y<-rnorm(5)
plot(x,y)
dev.off()
```

2.2.4 Die Linux-Version von R

Die Linux-Version von R ist identisch zur Windows-Version, nur die grafische Nutzeroberfläche unterscheidet sich. R ist unter Linux eine reine Konsolenanwendung, alle Ein- und Ausgaben erfolgen hier über die Tastatur.



	col1	col2	col3	col4	var5	var6	var7
1	1	3	2	1			
2	2	1	3	2			
3	12						
4	453						
5	1231						
6							
7							
8							
9							
10							
11							
12							

Abbildung 2.8. R: Editor unter Linux

Der Editor läßt sich ebenso wie unter Windows mit `edit` und `fix` aufrufen und bietet auch die gleiche Funktionalität.

2.2.5 Hilfesystem

R verfügt über ein umfangreiches Hilfesystem. Der Punkt „Help“ in der Menüleiste verfügt über mehrere Links zu den einzelnen Kapiteln des Hilfesystems:

Console: Extra-Fenster mit den wichtigsten Tastenkombinationen der Konsole.

FAQ on R: Die R Frequently Asked Questions im HTML-Format.

FAQ on R for Windows: Windows-spezifische Antworten im HTML-Format.

2 Analyse

R functions (Text): Bildschirm mit Eingabeleiste für Suchbegriff (ein Alias für das `help()`-Kommando, siehe unten).

HTML Help: Verknüpfung zur HTML-Startseite, die mehrere Handbücher und eine Javascript-basierende Suchfunktion enthält.

Manuals:

An Introduction to R

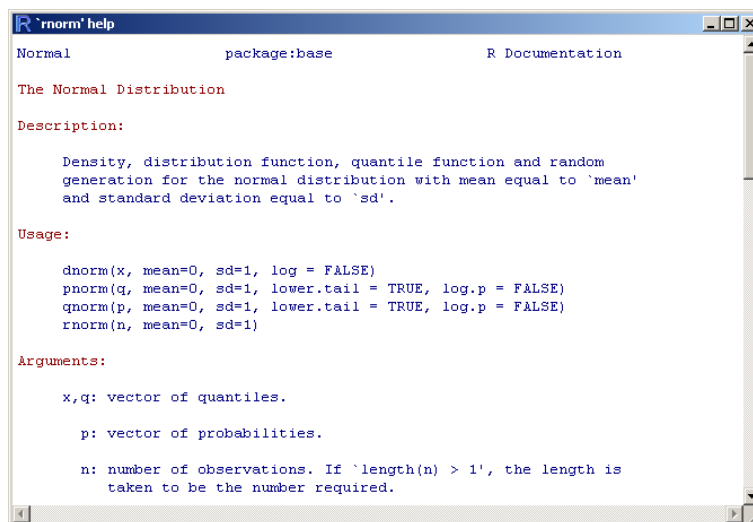
R Reference Manual

R Data Import/Export

R Language Manual

Writing R Extensions

Direkt aus R kann der Nutzer das Hilfesystem über den Befehl `help.start()` starten. Hilfe zu den einzelnen Befehlen erhält man über `help(Befehlsname)`. Das Kommando `help.search(Befehl)` erlaubt zusätzlich auch die unscharfe Suche nach einzelnen Begriffen.



```
R 'rnorm' help
Normal                package:base                R Documentation

The Normal Distribution

Description:

Density, distribution function, quantile function and random
generation for the normal distribution with mean equal to 'mean'
and standard deviation equal to 'sd'.

Usage:

dnorm(x, mean=0, sd=1, log = FALSE)
pnorm(q, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean=0, sd=1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean=0, sd=1)

Arguments:

x,q: vector of quantiles.

p: vector of probabilities.

n: number of observations. If 'length(n) > 1', the length is
taken to be the number required.
```

Abbildung 2.9. R: eingebautes Hilfesystem (Windows)

2.3 Minitab 13

Minitab Inc. (<http://www.minitab.com>) wurde im Jahr 1972 an der Pennsylvania State University gegründet, um die Lehre im Fach Statistik zu unterstützen. Das Kernprodukt, *Minitab*, ist besonders an amerikanischen Hochschulen verbreitet.

Der Bildschirm der aktuellen *Minitab* Version 13 unterteilt sich in drei Fenster: Output-Fenster, Projekt Manager und Dateneditor.

Das Outputfenster ist ständig geöffnet und kann nicht vom Nutzer geschlossen werden.

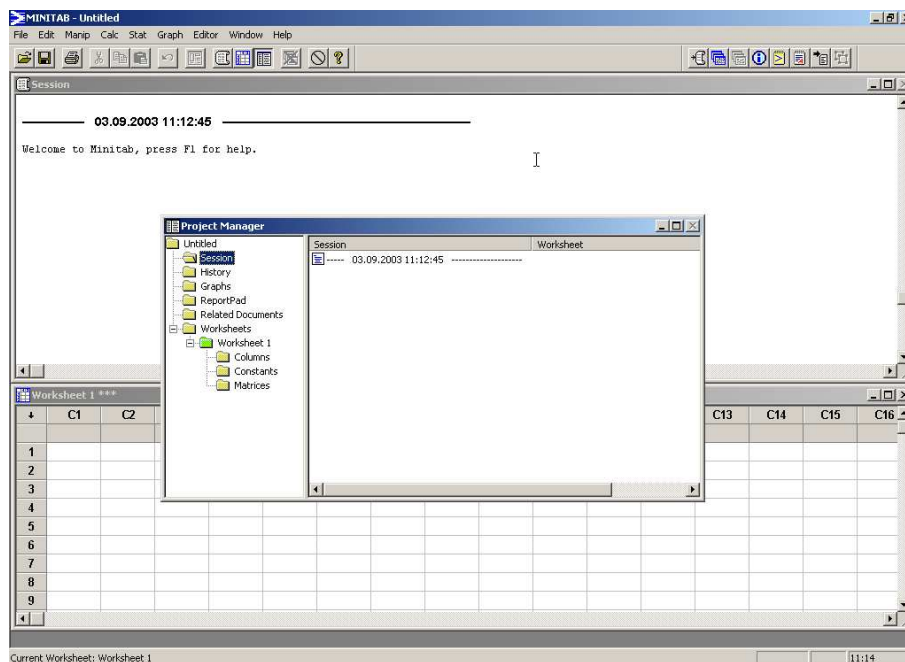


Abbildung 2.10. Minitab: Grafische Benutzeroberfläche

2.3.1 Datenimport

Neben der Importmöglichkeit über ODBC importiert *Minitab* eine Reihe weiterer Formate wie die eigenen Austauschformate, Textdateien, Dbase, Excel, Quattro Pro und Lotus 1-2-3.

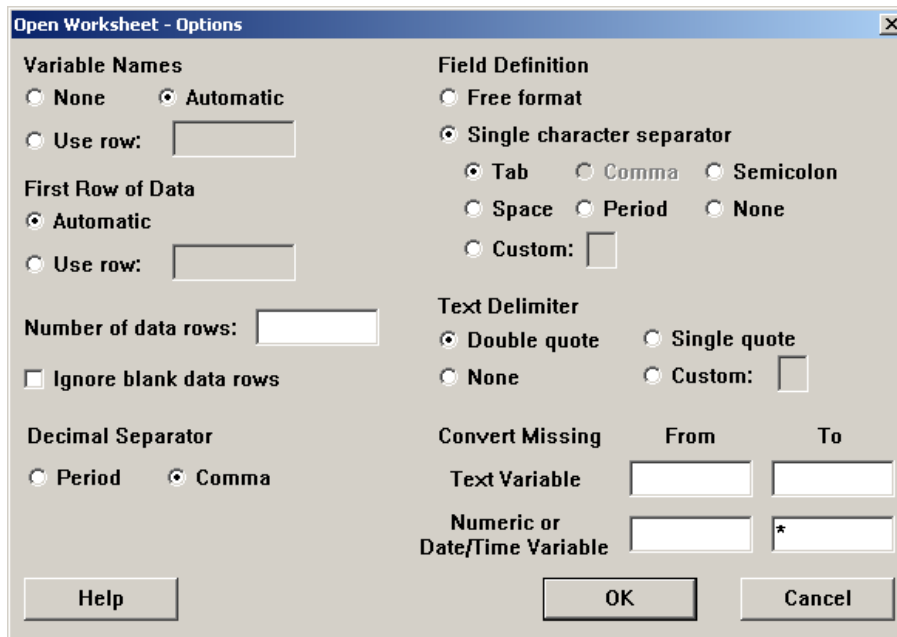


Abbildung 2.11. Minitab: Importoptionen

In den Import-Optionen lassen sich noch verschiedene Angaben zur Struktur der zu importierenden Dateien wie Dezimalseparator, Zeile der Variablennamen, Feldseparator und Zahl der zu importierenden Zeilen machen, bei den Tabellenkalkulationsformaten sind einige dieser Optionen grau und daher nicht zugänglich.

Leider besteht auch keine Möglichkeit, einzelne Tabellenblätter aus einer Excelmappe einzulesen – Minitab importiert die einzelnen Blätter nebeneinander in seinen Tabelleneditor.

2.3.2 Der Minitab-Dateneditor

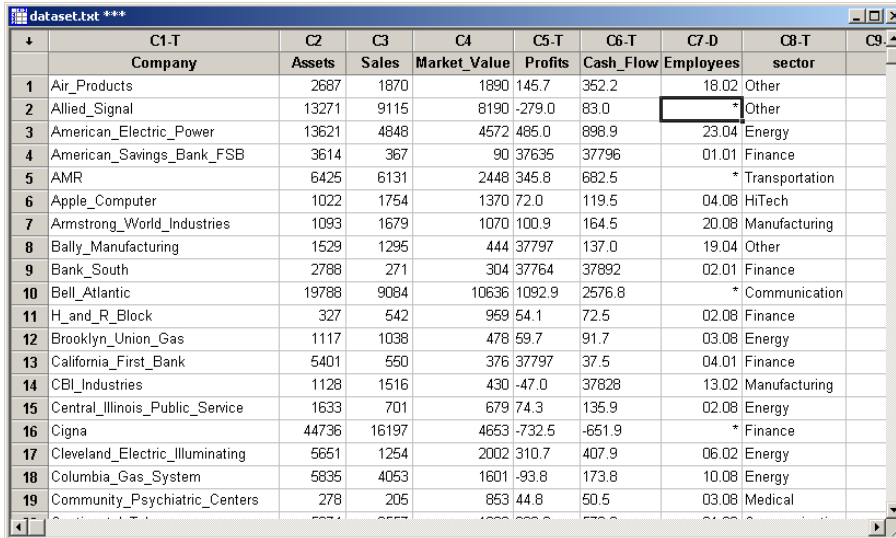
Die linke obere Ecke des Dateneditors enthält einen kleinen Pfeil, über den die Bewegungsrichtung nach einem Druck auf Enter bzw. Return eingestellt werden kann.

Im Gegensatz zu *S-Plus* existiert in *Minitab* keine Möglichkeit, einzelne Fälle mit einem zusätzlichen Namen anzusprechen, einzelne Variablen lassen sich aber zusätzlich zur standardmäßigen Beschriftung mit einem Namen versehen (*Minitab* importiert diese Namen auch aus Excel-Tabellen).

Die Zuteilung von Speicher (Zeilen und Spalten) geschieht dynamisch und ist elegant gelöst: Sobald in einer Spalte ein Wert eingetragen wird, taucht diese

2 Analyse

Spalte als Variable in den Dialogen zur Auswahl von Funktionen auf. Löscht man jedoch alle Werte einer Spalte, „verschwindet“ auch die dazugehörige Variable wieder aus den Dialogen.



	C1-T	C2	C3	C4	C5-T	C6-T	C7-D	C8-T	C9
	Company	Assets	Sales	Market Value	Profits	Cash Flow	Employees	sector	
1	Air_Products	2687	1870	1890	145.7	352.2	18.02	Other	
2	Allied_Signal	13271	9115	8190	-279.0	83.0	*	Other	
3	American_Electric_Power	13621	4848	4572	485.0	898.9	23.04	Energy	
4	American_Savings_Bank_FSB	3614	367	90	37635	37796	01.01	Finance	
5	AMR	6425	6131	2448	345.8	682.5	*	Transportation	
6	Apple_Computer	1022	1754	1370	72.0	119.5	04.08	HiTech	
7	Armstrong_World_Industries	1093	1679	1070	100.9	164.5	20.08	Manufacturing	
8	Bally_Manufacturing	1529	1295	444	37797	137.0	19.04	Other	
9	Bank_South	2788	271	304	37764	37892	02.01	Finance	
10	Bell_Atlantic	19788	9084	10636	1092.9	2576.8	*	Communication	
11	H_and_R_Block	327	542	959	54.1	72.5	02.08	Finance	
12	Brooklyn_Union_Gas	1117	1038	478	59.7	91.7	03.08	Energy	
13	California_First_Bank	5401	550	376	37797	37.5	04.01	Finance	
14	CBI_Industries	1128	1516	430	-47.0	37828	13.02	Manufacturing	
15	Central_Illinois_Public_Service	1633	701	679	74.3	135.9	02.08	Energy	
16	Cigna	44736	16197	4653	-732.5	-651.9	*	Finance	
17	Cleveland_Electric_Illuminating	5651	1254	2002	310.7	407.9	06.02	Energy	
18	Columbia_Gas_System	5835	4053	1601	-93.8	173.8	10.08	Energy	
19	Community_Psychiatric_Centers	278	205	853	44.8	50.5	03.08	Medical	

Abbildung 2.12. Minitab: Dateneditor

Das Kontextmenü enthält neben den Funktionen für die Zwischenablage auch die Befehle `Undo Typing` und `Format Column`. `Format Column` enthält mehrere Möglichkeiten der Zahlendarstellung, `Undo Typing` macht die letzte Eingabe rückgängig, ein versehentliches Überschreiben eines Wertes ist damit nicht möglich. Die Anzahl der `Undo`-Schritte läßt sich nicht festlegen.

Minitab verfügt zudem über einen intelligenten Mechanismus beim Einfügen von Zeilen und Spalten. In Abhängigkeit von der Cursor-Position kann der Anwender Spalten und Zeilen einfügen. Steht der Cursor bei leerer Tabelle in der ersten Spalte oder Zeile, lassen sich keine neuen Variablen hinzufügen. Befindet sich hingegen in dieser Zelle ein Wert, läßt sich eine weitere Zeile/Spalte einfügen.

Eine ebenfalls gelungene Funktion von *Minitab* sorgt für die automatische Zuordnung der Variablen. Die erste Zeile der Tabelle enthält den Typ der Variablen, so steht T für Text und D für Datumswerte. Steht kein Zeichen hinter dem Spaltenzähler, handelt es sich um eine numerische Variable.

Startet der Benutzer in einer Spalte mit der Eingabe von Text, wird der Spaltentyp automatisch auf den Wert "Text" gesetzt, bei der Eingabe von Zahlen automatisch auf einen numerischen Wert. Der Typ dieser Spalte läßt sich anschließend nur noch durch Konvertierung der Werte ändern.

2 Analyse

Ungewöhnlich ist *Minitab* hingegen beim Löschen von Daten aus Zellen. Löscht der Benutzer den Inhalt einer Zelle, bleibt diese Zelle nicht leer oder wird mit einem Symbol für „Missing Values“ gefüllt, *Minitab* schiebt stattdessen den Wert der nächsten Zeile hoch und vermischt so die Daten der einzelnen Fälle.

Der Nutzer kann zwar über das Kontextmenü eine Zelle einfügen, aber bei einer Datenstruktur mit sehr ähnlichen Werten kann dies leicht zur Verwirrung führen.

Gut gefällt mir die Möglichkeit in *Minitab*, ähnlich wie in *Stata*, einzelne Spalten auszublenden. Verschieben lassen sich die Spalten nicht über einen Mausbefehl, dazu ist ein Umweg über das Kontextmenü notwendig.

2.3.3 Grafikexport

Minitab unterstützt mit JPEG, PNG, TIF, BMP und dem eigenen MGF (*Minitab Graphics File*) fünf verschiedene Grafikformate, jedoch leider kein Vektorformat wie PS, SVG oder EPS. Für Schwarzweiss-Grafiken stehen eigene Filter bereit.

Der Sinn einer Funktion von *Minitab* erschließt sich nicht: Über das Kontextmenü lassen sich die einzelnen angezeigten Grafiken entsperren, durch diese Funktion kann der Nutzer einzelne Datenpunkte mit der Maus verschieben.

2.3.4 Hilfesystem

Praktisch gelöst ist hingegen die Verbindung zum Hilfesystem, das als HLP-Datei implementiert wurde. In vielen Dialogen findet sich ein Hilfebutton, der den Nutzer zum entsprechenden Abschnitt in der Hilfedatei bringt.

2.4 SPSS 11

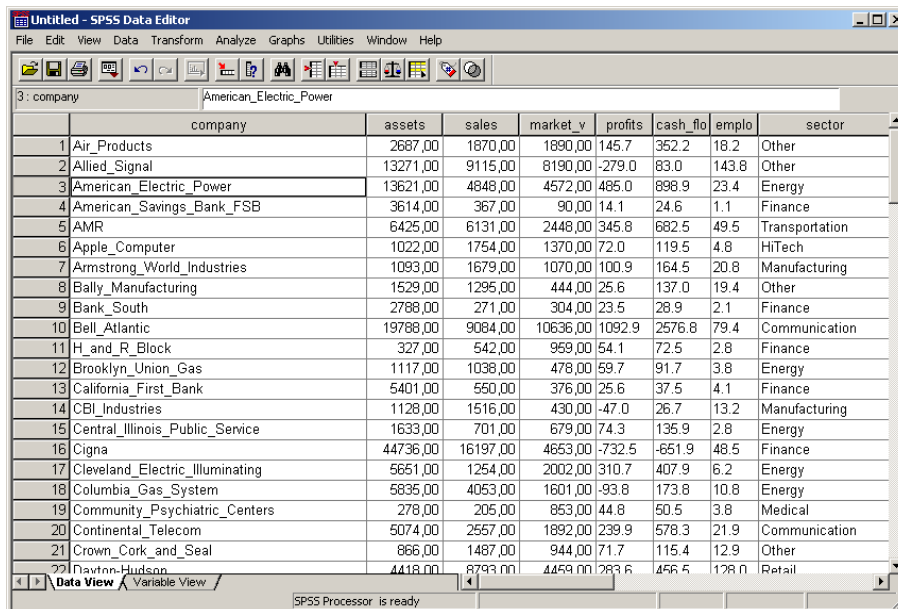
SPSS ist die Abkürzung für *Statistical Package for the Social Sciences* und bezeichnet das Kernprodukt der gleichnamigen US-amerikanischen Firma, die 1968 gegründet wurde, um Software zur Unterstützung von betrieblichen Entscheidungsprozessen zu programmieren.

SPSS 11 ist im Gegensatz zu den anderen untersuchten Paketen keine MDI-Anwendung. Die grafische Benutzeroberfläche besteht aus dem Tabelleneditor, der zur Eingabe der Daten dient und den Menüleisten, über die auf die Auswertungsmethoden zurückgegriffen wird.

2 Analyse

Das Programmpaket ist wahrscheinlich auch deshalb so erfolgreich, weil es den Nutzer vor der Eingabe von Programmcode bewahrt. Es besteht zwar auch die Möglichkeit, Skripte in einer Microsoft Visual Basic kompatiblen Sprache zu verfassen (Sax Basic Language), ein Großteil der Benutzer wird diese Funktionen jedoch nie benutzen, da die Mehrzahl der Nutzer aus den Geisteswissenschaften stammt, die schnell und bequem zu ihren Ergebnissen kommen wollen und denen die innerhalb des Programms gebotenen Möglichkeiten ausreichen.

Zusätzlich zu dem mausbasierten Modus bietet SPSS einen automatischen Modus zur unbeaufsichtigten Datenanalyse an. Er erlaubt es, einmal definierte Schrittfolgen von Analysen zu speichern und als Batch-Vorgang laufen zu lassen, was besonders für die routinemäßige Auswertung von Daten hilfreich ist.



The screenshot shows the SPSS Data Editor window with a data table. The table has the following columns: company, assets, sales, market_v, profits, cash_flow, emplo, and sector. The data is as follows:

	company	assets	sales	market_v	profits	cash_flow	emplo	sector
1	Air_Products	2687,00	1870,00	1890,00	145.7	352.2	18.2	Other
2	Allied_Signal	13271,00	9115,00	8190,00	-279.0	83.0	143.8	Other
3	American_Electric_Power	13621,00	4848,00	4572,00	485.0	898.9	23.4	Energy
4	American_Savings_Bank_FSB	3614,00	367,00	90,00	14.1	24.6	1.1	Finance
5	AMR	6425,00	6131,00	2448,00	345.8	682.5	49.5	Transportation
6	Apple_Computer	1022,00	1754,00	1370,00	72.0	119.5	4.8	HiTech
7	Armstrong_World_Industries	1093,00	1679,00	1070,00	100.9	164.5	20.8	Manufacturing
8	Bally_Manufacturing	1529,00	1295,00	444,00	25.6	137.0	19.4	Other
9	Bank_South	2788,00	271,00	304,00	23.5	28.9	2.1	Finance
10	Bell_Atlantic	19788,00	9084,00	10636,00	1092.9	2576.8	79.4	Communication
11	H_and_R_Block	327,00	542,00	959,00	54.1	72.5	2.8	Finance
12	Brooklyn_Union_Gas	1117,00	1038,00	478,00	59.7	91.7	3.8	Energy
13	California_First_Bank	5401,00	550,00	376,00	25.6	37.5	4.1	Finance
14	CB_I_Industries	1128,00	1516,00	430,00	-47.0	26.7	13.2	Manufacturing
15	Central_Illinois_Public_Service	1633,00	701,00	679,00	74.3	135.9	2.8	Energy
16	Cigna	44736,00	16197,00	4653,00	-732.5	-651.9	48.5	Finance
17	Cleveland_Electric_Illuminating	5651,00	1254,00	2002,00	310.7	407.9	6.2	Energy
18	Columbia_Gas_System	5835,00	4053,00	1601,00	-93.8	173.8	10.8	Energy
19	Community_Psychiatric_Centers	278,00	205,00	853,00	44.8	50.5	3.8	Medical
20	Continental_Telecom	5074,00	2557,00	1892,00	239.9	578.3	21.9	Communication
21	Crown_Cork_and_Seal	866,00	1487,00	944,00	71.7	115.4	12.9	Other
22	Davita-Hudson	4418,00	8793,00	4459,00	383.6	456.5	128.0	Retail

Abbildung 2.13. SPSS: Benutzeroberfläche

2.4.1 Dateneditor

Der Editor, der bei SPSS zur Eingabe der Daten benutzt wird, bietet eine Reihe von Funktionen, die nach Meinung des Autors in ähnlicher Form auch in Yxilon Eingang finden werden.

Der Editor besteht aus zwei übereinander liegenden Fenstern, *Data View* und *Variable View*, die über die entsprechenden Reiter am unteren Rand zugänglich sind.

2 Analyse

Der *Data View*-Bildschirm enthält die Tabelle, in der die Daten selbst eingegeben werden, *Variable View* beinhaltet die Metadaten der einzelnen Variablen (er ähnelt dem Variableneditor von *Statistica*). Zu diesen Metadaten gehören:

- Name (maximal acht Zeichen)
- Typ (Numeric, Date, Dollar/Custom Currency und String)
- Anzahl der Stellen
- Dezimalzeichen
- Label (genauere Bezeichnung der Variablen)
- Werte (um einzelnen Werte Labels zuzuordnen)
- Missing (zum Umcodieren fehlender Werte)
- Breite der Spalte
- Ausrichtung
- Skala (nominal, ordinal, metrisch)

Diese Metadaten generiert SPSS bei der Eingabe von Werte in den *Data View*-Bildschirm automatisch, so wird bei Eingabe einer Zeichenkette in der ersten Zeile der Variablen der Typ automatisch auf „String“ gesetzt und die Spaltenbreite auf die Länge des eingegebenen Strings gesetzt.

Setzt der User den Mauscursor in die Mitte einer leeren Tabelle und fügt einen Wert ein, füllt SPSS das entstandene Rechteck beginnend von der linken, oberen Ecke mit ',', der SPSS-eigenen Codierung für fehlende Werte. Gleichzeitig werden die gefüllten Spalten auch in der *Variable View* als numerische Variablen initialisiert.

Über das Ziehen von Spalten mit der Maus verschiebt SPSS die Variablen, dies kann in der *Data View* oder *Variable View* geschehen.

Der Editor besitzt zwei verschiedene Kontextmenüs, die über die rechte Maustaste aufgerufen werden. Mit dem ersten Menü, das nur zugänglich ist, wenn sich der Mauscursor über der ersten Zeile mit den Variablennamen befindet, können komplette Spalten ausgeschnitten, kopiert oder geleert werden. Außerdem besteht hier die Möglichkeit, Spalten auf- oder absteigend zu sortieren, Variablen einzufügen oder Spalten zu fixieren (nützlich für die Eingabe multi-variater Datensätze).

Im zweiten Menü, das angezeigt, wenn der Nutzer innerhalb der Tabelle die rechte Maustaste betätigt, können die Werte einzelner Zellen kopiert und ausgeschnitten werden, außerdem kann hier die Schriftart und Schriftgröße eingestellt werden.

2.4.2 Datenimport

1. Möglichkeit, ein bereits abgespeichertes Formatschema wieder einzuladen.
2. Eingabe des Typs (zeichengetrennt/feste Spaltenbreite) und ob Variablenamen in der Datei stehen.
3. Eingabe der Zeile des ersten Datensatzes, der Repräsentation der Fälle und Anzahl der zu importierenden Fälle.
4. Eingabe von Trennzeichen und Texttrenner.
5. Definition der einzelnen Variablentypen (don't import, numeric, string, date/time, Dollar).
6. Vorschau des zu importierenden Datensatzes mit der Möglichkeit, das Formatschema abzuspeichern.

SPSS verfügt auch über exzellente Importfunktionen für ODBC-Daten. Nach Auswahl der Datenquelle kann der Nutzer die einzelnen Spalten markieren, die importiert werden sollen.

Im Bildschirmmenü „limit retrieved cases“ lassen sich Zufallsstichproben der Fälle auswählen und der Import der Felder über SQL-Funktionen genauer steuern, dabei unterstützt SPSS auch binäre Operatoren.

Mit diesen Funktionen ist es beispielsweise möglich, alle Datensätze einer Quelle zu importieren, bei deren Namensfeld der zweite Buchstabe ein 'A' ist. Die Art und Weise, mit der SPSS den Nutzer durch die Erstellung der Abfragen führt, hat mir von allen betrachteten Softwarepaketen am besten gefallen.

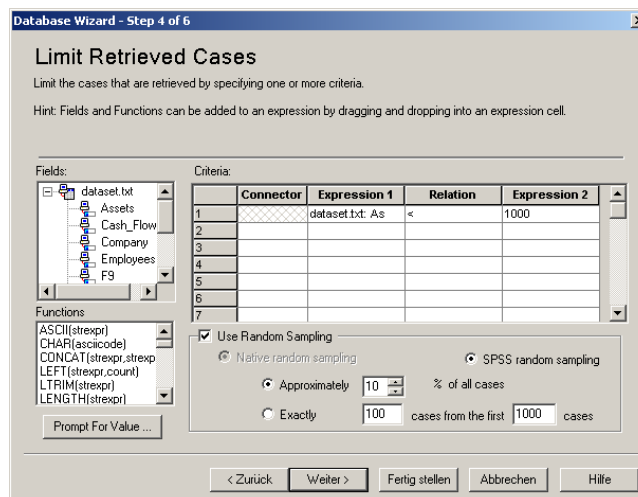


Abbildung 2.15. SPSS: ODBC-Importoptionen

2 Analyse

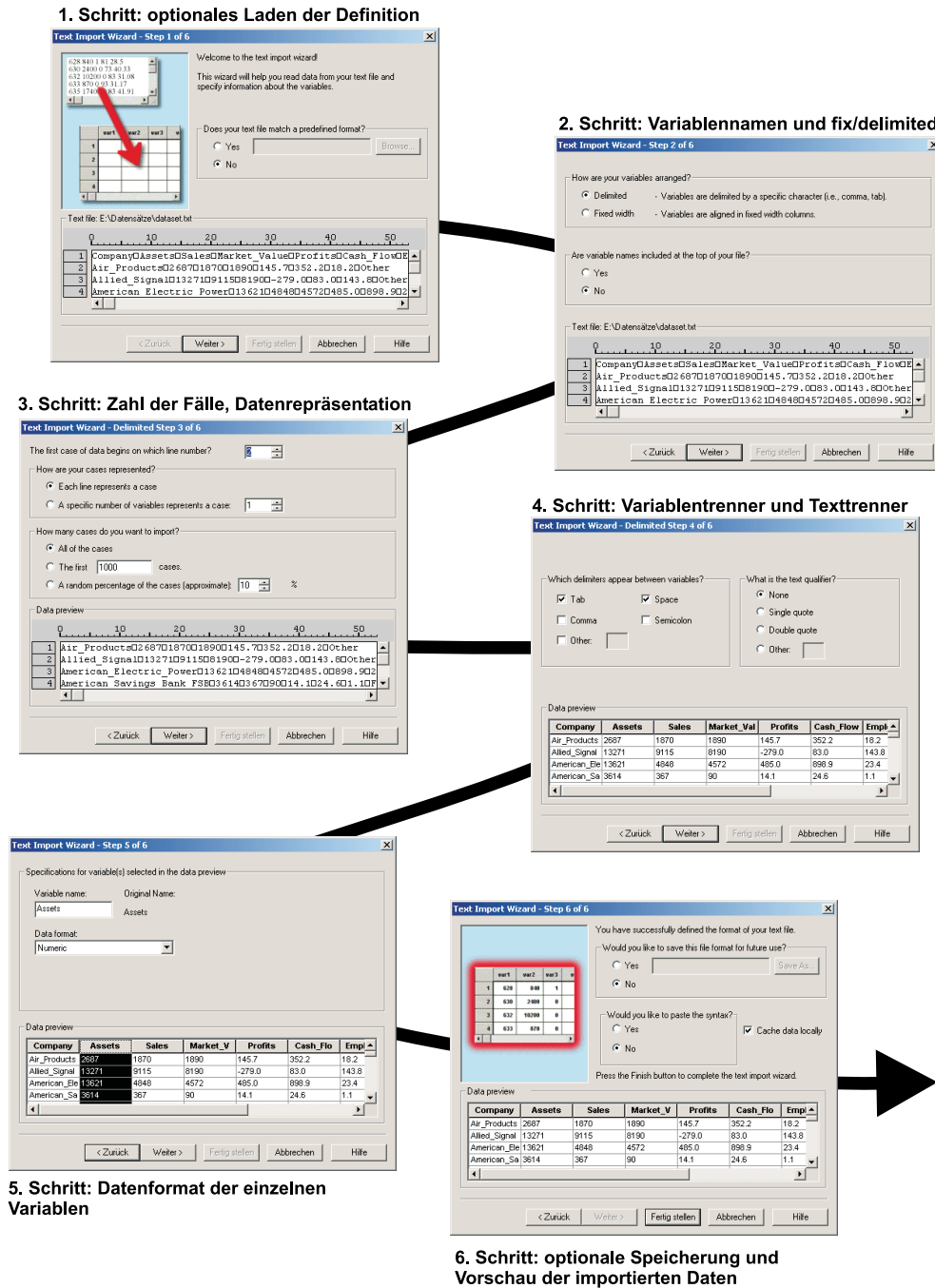


Abbildung 2.14. SPSS: Import von Textdateien

2.4.3 Export von Daten und Grafiken

SPSS besitzt kein eigenes Ausgabefenster innerhalb des Programms, alle erzeugten Grafiken sowie die Textausgabe gelangen in den SPSS Viewer. Dort kann der Nutzer dann die ausgegebenen Grafiken und Tabellen formatieren und im SPSS-eigenen SPO-Format speichern (dies sogar passwortgeschützt). Der Output läßt sich zusätzlich in verschiedene Exportformate überführen, wobei SPSS zwischen dem Export des gesamten Dokuments, dem Export der Grafiken und dem Export des Textes unterscheidet.

Komplette Dokumente und Text Ausgaben kann der Benutzer im XML-, HTML- oder Textformat abspeichern, für die Ausgabe von Grafiken stehen noch weitere Formate zur Verfügung, die jeweils über eigene, recht detaillierte Exportoptionen verfügen. SPSS verfügt auch über Exportfilter für EPS.

2.4.4 Hilfesystem

Although you can change the formatting of a table after it has been generated, it may be more economical to change the default TableLook so that your tables are created according to your needs, without the overhead of changing formatting every time.

Changing the Default Table Formatting

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid	Unmarried	3224	50.4	50.4	50.4
	Married	3176	49.6	49.6	100.0
	Total	6400	100.0	100.0	

Abbildung 2.16. SPSS: HTML Tutorial

2 Analyse

Das SPSS-Hilfesystem (erreichbar über den Menüpunkt *Help*) untergliedert sich in verschiedene Teile:

Topics: Die eigentliche Hilfe wird im Microsoft HLP-Format mitgeliefert, über die drei Reiter *Inhalt*, *Index* und *Suchen* kann der Anwender nach der gesuchten Funktion suchen.

Tutorial: *Tutorial* beinhaltet einen komplett in HTML geschriebenen Lehrgang, der mit einer dem HLP-System ähnlichen Oberfläche den Nutzer in die Arbeit mit SPSS einführt.

Syntax Guide: Dieses Menü beinhaltet Links zu einer Reihe von PDF-Dateien, die die SAX-Syntax für die Funktionen von SPSS enthalten.

Statistics Coach: Der *Statistics Coach* ist ein umfangreiches Hilfewerkzeug, bestehend aus Datei im Microsoft CHM-Format. Es erlaubt dem Anwender, jeweils aus einer Liste mit verschiedenen Vorschlägen zu wählen. Im ersten Dialog (siehe Bild 2.17) wird der Nutzer nach dem Ziel seiner Arbeit gefragt. Wenn dieses Ziel (Bsp: „Compare groups for significant differences“) feststeht, wird in den folgenden Bildschirmdialogen eingegrenzt, welche Skalen die Daten haben, wie viele Gruppen unterschieden werden, etc. Am Ende des Dialogs wird der der Fragestellung entsprechende Dialog-Bildschirm geöffnet und der Nutzer kann die Variablen zuordnen. Wurden noch keine Werte eingegeben, gibt SPSS eine Fehlermeldung aus.

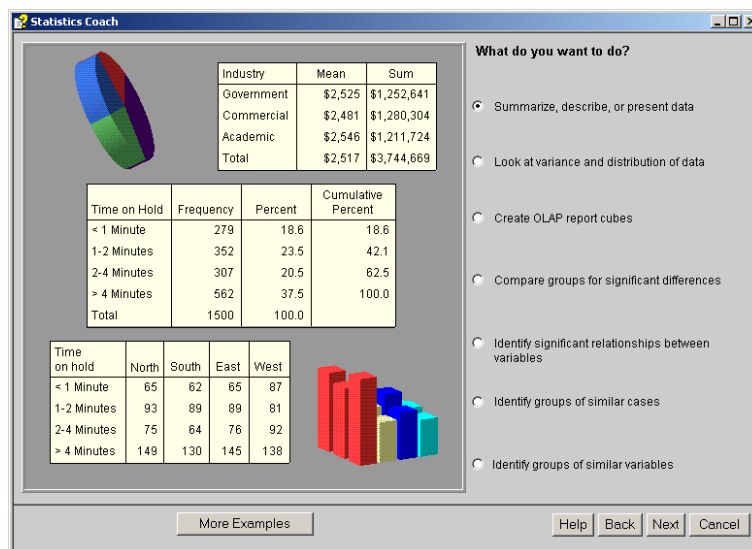


Abbildung 2.17. SPSS: Statistics Coach

2.5 Gauss 5

Das Softwarepaket *Gauss* der amerikanischen Firma Aptech (www.aptech.com) ist in Umfang und Bedienung vergleichbar mit XploRe. Es bietet eine konsolenorientierte Oberfläche, die über die Eingabe von Quelltexten in der Gaussia-eigenen Syntax benutzt wird. Die wenigen Menüs, die *Gauss* für den Nutzer bereitstellt, dienen überwiegend dem Zugriff auf Standardfunktionen wie Datei- und Druckoperationen sowie dem Zugriff auf die verschiedenen Fenster und das Hilfesystem.

Interessant ist die Möglichkeit, fertige Skripte zu kompilieren. Dadurch wird eine binäre Datei erzeugt, die erheblich schneller verarbeitet werden kann als interpretierte Skripte, da die Analyse des Quelltextes entfällt.

Außerdem bietet *Gauss* die Möglichkeit, geschriebene Module zu debuggen. Durch einen Mausklick lassen sich Haltepunkte setzen, zusammen mit einem Watch (einem „Beobachter“) lassen sich so Änderungen an Variablen während der Programmausführung verfolgen, was für die Fehlersuche sehr hilfreich ist.

Ebenso wie XploRe und R ist es nicht nur auf die Windows-Welt beschränkt. So werden neben Linux noch fünf weitere UNIX-Derivate unterstützt.

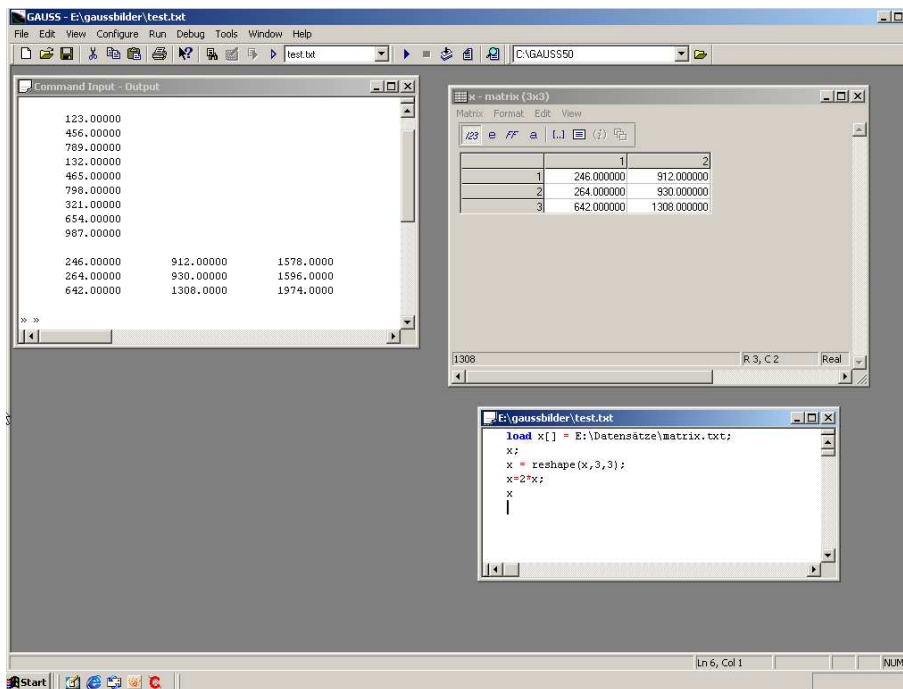


Abbildung 2.18. Gauss: grafische Oberfläche

2.5.1 Import und Export

Gauss besitzt keine Möglichkeit, Datensätze über Menüfunktionen zu laden. Daten importiert der Nutzer mit Hilfe verschiedener Befehle, so lesen die beiden folgenden Befehle eine 3×3 Matrix aus einer Datei in einen Vektor und formen diesen Vektor wieder zur ursprünglichen Matrix um.

```
load x[] = E:\Datensätze\matrix.txt;  
x = reshape(x,3,3);
```

2.5.2 Dateneditor

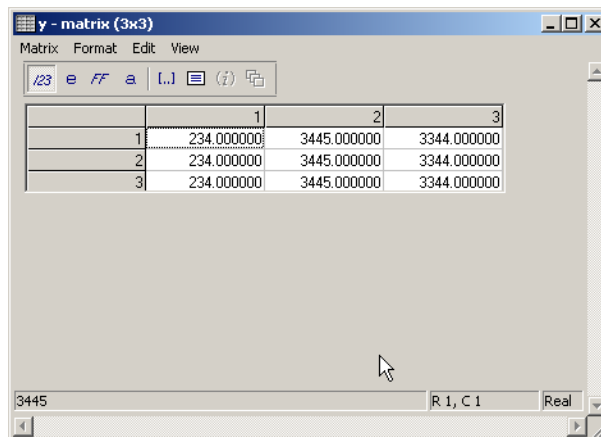


Abbildung 2.19. Gauss: Matrix Editor

Im Vergleich zu den anderen betrachteten Paketen fällt der Matrixeditor von *Gauss 5* bescheiden aus. Startet man den Matrixeditor über den Menüpunkt `Tools` → `Matrix Editor`, erwartet *Gauss* die Eingabe des Namens der Matrix, die man editieren möchte. Die Matrix muß im Speicher schon vorhanden sein, ansonsten wird *Gauss* eine Fehlermeldung ausgegeben.

Der Nutzer kann die bestehenden Objekte im Editor auch nicht in der Größe ändern, es lassen sich keine Zeilen oder Spalten hinzufügen bzw. löschen.

Das Menü des Editors enthält die vier Unterpunkte `Matrix`, `Format`, `Edit` und `View`, die die folgenden Befehle bereitstellen:

Matrix: Load, reload, auto-reload und save zum Laden und Speichern von Matrizen.

Format: Hier kann der Datentyp jeder Spalte der Matrix eingestellt werden, verfügbar sind Decimal, Scientific, Hexadecimal und Char; zusätzlich läßt sich die Präzision einstellen.

2 Analyse

Edit: Mit `Clear all` läßt sich der Inhalt der Matrix löschen, über `Preferences` kann der Benutzer die Höhe und Breite der Spalten, Datenformat, Präzision und Schrittrichtung nach `Return/Enter` festlegen.

View: Das Untermenü `View` enthält Funktionen zur Anzeigen von Dimension, `real parts` und `imaginary parts`. Über das Häkchen von `stay on Top` kann gesteuert werden, ob das Editorfenster stets im Vordergrund bleibt.

Bei der Arbeit mit dem Dateneditor fiel auf, daß die Konsistenz der editierten Daten in *Gauss* möglicherweise nicht in jedem Fall konsistent ist.

Der Benutzer kann mehrere Instanzen des Matrixeditors starten und in diese Instanzen ein und dieselbe Matrix laden.

Wird jetzt in einer Matrix ein Wert geändert und die geänderte Matrix gespeichert, weiß der zweite Editor von dieser Änderung nichts. Wird der zweite Editor jetzt ebenfalls gespeichert, gehen die Änderungen der ersten Speicherung verloren.

Gauss bietet zwar im Kontextmenü die Möglichkeit des Update bzw. Auto-Update für den Inhalt des Editors an, dies funktionierte aber bei der von mir genutzten Version nicht.

2.5.3 Grafikexport

Gauss bietet vier Formate für den Grafikexport an:

- Windows Bitmap
- HPGL
- Enhanced Postscript
- Enhanced Metafile

2.5.4 Hilfesystem

Ebenso wie in den meisten anderen Softwarepaketen besteht das Hilfesystem in *Gauss* aus einer Datei im *Windows Help Format*.

Zusätzlich bietet der Hersteller Aptech eine Reihe von PDF-Dateien zum Download an:

- GAUSS 5.0 Quick-Start Guide
- GAUSS 5.0 User Guide

2 Analyse

- GAUSS 5.0 Language Reference
- GAUSS 5.0 Engine Manual
- GAUSS Language Basics

2.6 Statistica 6

Die Firma StatSoft (<http://www.statsoftinc.com>), die *Statistica* entwickelt und vermarktet, wurde im Jahr 1984 gegründet. Das Ziel war, eine leicht bedienbare Software zu schaffen, die die Datenanalyse in verschiedenen Bereichen der Wissenschaft unterstützt.

Am Beginn jeder Arbeit mit *Statistica*, das ebenso wie *SPSS* über Visual Basic programmiert wird, steht der Import beziehungsweise die Erstellung der Daten. Bei der Erstellung eines neuen Datensatzes kann der Nutzer die Anzahl der Fälle und Variablen bestimmen und entscheiden, ob der Datensatz in eine Arbeitsmappe eingefügt wird.

2.6.1 Datenerstellung

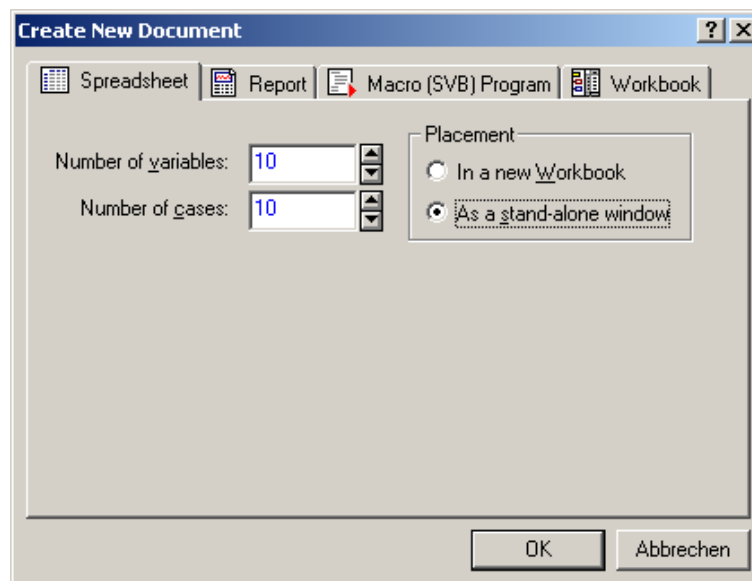


Abbildung 2.20. Statistica: Erstellen neuer Dokumente

2 Analyse

2.6.2 Datenimport

Sehr leicht ist der Import von ODBC-Daten. Nachdem unter Datenquellen (ODBC) in den System-Einstellungen des Rechners eine neue System-DSN erstellt wurde, läßt sich im *Query Model* von *Statistica* eine neue Verbindung zur Datenquelle einrichten.

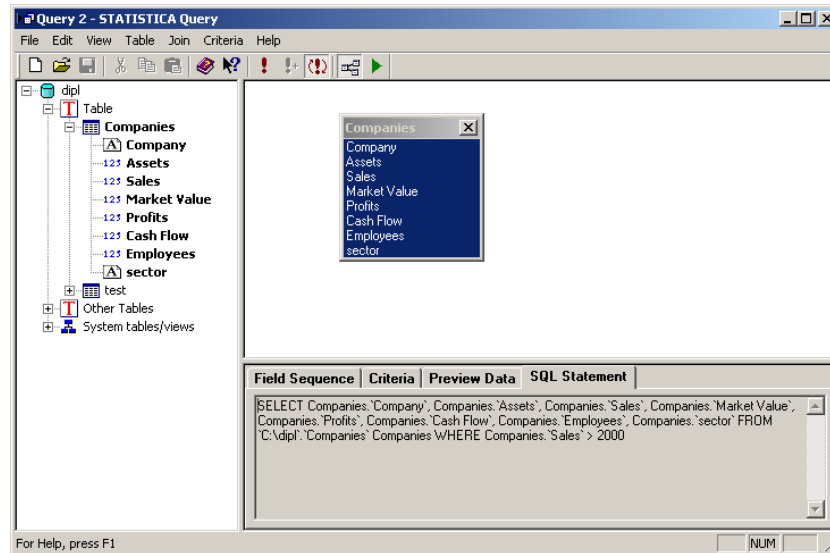


Abbildung 2.21. Statistica: ODBC-Importdialog

Unter Add Criteria lassen sich logische Ausdrücke zur Eingrenzung der Menge der Importdatensätze erstellen, Preview Data zeigt eine Voransicht der zu importierenden Daten und SQL Statement die Repräsentation der Abfrage in SQL.

Abschließend kann der Nutzer entscheiden, ob die Daten in eine neue oder bereits vorhandene Tabelle eingefügt werden.

2 Analyse

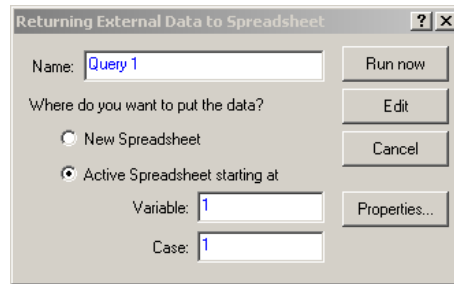


Abbildung 2.22. Statistica: ODBC-Importauswahl

Zusätzlich zum Import via ODBC bietet *Statistica 6* folgende Importfilter:

RTF	dBase
HTML	Excel
Lotus 1-2-3	SPSS
ASCII-Text	diverse Statistica

Tabelle 2.2. Statistica: Importformate

Als sehr komfortabel gestaltet sich der Import von Excel-Tabellen.

Wird eine Excel-Datei geöffnet, kann der Nutzer wählen, ob einzelne Tabellenblätter oder die komplette Datei mit allen Tabellenblättern importiert werden sollen.

Wird die komplette Datei importiert, läßt sich festlegen, ob die Formatierung der Zellen sowie der Spalten- und Zeilenbezeichnung aus der ersten Spalte bzw. Zeile übernommen werden soll.

2 Analyse

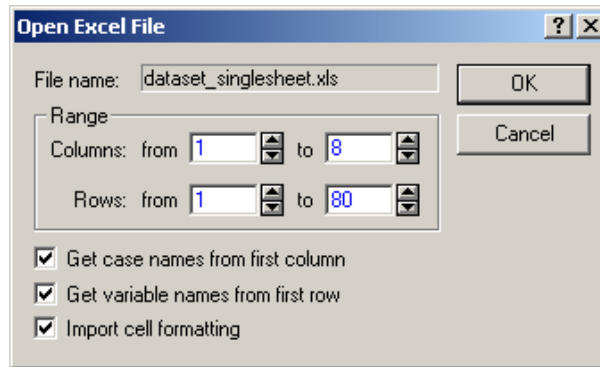


Abbildung 2.23. Statistica: Import von Excel-Tabellen

Statistica erkennt durch die Anzahl der vollen Zeilen und Spalten automatisch die Größe der zu importierenden Tabelle und bietet diese Werte als Voreinstellung an.

Textdateien lassen sich auf drei verschiedene Arten importieren:

auto: deckt den größten Teil der möglichen Textformate automatisch ab

free: erlaubt die Erstellung individueller Formate, die nicht von *auto* oder *fixed* erfaßt werden

fixed: importiert Dateien mit fester Spaltenbreite

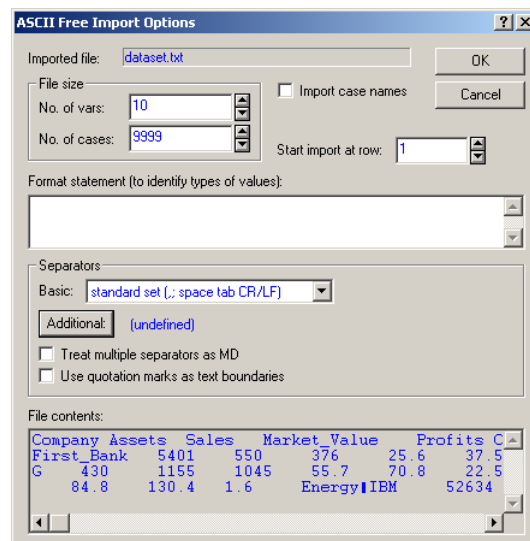


Abbildung 2.24. Statistica: ASCII Text Import „Free“

2 Analyse

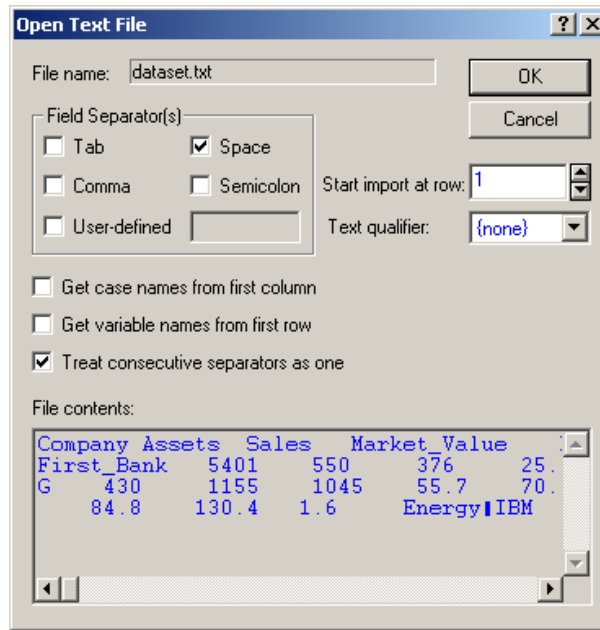


Abbildung 2.25. Statistica: Text Import „Auto“

2.6.3 Der Tabelleneditor in Statistica 6

Nach dem Import wird der Datensatz wie in Bild 2.26 dargestellt, die Anzahl der Fälle und Variablen wird in der Statuszeile angezeigt.

Data: dataset* (7v by 79c)

	1	2	3	4	5	6	7
	Assets	Sales	Market Value	Profits	Cash Flow	Employees	sector
Air_Products	2687	1870	1890	145,7	352,2	18,2	Other
IBM	52634	50056	95697	6555	9874	400,2	HiTech
Cigna	44736	16197	4653	-732,5	-651,9	48,5	Finance
American_Savings_B:	3614	367	90	14,1	24,6	1,1	Finance
AMR	6425	6131	2448	345,8	682,5	49,5	Transportation
Apple_Computer	1022	1754	1370	72	119,5	4,8	HiTech
Armstrong_World_Ind	1093	1679	1070	100,9	164,5	20,8	Manufacturing
Bally_Manufacturing	1529	1295	444	25,6	137	19,4	Other
Bank_South	2788	271	304	23,5	28,9	2,1	Finance
Mellon_Bank	33406	3222	1413	201,7	246,7	15,8	Finance
H_and_R_Block	327	542	959	54,1	72,5	2,8	Finance
Brooklyn_Union_Gas	1117	1038	478	59,7	91,7	3,8	Energy
California_First_Bank	5401	550	376	25,6	37,5	4,1	Finance
CBI_Industries	1128	1516	430	-47	26,7	13,2	Manufacturing
Central_Illinois_Public	1633	701	679	74,3	135,9	2,8	Energy
General_Electric	26432	28285	33172	2336	3562	304	HiTech
Cleveland_Electric_Ill	5651	1254	2002	310,7	407,9	6,2	Energy
Columbia_Gas_Syste	5835	4053	1601	93,8	173,8	10,8	Energy

Abbildung 2.26. Statistica: Importierter Companies-Datensatz

2 Analyse

Ein Doppelklick auf die Spaltenbezeichnung öffnet das umfangreiche Formatierungsmenü. Hier kann der Nutzer das Datenformat und Formatierung der Spalte festlegen, der Umfang des Menüs erinnert dabei sehr an Excel und andere Tabellenkalkulationen.

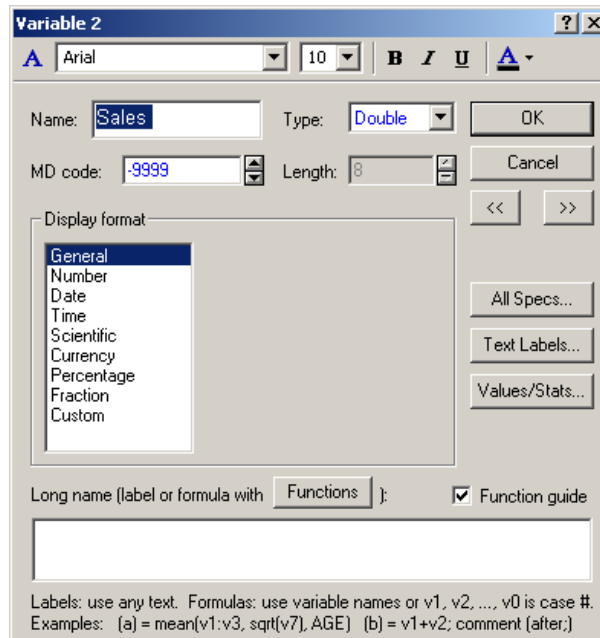


Abbildung 2.27. Statistica: Variablen-Editor

Für die Werte im Editor lassen sich sogar Schriftart und -schnitt sowie die Fontgröße vorgeben. Im Eingabefeld am unteren Rand des Menüs kann der Anwender die Variablen nicht nur mit zusätzlichem Text beschreiben, hier besteht auch die Möglichkeit, den Wert einer Variablen durch eine mathematische Funktion aus anderen Variablen zu errechnen.

Das Formatierungsmenü enthält noch drei weitere Untermenüs:

All Specs: öffnet eine Tabelle, mit der der Name, Typ und MD Code (Missing Data Code) aller Variablen der Tabelle geändert werden kann.

Text Labels: bietet die Möglichkeit, den einzelnen Datenpunkten erklärenden Text hinzuzufügen.

Values/Stats: zeigt alle Werte der Variablen zusammen mit Typ, Anzahl der Datenpunkte, Mittelwert und Standardabweichung.

2 Analyse

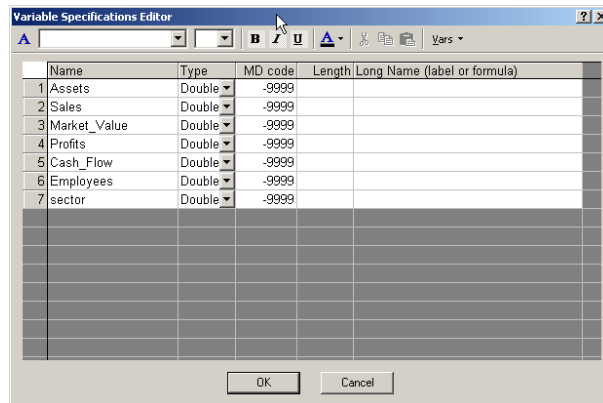


Abbildung 2.28. Statistica: Tabelleneditor für Variablentypen

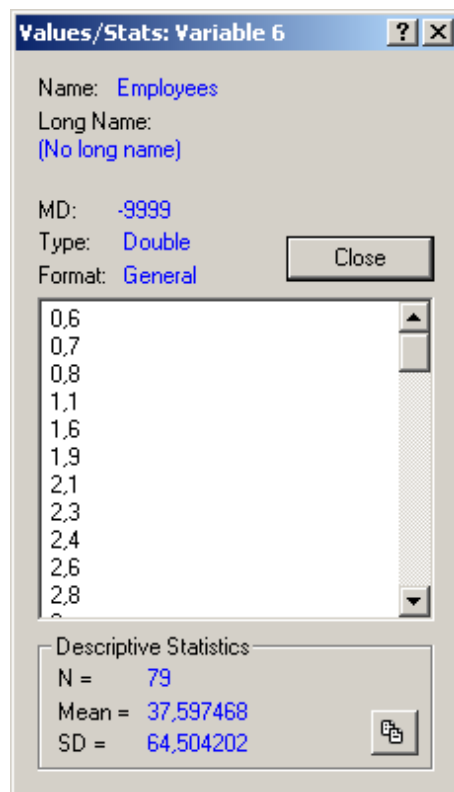


Abbildung 2.29. Statistica: Values/Stats

Das Kontextmenü der einzelnen Tabellen in *Statistica 6*, das man über die rechte Maustaste aufruft, bietet zahlreiche Möglichkeiten, die Daten in der Tabelle sowohl zu ändern als auch auszuwerten.

2 Analyse

Zusätzlich zu den standardmäßig vorhandenen Funktionen für die Zwischenablage, Kopieren, Ausschneiden und Einfügen, finden sich hier Kommandos, um grafische und numerische Auswertungen auf einen bestimmten Teil der Daten zu beschränken oder die Formatierung und den Inhalt der Zellen zu löschen.

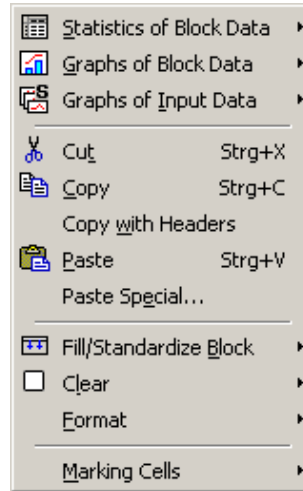


Abbildung 2.30. Statistica: Kontextmenü der Tabellen

Hinter dem Menüpunkt *Fill/Standardize* verbergen sich Befehle zu Standardisierung von Zeilen und Spalten und zum Füllen mit Zufallsvariablen. Leider standardisiert das Programm dabei nur über die Standardabweichung der Stichprobe, hier sollte dem Nutzer auch die Möglichkeit der Standardisierung über die Standardabweichung der Grundgesamtheit erlaubt sein.

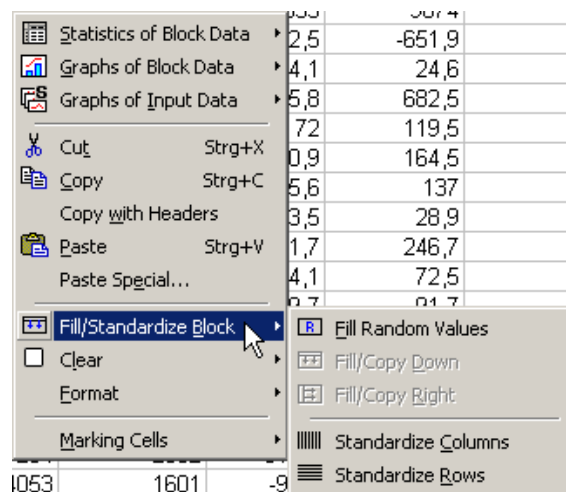


Abbildung 2.31. Statistica: Fill/Standardize Menü

2 Analyse

Hinter Statistics of Block Data, Graphs of Block Data und Graphs of Input Data verbergen sich weitere Menüs zur Datenauswertung. Die ersten beiden Menüs wenden Grafik- und mathematische Operationen nur auf den aktuell markierten Block der Tabelle an, so lassen sich einzelne Methoden und Grafiken für einzelne Teile der Tabelle anwenden, ohne diese Daten zuvor in ein anderes Tabellenblatt zu kopieren.

Graphs of Input Data stellt einen Shortcut zu den am häufigsten verwendeten Grafikoperationen dar.

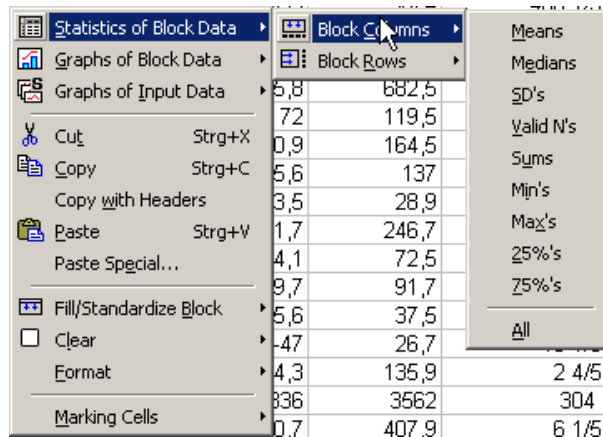


Abbildung 2.32. Statistica: blockweise Anwendung von Methoden

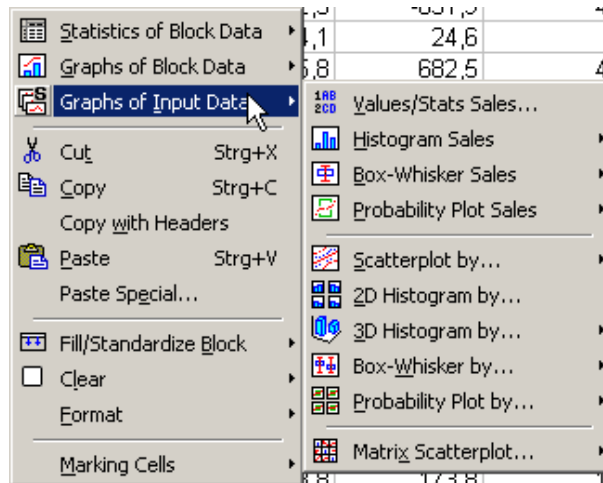


Abbildung 2.33. Statistica: blockweise Anwendung von Grafiken

2.6.4 Der Workbook-Manager

Ähnlich wie *S-Plus* bietet auch *Statistica* einen Workbookmanager, in dem die einzelnen Teile einer Datenauswertung wie Datensätze, Grafiken und Reports zusammen aufbewahrt werden können. Im linken Fenster befindet sich der Objektbaum, in dem die Objekte (Tabellen, Grafiken, Matrizen) in Ordnern abgelegt werden.

Sehr komfortabel ist die Möglichkeit, nicht nur *Statistica*-eigene Formate in diesem Baum zu hinterlegen, sondern auch OLE/ActiveX Komponenten. So läßt sich beispielsweise eine CorelDraw Zeichnung einfügen, die dann ihrerseits eine *Statistica*-Tabelle enthalten kann, siehe Abbildung 2.33.

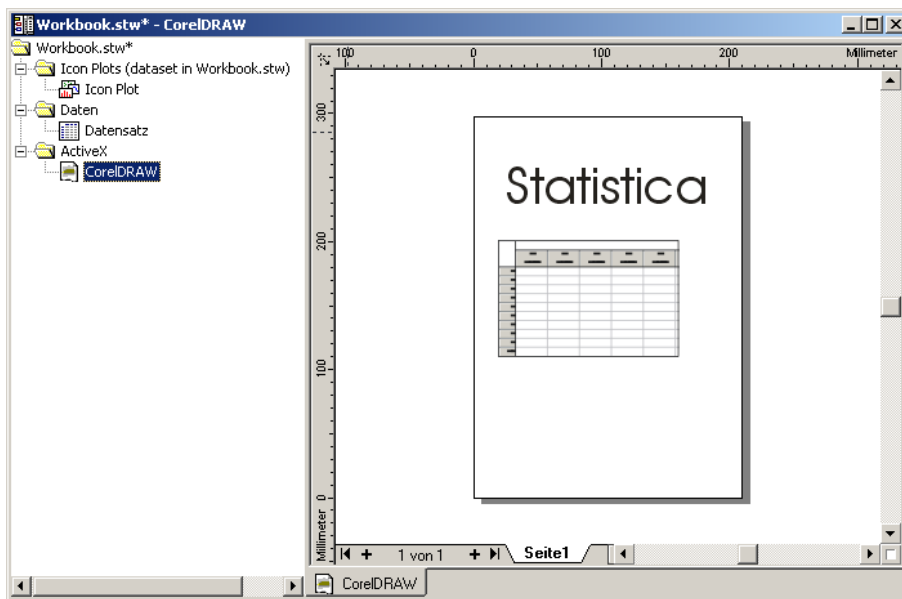


Abbildung 2.34. Statistica: ActiveX Komponenten im Workbook-Manager

2.6.5 Export-Optionen

Im Vergleich zu anderen Programmen sind die Optionen für den Grafikexport überschaubar. Statistica exportiert in das hauseigene STG-Format, in JPEG, BMP, PNG und WMF. Exporte nach EPS oder PS sind nicht verfügbar.

Statistica	dBase
HTML	Excel
Lotus 1-2-3	SPSS
ASCII-Text	Quattro Pro

Tabelle 2.3. Statistica: Exportformate

2 Analyse

Leider lassen sich Grafiken und Tabellen nicht direkt aus dem Workbook-Manager exportieren, sie müssen zuerst aus dem Workbook per Drag & Drop auf die Arbeitsfläche gezogen werden, von wo aus sie dann exportierbar sind.

2.6.6 Hilfesystem

Zusätzlich zu den Hilfedateien im Windows-eigenen HLP-Format stellt StatSoft Inc. über das Internet ein umfangreiches elektronisches Buch bereit (<http://www.statsoft.com/textbook/stathome.html>), das die statistischen Grundlagen der in *Statistica* benutzten Methoden erklärt. Ein sehr ausführliches Glossar, eine große Zahl an Referenzen und der „Statistical Advisor“, machen dieses E-book zu einer interessanten Quelle für den Umgang mit *Statistica*.

2.7 Stata 8/SE

Stata, entwickelt und vertrieben von der US-amerikanischen Stata Corporation, war bis zur Version 7 ebenso wie *R* und *Gauss* streng konsolenorientiert. Erst seit der aktuellen Version 8 verfügt Stata über zahlreiche Menüs zum Zugriff auf Datenmanipulation, grafische und numerische Auswertungsmethoden.

Stata unterscheidet sich in seiner Herangehensweise von anderen matrizenbasierten Programmen, wie beispielsweise *Gauss* oder *XploRe*, dadurch, dass hier die Daten in Form einzelner Variablen vorliegen, die dann bei den Auswertungsmethoden benutzt werden.

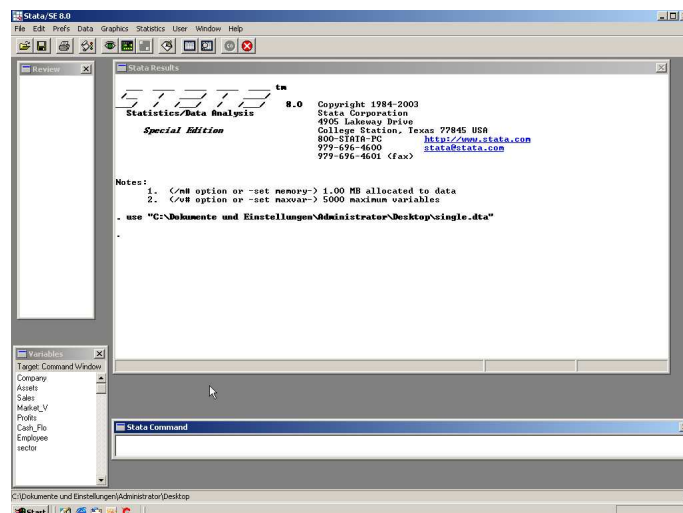


Abbildung 2.35. Stata: Benutzeroberfläche

2.7.1 Datenimport

Stata selbst liest nur verschiedene ASCII-Dialekte direkt ein, alle anderen Formate müssen außerhalb von Stata nach ASCII konvertiert werden. Die Stata Corporation bietet jedoch mit Stat/Transfer ein eigenes Programm für den Datenaustausch zwischen verschiedenen Programmen an.

In seiner aktuellen Version 7 konvertiert *Stat/Transfer* zwischen einer Vielzahl von Formaten (siehe Tabelle 2.4):

Microsoft Access	ODBC
dBase	OSIRIS
ASCII	Paradox
Epi Info	Quattro Pro
Microsoft Excel	S-Plus
Fixed format ASCII	SAS
FoxPro	SAS Transport
Gauss	SAS Value Labels
JMP	SAS Version 7-9
LIMDEP	SPSS Datafiles
Lotus 1-2-3	SPSS Portable Files
Matlab 5	Stata
Mineset	Statistica
Minitab	Systat (Win/Mac)

Tabelle 2.4. Stat/Transfer: unterstützte Dateiformate

Beim Import des Beispiel-Datensatzes aus einer Text- beziehungsweise Excel-Datei übernimmt Stat/Transfer automatisch Typ und Anzahl der Variablen und bietet außerdem an, die Zahl der importierten Datensätze durch verschiedene relationale Operatoren zu beschränken.

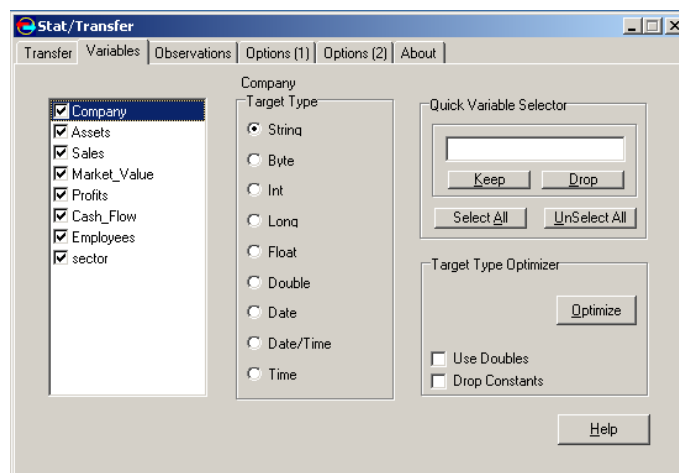


Abbildung 2.36. Stata: Stat/Transfer

2 Analyse

Der Nutzer kann eine Vielzahl von Parametern ändern, um die Formate von Ein- und Ausgabedateien festzulegen.

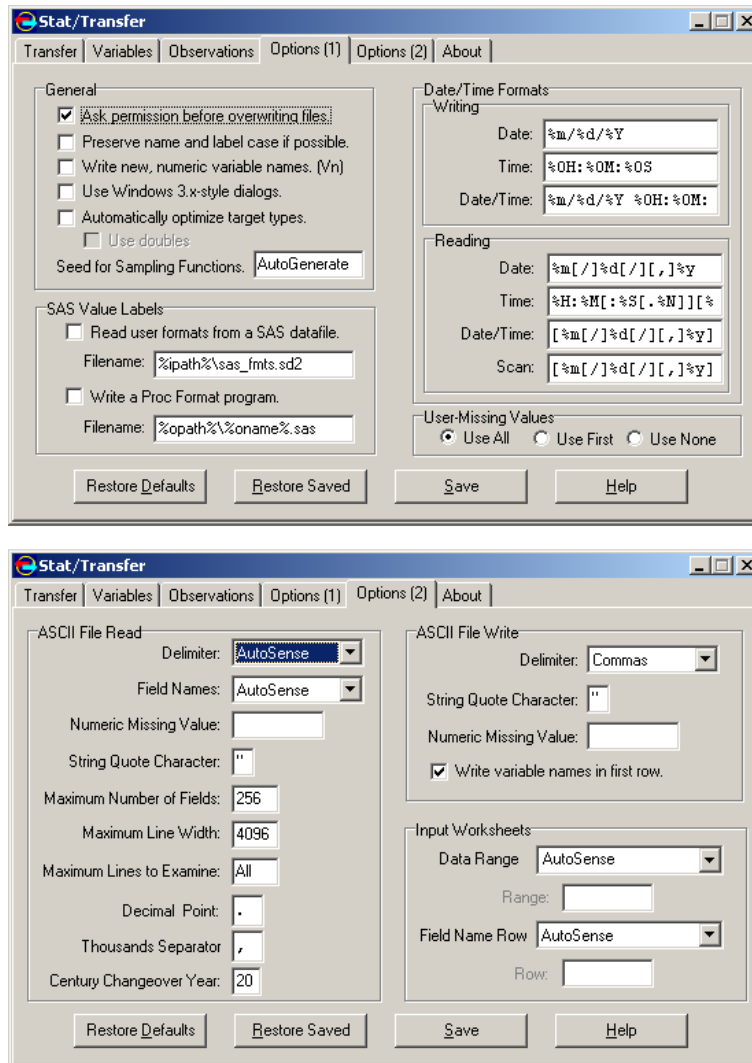


Abbildung 2.37. Stata: Stat/Transfer Optionen

2.7.2 Datenmanipulation

Stata beinhaltet zwei Spreadsheet-Umgebungen, den *Data Browser* und den *Data Editor*, die über die entsprechenden Buttons in der Menüleiste und die Kommandos `edit` und `browse` zur Verfügung stehen. Während der *Data Browser* eine schreibgeschützte Umgebung zur reinen Ansicht der Daten bietet, lassen sich die Werte im *Data Editor* durch den Nutzer auch editieren.

	Company	Assets	Sales	Market_U	Profits	Cash_Flo
1	Air_Products	2687	1870	1890	145.7	352.2
2	IBM	52634	50056	95697	6555	9874
3	Cigna	44736	16197	4653	-732.5	-651.9
4	American_Savings_Bank_FSB	3614	367	90	14.1	24.6
5	AMR	6425	6131	2448	345.8	682.5
6	Apple_Computer	1022	1754	1370	72	119.5
7	Armstrong_World_Industries	1093	1679	1070	100.9	164.5
8	Bally_Manufacturing	1529	1295	444	25.6	137
9	Bank_South	2788	271	304	23.5	28.9
10	Mellon_Bank	33406	3222	1413	201.7	246.7
11	H_and_R_Block	327	542	959	54.1	72.5
12	Brooklyn_Union_Gas	1117	1038	478	59.7	91.7
13	California_First_Bank	5401	550	376	25.6	37.5
14	CBI_Industries	1128	1516	430	-47	26.7
15	Central_Illinois_Public_Service	1633	701	679	74.3	135.9
16	General_Electric	26432	28285	33172	2336	3562
17	Cleveland_Electric_Illuminating	5651	1254	2002	310.7	407.9
18	Columbia_Gas_System	5835	4053	1601	-93.8	173.8
19	Community_Psychiatric_Centers	278	205	853	44.8	50.5
20	Continental_Telecom	5074	2557	1892	239.9	578.3
21	Crown_Cork_and_Seal	866	1487	944	71.7	115.4
22	Dayton-Hudson	4418	8793	4459	283.6	456.5
23	Digital_Equipment	6914	7029	7957	400.6	754.7
24	Dillard_Department_Stores	862	1601	1093	66.9	106.8

Abbildung 2.38. Stata: Variableneditor

2.7.3 Grafikexport

Stata verfügt über eine Auswahl an Exportfiltern, um grafischen Output zu speichern. Der Exportdialog, den man über das Kontextmenü einer Grafik erreicht (das auch noch Funktionen zum Drucken und kopieren in die Zwischenablage enthält), bietet die Möglichkeit, in die folgenden Formate zu exportieren:

Stata Graph	As-is Graph
Windows Metafile	Enhanced Metafile
PNG	Postscript
Encapsulated Postscript	

Tabelle 2.5. Stata: Dateiformate für Grafikexport

Diese Formate decken alle üblichen Einsatzzwecke (Internet, \LaTeX , Word) ab, in denen die exportierten Grafiken später weiterverwendet werden.

2.7.4 Export

Ebenso wie beim Import von Daten bietet *Stata* eine geringe Auswahl an Exportformaten. Einzelne Datensätze lassen sich im *Stata 7* und *8* Format speichern, über das Untermenü *Export* kann der Anwender Datensätze ins CSV- und ASCII-Format exportieren.

2 Analyse

Der Export-Dialog untergliedert sich in zwei Bildschirme (drei beim ASCII-Export), wo die zu exportierenden Variablen, der Dateiname und weitere Exportoptionen angegeben werden können. Über den `if/in` Bildschirm kann der Nutzer mittels mathematischer und logischer Ausdrücke festlegen, welche Werte er exportieren möchte oder die Spanne der zu exportierenden Variablen anhand von Start- und Endzeile angeben.

Wird ins ASCII-Format exportiert, läßt sich zusätzlich noch die Ausrichtung der Strings festlegen.

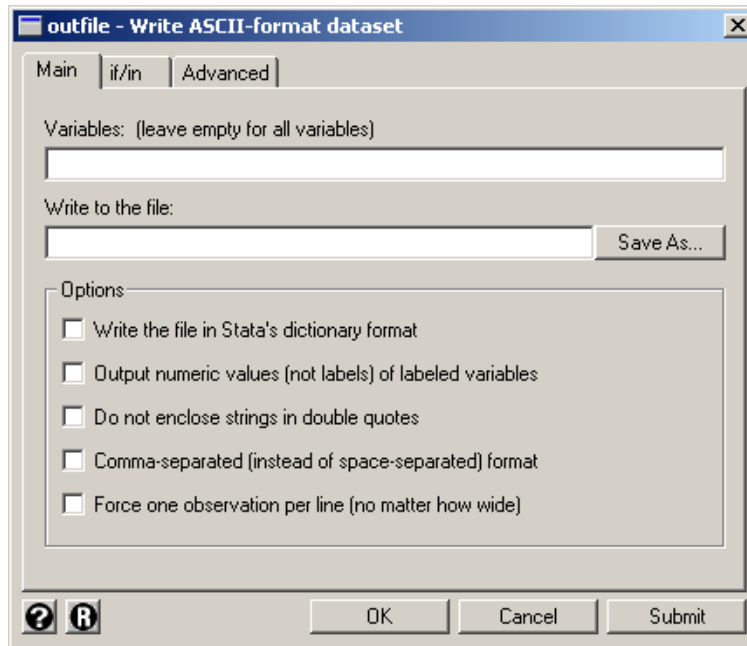


Abbildung 2.39. Stata: Export-Dialog

2.7.5 Hilfesystem

Das Hilfesystem liegt in einem STATA-eigenen Format vor und besteht aus einer Vielzahl von Dateien, die im ASCII-Format abgespeichert sind. Es existieren für Stata einige Programme (`makehlp`), die es dem Nutzer erlauben, eigene Hilfedateien zu schreiben.

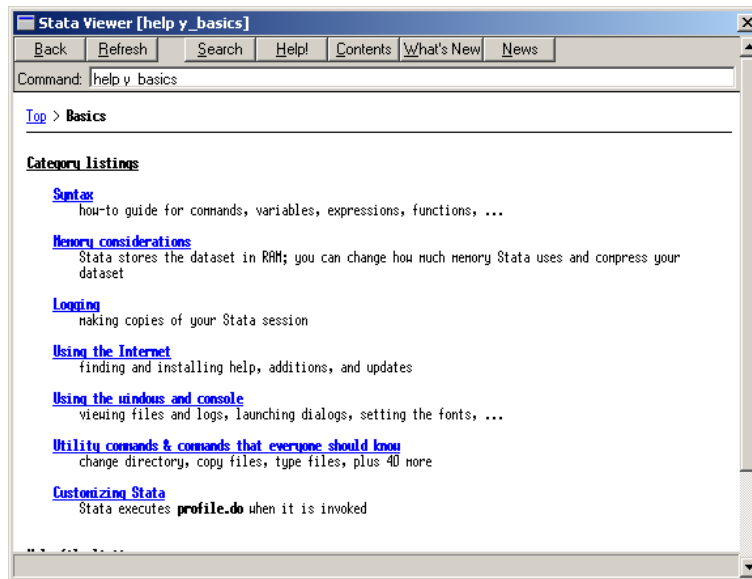


Abbildung 2.40. Stata: Hilfesystem

2.8 Tabellarische Übersichten

Name	MDI	Preis in Eur	Betriebssysteme	Hilfeformat	WWW
S-Plus	ja	n.v.	Win, Unix	HLP	www.insightful.com
R	ja (Win)	frei	Win, Unix	eigenes, HTML	www.r-project.org
Gauss	ja	3500	Win, Unix	HLP	www.aptech.com
Minitab	ja	1400	Win, Unix	HLP	www.minitab.com
SPSS	nein	1400	Win, Unix	HLP	www.spss.com
Stata	ja	1400	Win, Unix	eigenes	www.stata.com
Statistica	ja	900	Win, Unix	HLP	www.statsoftinc.com

Tabelle 2.6. Übersicht Statistikprogramme

Die angegebenen Preise stammen entweder direkt von der Herstellerseite oder wurden den Webseiten von Distributoren entnommen. Der Preis steht hier für den Kaufpreis einer kommerziellen Einzelplatzlizenz, sofern ermittelbar (Unix schließt auch Versionen für Linux mit ein).

Name	TXT	ODBC	Excel	SPSS	DBASE
S-Plus	ja	ja	ja	ja	ja
R	ja	ja	nein	ja	nein
Gauss	ja	ja	ja	nein	ja
Minitab	ja	nein	ja	nein	ja
SPSS	ja	ja	ja	ja	ja
Stata	ja	ja	nein	nein	nein
Statistica	ja	ja	ja	ja	ja

Tabelle 2.7. eingebaute Datenimportformate

Name	BMP	EPS/PS	JPEG	TIFF	PNG	EMF/WMF
S-Plus	ja	ja	ja	ja	ja	ja
R	ja	ja	nein	nein	nein	ja
Gauss	ja	ja	nein	nein	nein	ja
Minitab	ja	nein	ja	ja	ja	nein
SPSS	ja	ja	ja	ja	ja	ja
Stata	ja	ja	nein	nein	nein	ja
Statistica	ja	nein	ja	nein	ja	ja

Tabelle 2.8. Formatübersicht Grafikexport

2.9 Fazit

Es hat sich gezeigt, dass zwischen den einzelnen Paketen große Unterschiede bestehen, was den Import und Export von Daten und Grafiken betrifft.

Beim Import aus ODBC-Quellen sind die meisten Programme, die diese Schnittstelle unterstützen, komfortabel. Die gewünschten Daten lassen sich leicht auswählen und importieren.

Der Umgang mit den einzelnen Dateneditoren der einzelnen Programme fiel recht unterschiedlich aus. Zwischen den nach Ansicht des Autors sehr guten Editoren von S-Plus und Statistica zu dem rudimentären Editor von Gauss, das zu den teuersten Programmen gehört, bestehen erhebliche Unterschiede.

Beim Export fiel auf, dass die Unterstützung von vektor-basierten Formaten bei nur wenigen Programmen ausgereift war. Unter den untersuchten Programmen war keines, das eine eingebaute Unterstützung für das auf

2 Analyse

XML-basierende SVG-Format bot. Für R gibt es einen externen Treiber (<http://www.darkridge.com/~jake/RSvg>), der aber in dieser Arbeit nicht getestet werden konnte.

3 Implementierung

Nachdem im ersten Kapitel untersucht wurde, wie verschiedene Anbieter statistischer Software die Verwaltung von Daten in ihren Produkten umgesetzt haben, hat der Autor einen Matrizeneditor für XploRe implementiert und möchte die Ergebnisse in diesem Kapitel darlegen.

Da Borland C++ 5 aus dem Jahr 1997 stammt und nicht mehr durch den Hersteller unterstützt wird, wurde durch das Yxilon-Team beschlossen, die Oberfläche von XploRe komplett neu zu implementieren. Als Entwicklungsumgebung wurde die neueste Version des Borland C++ Builders gewählt.

3.1 Die grafische Benutzeroberfläche

Die grafische Benutzeroberfläche basiert auf der Arbeit von Martin Schröter, der das bestehende XploRe Interface mit dem Borland C++ Builder 6 umsetzte. Ausgehend von seiner Arbeit wurde der Editor und seine Funktionalität integriert.

3.2 Der Dateneditor

Beim Programmstart wird der Dateneditor mit einer voreingestellten Größe von 25×25 Zellen geladen, die Werte für die Zeilen- und Spaltenzahl können auch durch den Nutzer im *[Editor]*-Abschnitt der *xploRe.ini* verändert werden. Zur Bedienung der Editor-Funktionen wurde das Untermenü *Data* in der Menüleiste erweitert, das die Befehle für das Laden und Speichern von Objekten sowie den Befehl für das Setzen von Zeilen- und Spaltenzahl enthält.

3.2.1 Laden von Datensätzen

Über eine DLL (Dynamic Link Library), die mein Kollege Szymon Borak und ich entwickelt haben, kann der Nutzer mit dem Befehl *Open* im *Data*-Menü Datensätze in den Dateneditor laden. Die Implementation als DLL hat den Vorteil, daß einzelne Teile in den Leseroutinen der DLL geändert werden können, ohne das Hauptprogramm ändern zu müssen.

3 Implementierung

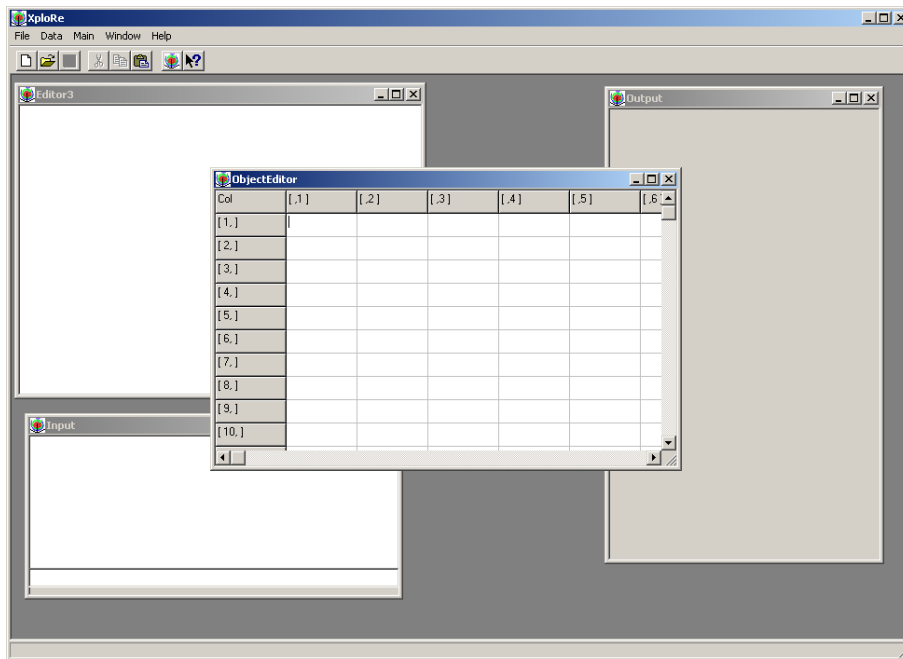


Abbildung 3.1. Yxilon: grafische Benutzeroberfläche

The screenshot shows the 'ObjectEditor' window with a data table. The table has 7 columns labeled '[.1]' through '[.7]' and 11 rows labeled '[1.]' through '[11.]'. The data values are as follows:

Col	[.1]	[.2]	[.3]	[.4]	[.5]	[.6]	[.7]
[1.]	0,1018	0,0288	0,0148	0,0529	0	0	0
[2.]	0,1065	0,038	0,0146	0,0476	0	0	0
[3.]	0,1068	0,0292	0,0262	0,0429	0	0	0
[4.]	0,1154	0,0183	0,0566	0,0312	0	0	0
[5.]	0,1176	0,0161	0,0084	0,0616	0,0021	0,0081	0
[6.]	0,1189	0,0489	0,0095	0,0496	0	0	0
[7.]	0,1192	0,0195	0,0388	0,0516	0	0,0015	0
[8.]	0,1194	0,0254	0,0196	0,0687	0	0	0
[9.]	0,1195	0,0314	0,0139	0,0528	0	0	0
[10.]	0,1233	0,0188	0,0138	0,0537	0	0,0008	0,01
[11.]	0,1245	0,0604	0	0,061	0	0	0

Abbildung 3.2. Yxilon: Dateneditor

3 Implementierung

Ein Klick auf den Menüeintrag `Open` öffnet ein Bildschirmmenü, in dem der Anwender Dateiname, Spaltentrennzeichen und die zu importierenden Spalten und Zeilen angeben kann. Voreingestellte Trenner sind Leerzeichen und Tabulatoren, standardmäßig werden alle Spalten und Zeilen eingelesen.

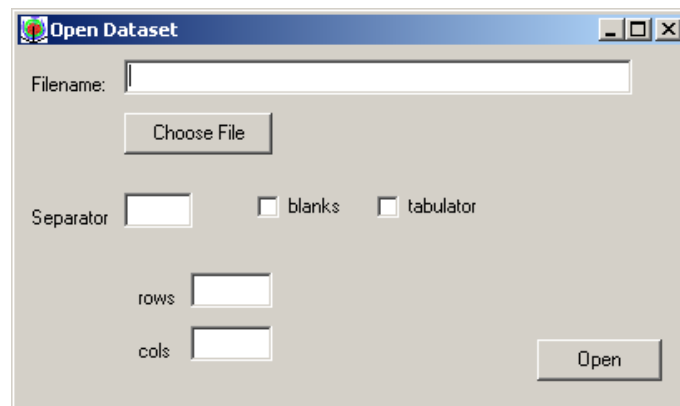


Abbildung 3.3. Yxilon: Open Data-Dialog

3.2.2 Das Popup-Menü

Klickt der Nutzer innerhalb des Dateneditors mit der rechten Maustaste, öffnet sich ein sogenanntes Popup-Menü, das folgende die Funktionen bereitstellt:

Delete Row: Dieser Befehl löscht die aktuelle Zeile nach einer Sicherheitsabfrage.

Delete Column: Delete Column löscht die aktuelle Spalte. Ebenso wie bei Delete Row muß der Nutzer den Löschvorgang bestätigen.

Insert Row: fügt eine Zeile vor der aktuellen Zeile ein.

Insert Column: Fügt eine Spalte vor der aktuellen Spalte ein.

Clear All: Löscht, nach Bestätigung durch eine Abfrage, den Inhalt.

Hide Column: setzt die Breite der aktuellen Spalte auf 0.

Show all Columns: Setzt die Breite aller Spalten auf einen vordefinierten Wert.

Cut: Schneidet den Inhalt der aktuellen Zelle aus.

Copy: Kopiert den Inhalt der aktuellen Zelle.

Paste: Fügt einen Wert aus der Zwischenablage ein.

3 Implementierung

Die Funktionen zum Einfügen und Löschen von Zeilen und Spalten funktionieren alle ähnlich, sie arbeiten mit Schleifen. So wird beim Einfügen einer Zeile am Ende der Tabelle eine Zeile hinzugefügt und der Inhalt der Tabelle von der aktuellen Zeile bis zum Ende der Tabelle verschoben. Dadurch entsteht an der aktuellen Cursorposition eine Leerzeile. Die Verfahrensweise beim Einfügen von Spalten ist analog.

Die Löschfunktionen arbeiten genau entgegengesetzt. Erst wird der Inhalt der folgenden Spalten/Zeilen um eine Zeile nach links bzw. oben verschoben, anschließend wird die Spalten bzw. Zeilenzahl um 1 verringert.

Die Befehle `Cut`, `Copy` und `Paste` verwenden interne Borland Standardaktionen. Dazu wird eine neue Instanz von `TAction` angelegt, einem Borland-eigenen Container für Befehle. Dieser Container kann anschließend mit den vordefinierten oder eigenen Befehlen gefüllt werden, die dann dem Popuppemü zugewiesen werden. Dadurch ist es nicht notwendig, die Funktionen der Win32-API selbst zu implementieren.

3.3 Die *xplore.ini*

Microsoft empfiehlt, für das Speichern von Benutzereinstellungen die Registry zu benutzen. Um aber soweit wie möglich von den Einstellungen der Registry unabhängig zu bleiben, habe ich mich für eine herkömmliche INI-Datei entschieden, die den Vorteil einer leichten Änderbarkeit durch den Benutzer bietet. Die Werte werden über Win32-API Funktionen ausgelesen und stehen dann als Variablenwerte zur Verfügung.

Die *xplore.ini* gliedert sich in fünf Abschnitte, die den entsprechenden Teilen der GUI entsprechen:

- Program
- Editor
- Input
- Output
- APSS

Durch die Einstellungen kann für jedes einzelne der GUI-Fenster festgelegt werden, wo es auf dem Bildschirm erscheint und in welcher Größe es dargestellt wird.

3 Implementierung

```
[Editor]
rows = 25
cols = 25
mode = 0
Width = 500
Height= 300
Top = 50
Left = 20

[Program]
Width = 1000
Height= 750
Top = 0
Left = 0

[Output]
Width = 300
Height= 400
Top = 20
Left = 700

[Input]
Width = 300
Height= 200
Top = 400
Left = 25

[APSS]
xpl4nethome = "http://www.xplore-stat.de"
```

3.4 Wechsel zwischen Zeilen- und Spaltenmodus

Eine der innovativen Funktionen von *Minitab* war die Änderung der Laufrichtung über einen Klick in die oberste linke Ecke. Da dies für die Eingabe von Daten sehr nützlich ist, wurde diese Funktion auch für *Yxilon* implementiert: Über die globale Variable `mode`, deren Wert mit jedem Klick invertiert wird, läßt sich steuern, ob nach dem Druck auf `Enter` oder `Return` die rechte oder untere Zelle aktiviert wird.

Beim Start des Programms wird der Wert der Variablen aus der INI-Datei geladen. Alle Funktionen, die dann den Wert benötigen, lesen ihn aus dieser Variablen.

4 Ausblick

Mit der Beendigung dieser Diplomarbeit geht die eigentliche Arbeit an Yxilon selbst weiter. Innerhalb der nächsten Jahre soll ein Framework entstehen, das von Wissenschaftlern und Studenten gleichermaßen für die effiziente Auswertung ihrer Daten benutzt werden kann und reproduzierbare Forschung ermöglicht. Bild 4.1 zeigt den Aufbau von Yxilon als modulare Anwendung, die auf verschiedenen Plattformen läuft.

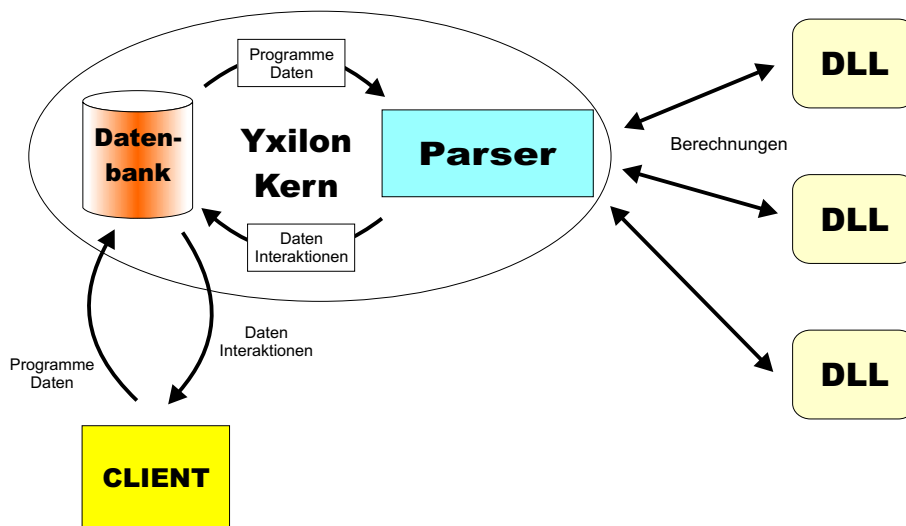


Abbildung 4.1. Yxilon: Übersicht

4.1 Integrierter Hilfebrowser

Das Hilfesystem von *XploRe* kann über ein zusätzliche MDI-Fenster direkt in die grafische Oberfläche integriert werden. In Nutzerbefragungen wird zu klären sein, ob diese integrierte Hilfe von den Anwendern akzeptiert wird oder die bisherige Lösung (externe Hilfe über den Standardbrowser) bevorzugt wird.

4.2 Internationalisierung

Yxilon wird als voreingestellte Sprache Englisch benutzen, es sollte aber dem Nutzer möglich sein, die grafische Benutzeroberfläche auf seine Sprache anzupassen.

Erste Tests mit einer zusätzlichen INI-Datei, die die notwendigen Zeichenketten enthielt, waren vielversprechend und werden in der Zukunft fortgesetzt. Um auch nichteuropäische Zeichen zu integrieren, wird die Zusammenarbeit mit der japanischen *Three One Systems* angestrebt, die auch die Lokalisierung von XploRe 4 für den japanischen Markt vorgenommen hatten.

4.3 Parser

Der Parser bildet die Schnittstelle zwischen den Eingaben des Benutzers (entweder über das Input-Fenster oder ein Quantlet) und den zugehörigen Funktionsaufrufen. In der nahen Zukunft wird das vorrangige Ziel der Yxilon-Gruppe sein, diesen Parser zu komplettieren.

Der Parser wird vermutlich auch eingesetzt werden, um die Eingaben des Anwenders auf ihre Validität zu untersuchen. Im Moment findet keine Validitätsuntersuchung statt, der Dateneditor akzeptiert jeden String, auch solche mit Sonderzeichen.

4.4 Grafik

Einer der schwierigsten Abschnitte der Programmierung wird die Darstellung von zwei- und dreidimensionaler Grafik sein. Erste Versuche mit den im Borland C++ Builder vorhandenen Grafikroutinen verliefen erfolgversprechend.

Als interessant könnte sich dabei die Unterstützung von OpenGL (<http://www.opengl.org>) erweisen. Dieser Industriestandard, der 1992 vorgestellt wurde, ist die am weitesten verbreitete Schnittstelle zur Grafikprogrammierung, die meisten Grafikkarten bringen hardwarebasierende Unterstützung für OpenGL mit.

Führende Mathematik-Programme wie Mathematica, Maple, Mathcad und Matlab benutzen zur Darstellung hochwertiger Grafik OpenGL.

4.5 UML

Oestereich (2002) beschreibt die *Unified Modelling Language* so:

Die *Unified Modelling Language* ist eine Sprache und Notation zur Spezifikation, Konstruktion, Visualisierung und Dokumentation von Modellen für Softwaresysteme.

Die UML ist durch bedeutende Institutionen wie die ISO (International Standard Organization) und OMG (Object Management Group) zertifiziert, ihr Einsatz in Softwareprojekten verbessert nicht nur die Dokumentation, sondern erleichtert auch die Design- und Entwicklungsarbeit. UML-Diagramme wurden schon bei der Modellierung von XploRe-Komponenten genutzt, siehe Feuerhake (2002).

Mit Softwarepaketen wie *Rational Rose* (www.rational.com, jetzt IBM) kann die Klassenstruktur von Programmen am Bildschirm modelliert werden, ohne eine einzelne Zeile Code schreiben zu müssen. Die Software ist sogar in der Lage, aus den fertig modellierten Klassen den Programmcode mit Funktionsmustern und Variablen zu erzeugen, der dann vom Programmierer nur noch fertig ausprogrammiert werden muß.

Die UML enthält eine Reihe von Diagrammtypen, mit denen die Struktur und das Verhalten einer Anwendung beschrieben werden kann, dazu gehören:

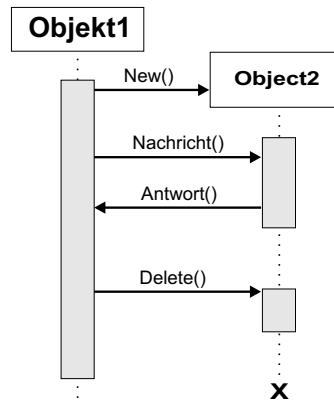
Klassen-Diagramm: Klassendiagramme werden dazu genutzt, um Attribute und Operationen zu beschreiben, die mit dieser Klasse erzeugt werden können.

Komponenten-Diagramm: Komponenten sind meist Aggregationen mehrerer Klassen, angestrebt wird im Unterschied zu Klassen auch die prinzipielle Austauschbarkeit. Ein mögliches Beispiel wäre die in einer DLL gekapselte `Read()`-Function zum Lesen von Datensätzen.

Sequenzdiagramm: Sequenzdiagramme dienen der Visualisierung der Nachrichten im Zeitablauf, die von den einzelnen Objekten innerhalb eines Softwareprogramms gesendet werden.

Für Beispiele von UML-Diagrammen siehe Bild 4.2

Sequenzdiagramm



Klassendiagramm

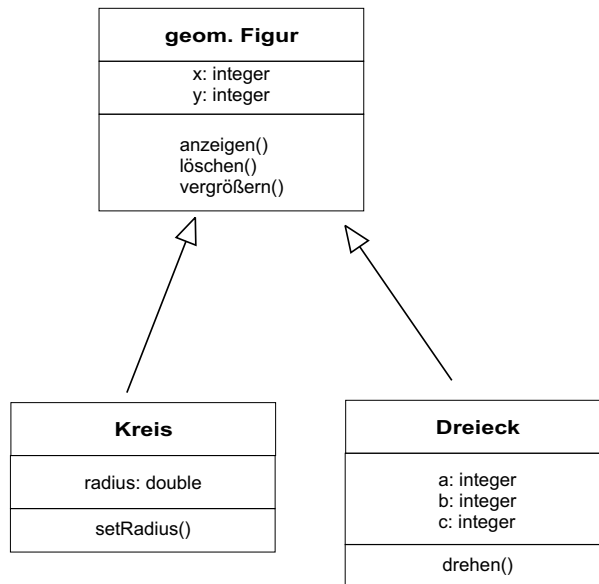


Abbildung 4.2. UML: Diagrammbeispiele

5 Anhang

5.1 Microsoft Hilfedateien

Seit den frühen 90er Jahren hat Microsoft einen Standard für Hilfedateien entwickelt, der von vielen Softwareanbietern für eigene Programme benutzt wird. Meist wird das Hilfesystem über die F1-Taste aufgerufen.

Dieses Hilfesystem besteht in den meisten Fällen aus einer Reihe von miteinander verlinkten Dokumenten, die ebenfalls Bilder, Grafiken oder Links enthalten. Die einzelnen Hilfedateien können in Kapitel unterteilt werden, ein Index sowie eine einfache Suchfunktion ermöglichen den schnellen Zugriff auf einzelne Suchbegriffe. Das HLP-Format ist in seinen Funktionen nur schlecht beschrieben und wird seit einigen Jahren durch das HTML-basierte CHM-Format verdrängt.

In der neuen .NET-Umgebung existiert mit MS Help 2.0 ein neuer Standard, der komplett in die Entwicklungsumgebung integriert ist.

Vorteil des Microsoft Hilfesystems ist die einheitliche Bedienung in allen Programmen, die dieses Format benutzen. Nachteilig für den Nutzer ist aber, dass keine eigenen Bestandteile zum Hilfesystem hinzugefügt werden können.

5.2 Kopien zitierter Literatur

Die folgenden fünf Seiten enthalten Kopien der beiden Internetquellen, dahinter folgt der benutzte Datensatz des DASL.

- Intel (2003)
- Eichinger (2001)

Um die Importfunktionen der einzelnen Statistikprogramme zu testen, wurde auf den Datensatz „Companies“ der *Data and Story Library* (DASL, <http://lib.stat.cmu.edu/DASL>) zurückgegriffen, der 79 Fälle mit Fakten der Forbes 500 Liste (eine jährliche Aufstellung der 500 größten Unternehmen) des Jahres 1986 enthält. Der Datensatz ist unverändert, es wurden lediglich die

5 Anhang

Firmen-Doppelnamen mit Unterstrichen zusammengefügt und kaufmännisches Und-Zeichen durch „and“ ersetzt, um möglichen Problemen beim Import der Daten zu begegnen. Der Datensatz enthält acht Variablen:

- Company: Company Name
- Assets: Anlagevermögen in Millionen \$
- Sales: Umsatz in Millionen \$
- Market Value: Börsenwert der Firma in Millionen \$
- Profits: Gewinn in Millionen \$
- Cash Flow: Cash Flow in Millionen \$
- Employees: Anzahl der Angestellten in Tausend
- Sector: Marktsektor, dem das Unternehmen zuzurechnen ist.

5 Anhang

Company	Assets	Sales	Value	Profits	Cash Flow	Employees	sector
Air Products	2687	1870	1890	145,7	352,2	18,2	Other
IBM	52634	50056	95697	6555,0	9874,0	400,2	HiTech
Cigna	44736	16197	4653	-732,5	-651,9	48,5	Finance
American Savings Bank FSB	3614	367	90	14,1	24,6	1,1	Finance
AMR	6425	6131	2448	345,8	682,5	49,5	Transportation
Apple Computer	1022	1754	1370	72,0	119,5	4,8	HiTech
Armstrong World Industries	1093	1679	1070	100,9	164,5	20,8	Manufacturing
Bally Manufacturing	1529	1295	444	25,6	137,0	19,4	Other
Bank South	2788	271	304	23,5	28,9	2,1	Finance
Mellon Bank	33406	3222	1413	201,7	246,7	15,8	Finance
H and R Block	327	542	959	54,1	72,5	2,8	Finance
Brooklyn Union Gas	1117	1038	478	59,7	91,7	3,8	Energy
California First Bank	5401	550	376	25,6	37,5	4,1	Finance
CBI Industries	1128	1516	430	-47,0	26,7	13,2	Manufacturing
Central Illinois Public Service	1633	701	679	74,3	135,9	2,8	Energy
General Electric	26432	28285	33172	2336,0	3562,0	304,0	HiTech
Cleveland Electric Illuminating	5651	1254	2002	310,7	407,9	6,2	Energy
Columbia Gas System	5835	4053	1601	-93,8	173,8	10,8	Energy
Community Psychiatric Centers	278	205	853	44,8	50,5	3,8	Medical
Continental Telecom	5074	2557	1892	239,9	578,3	21,9	Communication
Crown Cork and Seal	866	1487	944	71,7	115,4	12,6	Other
Dayton-Hudson	4418	8793	4459	283,6	456,5	128,0	Retail
Digital Equipment	6914	7029	7957	400,6	754,7	87,3	HiTech
Dillard Department Stores	862	1601	1093	66,9	106,8	16,0	Retail
Dreyfus	401	176	1084	55,6	57,0	0,7	Finance
Eg and G	430	1155	1045	55,7	70,8	22,5	HiTech
Ex-Cell-O	799	1140	683	57,6	89,2	15,4	Other
First American	4789	453	367	40,2	51,4	3,0	Finance
First Empire State	2548	264	181	22,2	26,2	2,1	Finance
First Tennessee National	5249	527	346	37,8	56,2	4,1	Finance
Florida Progress	3494	1653	1442	160,9	320,3	6,4	Energy
Fruehauf	1804	2564	483	70,5	164,9	26,6	Manufacturing
Norwest	21419	2516	930	107,6	164,7	15,6	Finance
Giant Food	623	2247	797	57,0	93,8	18,6	Retail
Great A and P Tea	1608	6615	829	56,1	134,0	65,0	Retail
Halliburton	4662	4781	2988	28,7	371,5	66,2	Manufacturing
Hewlett-Packard	5769	6571	9462	482,0	792,0	83,0	HiTech
Hospital Corp of America	6259	4152	3090	283,7	524,5	62,0	Medical
Idaho Power	1654	451	779	84,8	130,4	1,6	Energy
Bell Atlantic	19788	9084	10636	1092,9	2576,8	79,4	Communication
IU International	999	1878	393	-173,5	-108,1	23,3	Transportation
Kansas Power and Light	1679	1354	687	93,8	154,6	4,6	Energy
Kroger	4178	17124	2091	180,8	390,4	164,6	Retail
Liz Claiborne	223	557	1040	60,6	63,7	1,9	Other
LTV	6307	8199	598	-771,5	-524,3	57,5	Manufacturing
Marine Corp	3720	356	211	26,6	34,8	2,4	Finance
May Department Stores	3442	5080	2673	235,4	361,5	77,3	Retail
Phillips Petroleum	14045	15636	2754	418,0	1462,0	27,3	Energy
Mesa Petroleum	1257	355	181	167,5	304,0	0,6	Energy
Montana Power	1743	597	717	121,6	172,4	3,5	Energy
American Electric Power	13621	4848	4572	485,0	898,9	23,4	Energy
NCR	3940	4317	3940	315,2	566,3	62,0	HiTech
Norstar Bancorp	8998	882	988	93,0	119,0	7,4	Finance
Allied Signal	13271	9115	8190	-279,0	83,0	143,8	Other
Owens-Corning Fiberglas	2366	3305	1117	131,2	256,5	25,2	Manufacturing
Pan Am	2448	3484	1036	48,8	257,1	25,4	Transportation
Peoples Energy	1440	1617	639	81,7	126,4	3,5	Energy
National City	12505	1302	702	108,4	131,4	9,0	Finance
PPG Industries	4084	4346	3023	302,7	521,7	37,5	Manufacturing
Public Service Co of New Mexico	3010	749	1120	146,3	209,2	3,4	Energy
Republic Airlines	1286	1734	361	69,2	145,7	14,3	Transportation
AH Robins	707	706	275	61,4	77,8	6,1	Medical
San Diego Gas and Electric	3086	1739	1507	202,7	335,2	4,9	Energy
Shared Medical Systems	252	312	883	41,7	60,6	3,3	Medical
Southeast Banking	11052	1097	606	64,9	97,6	7,0	Finance
Sovran Financial	9672	1037	829	92,6	118,2	8,2	Finance
Stop and Shop Cos	1112	3689	542	30,3	96,9	43,5	Retail
Supermarkets General	1104	5123	910	63,7	133,3	48,5	Retail
Texel	478	672	866	67,1	101,6	5,4	HiTech
Textron	10348	5721	1915	223,6	322,5	49,5	Manufacturing
TWA	2769	3725	663	-208,4	12,4	29,1	Transportation
Turner	752	2149	101	11,1	15,2	2,6	Manufacturing
United Financial Group	4989	518	53	-3,1	-0,3	0,8	Finance
United Technologies	10528	14992	5377	312,7	710,7	184,8	Manufacturing
Valero Energy	1995	2662	341	34,7	100,7	2,3	Energy
Warner Communications	2286	2235	2306	195,3	219,0	8,0	Other
Western Air Lines	952	1307	309	35,4	92,8	10,3	Transportation
Wickes Cos	2957	2806	457	40,6	93,5	50,0	Retail
FW Woolworth	2535	5958	1921	177,0	288,0	118,1	Retail

Tabelle 5.1. „Companies“ Datensatz

Usability - Vorbemerkungen

"Usability is the measure of the quality of the user experience when interacting with something -- whether a Web site, a traditional software application, or any other device the user can operate in some way or another."

(Jakob Nielsen, 1998)

Trotz dieser breit angelegten Definition einer der Gallionsfiguren der Usability-Bewegung, beschränkt sich die Bezeichnung Usability in der Praxis auf computerverwandte Themen; meist nur auf Software. Um diesem Umstand die Ehre zu geben, wird im Folgenden auch nur von Computersoftware die Rede sein - immer eingedenk der Tatsache, daß auch in zunehmendem Maß andere komplexe elektronische Gerätschaften wie Handies oder Videorekorder auf ihre Usability überprüft werden.

Der Begriff Usability konnte sich deshalb im deutschen Sprachraum durchsetzen, weil alle Übersetzungsversuche (z. B. in Benutzbarkeit) keine bessere Verpackung für seine eigentliche Bedeutung sind. Da er außerdem mit den Begriffen Usefulness und Utility verwandt ist, würden zusätzliche deutsche Alternativen das Bild nur noch undurchsichtiger machen.

Diese Seiten sollen einen Einstieg in den Themenbereich geben. Verschiedene Begriffe werden erläutert. Es wird versucht, diese in einen einheitlichen Rahmen zu fassen, der es erlaubt, ihre Stellung zueinander zu erkennen.

Usability ist eine hypothetische Eigenschaft, die Software zugeschrieben wird, wenn sie "benutzerfreundlich", "angenehm zu bedienen", "geeignet zur Erfüllung einer bestimmten Aufgabe" und dergleichen mehr ist.

Immer größere Konkurrenz in der Softwarebranche führte und führt noch immer dazu, daß Programme, die diese Aspekte in höherem Maße berücksichtigen, ihre Marktposition verbessern können. Die Betonung liegt auf der unscharfen Formulierung "in höherem Maße". Was bedeutet "angenehm zu bedienen" konkret? Wie können all diese Charaktereigenschaften von Software in konkrete Ziele verwandelt werden?

Es wurde erkannt, daß Usability zwar als Ziel erstrebenswert ist, daß der Weg zu diesem Ziel jedoch noch reichlich unerforscht ist. Bislang wurde die Qualität von Software häufig nach einem eher darwinistischen Prinzip erprobt: Setzt sich ein Programm auf dem Markt durch, ist es gut - und umgekehrt. Andererseits: Sich auf einzelne Aspekte der Mensch-Computer-Schnittstelle zu beschränken und diese zu verbessern, wie es seit langem von Seite der wissenschaftlichen Grundlagenforschung versucht wurde, ließ den Weg unnötig steinig und langwierig erscheinen.

Aus dieser Zwickmühle heraus entwickelte sich als ein Kompromiß das **Usability Engineering**. Als Ingenieurwissenschaft sollte es sowohl praktischen Erfordernissen, aber auch wissenschaftlicher Überprüfung gerecht werden:

[Usability](#)

[Usability -
Vorbemerkungen](#)

[Usability -
Definition](#)

[Usability
Engineering](#)

[Usability
Engineering - Ziele](#)

[Usability Tests](#)

[Usability - Kosten](#)

[Usability Inspection](#)

[Heuristische
Evaluation](#)

[Cognitive
Walkthrough](#)

[Literatur](#)

- Das allgemeine Ziel Usability sollte je nach der spezifischen Situation in konkrete Eigenschaften o. Attribute unterteilt werden können.
- Diese Attribute sollten meßbar sein.
- Die Verantwortlichen sollten in der Lage sein können, kritische Werte für die Attribute anzugeben.
- Schließlich sollte ein Vergleich der beobachteten oder gemessenen Ist- mit den geforderten Soll-Werten eine Beurteilung der Usability der Software zulassen.

Dieser ganze Vorgang wird als **Prozeß des Usability Engineering** bezeichnet.

Diese Darstellung ist natürlich stark vereinfacht. Tatsächlich ist Usability Engineering ein iterativer Prozeß. Konnten in einer Überprüfungsphase gravierende Usabilityprobleme festgestellt werden, sollten diese natürlich eliminiert und ein weiterer Testlauf gestartet werden.

Im folgenden soll das Usability Engineering als Rahmen für weitere Ausführungen dienen. Darin sollen neben den bereits angesprochenen die Begriffe **Usability Testing, Usability Inspection, Usability Evaluation** definiert und gegeneinander abgegrenzt werden.



weiter: → [Usability - Definition](#)

zurück: ← [Usability](#)

✉ [Armin Eichinger](#)

Usability - Definition

Der Begriff Usability tritt bei verschiedenen Gelegenheiten in verschiedenen Geschmacksrichtungen auf. Einig sind sich diese meist nur in dem Punkt, daß es sich um ein facettenreiches Konstrukt handelt. Oft wird daher auf folgende Definition der Internationalen Organisation für Standardisierung (ISO 9241) zurückgegriffen:

Usability eines Produktes ist das Ausmaß, in dem es von einem *bestimmten Benutzer* verwendet werden kann, um *bestimmte Ziele* in einem *bestimmten Kontext* **effektiv**, **effizient** und **zufriedenstellend** zu erreichen.

Einige Punkte dieser Definition verdienen besondere Aufmerksamkeit:

- Usability ist demnach nicht allein die Eigenschaft eines Produktes, sondern das Attribut einer Interaktion eines Benutzers mit einem Produkt innerhalb eines bestimmten Kontextes.
- Die Usability eines Produktes kann nicht ohne weiteres auf andere Benutzer des gleichen Produktes übertragen werden.
- Ein Produkt sollte den Benutzer in die Lage versetzen, genaue und komplette Ergebnisse zu erzielen (Effektivität).
- Die Ressourcen, die ein Benutzer in diese Interaktion investieren muß, sollten in Relation zum Ergebnis stehen (Effizienz).

Notwendige Elemente zur Bestimmung der Usability einer Interaktion Benutzer-Produkt sind also folgende:

- Eine Beschreibung der Ziele der Interaktion
- Eine Beschreibung der Benutzer
- Eine Beschreibung der Aufgaben (d. h. der Handlungen, die auszuführen sind, um ein Ziel zu erreichen)
- Eine Beschreibung der Ausstattung (insbesondere Software und Hardware)
- Eine Beschreibung der Umgebung (d. h. der relevanten Eigenschaften der physikalischen und sozialen Umwelt; z. B. Organisationsstruktur, Raumtemperatur)
- Usability Meßgrößen (d. h. meßbare Attribute, die sich auf Effizienz, Effektivität und Zufriedenheit beziehen; z. B. benötigte Zeit, Fehlerraten, Fragebögen)

Nicht alle diese Aspekte sind bei jeder Analyse in gleichem Maße zu berücksichtigen; jede Situation verlangt eine spezifische Gewichtung der relevanten Faktoren. Entscheidend ist, daß der Kontext hinreichend detailliert beschrieben wird, so daß alle Faktoren, die einen Einfluß auf die Usability nehmen, identifiziert werden können.



weiter: → [Usability Engineering](#)

zurück: ← [Usability - Vorbemerkungen](#)

[Usability](#)

[Usability -
Vorbemerkungen](#)

[Usability -
Definition](#)

[Usability
Engineering](#)

[Usability
Engineering - Ziele](#)

[Usability Tests](#)

[Usability - Kosten](#)

[Usability Inspection](#)

[Heuristische
Evaluation](#)

[Cognitive
Walkthrough](#)

[Literatur](#)

intel.

United States Home | Select a Location

Site Map | Contact Us | About Intel

Products

Support

Search

 Home Computing
 Business Computing
 Developer
 Reseller / Solutions

Advanced Search

Research at Intel

▶ Research Areas

▶ Research Labs

▶ University Programs

▶ People

▶ News

▶ Events

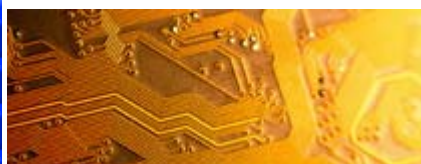
▶ Open Source Software

▶ Feedback

▶ Search

▶ Research & Development at Intel

▶ Intel Technology Journal

[▶ Research at Intel](#)
[▶ Research Areas](#)
[▶ Silicon](#)
[▶ Moore's Law](#)


Silicon

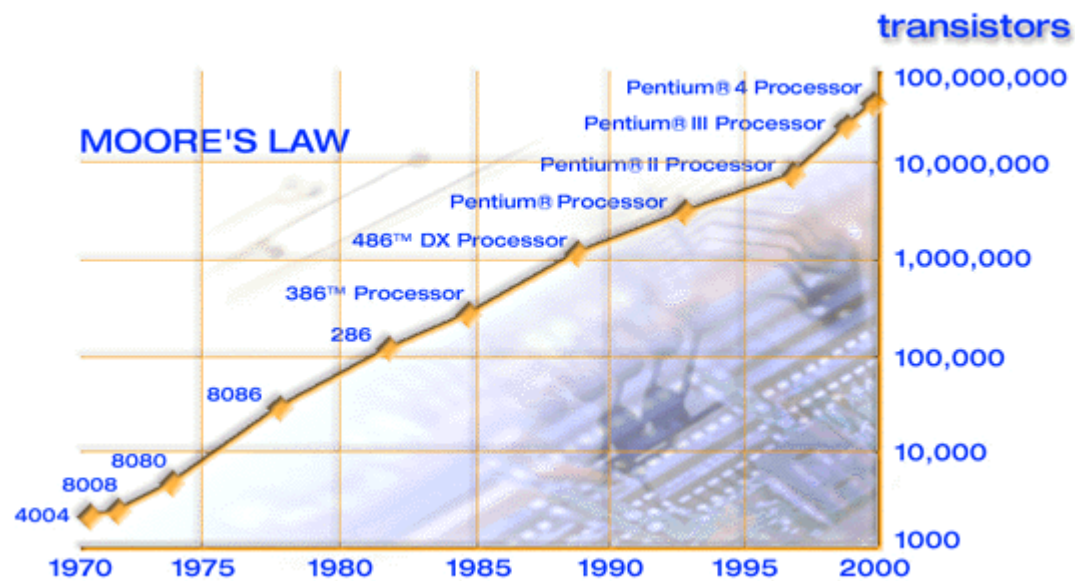
Moore's Law

Overview

Gordon Moore made his famous observation in 1965, just four years after the first planar integrated circuit was discovered. The press called it "Moore's Law" and the name has stuck. In his [original paper](#), Moore observed an exponential growth in the number of transistors per integrated circuit and predicted that this trend would continue. Through Intel's relentless technology advances, Moore's Law, the doubling of transistors every couple of years, has been maintained, and still holds true today. Intel expects that it will continue at least through the end of this decade. The mission of Intel's technology development team is to continue to break down barriers to Moore's Law.

Focus Research

- [Record Setting Processes](#)
- [Lithography](#)
- [300 mm Wafers](#)
- [Flash & Other Non-Volatile Memory Technologies](#)
- [Packaging](#)



	Year of introduction	Transistors
4004	1971	2,250
8008	1972	2,500
8080	1974	5,000
8086	1978	29,000
286	1982	120,000
386™ processor	1985	275,000
486™ DX processor	1989	1,180,000
Pentium® processor	1993	3,100,000
Pentium II processor	1997	7,500,000
Pentium III processor	1999	24,000,000
Pentium 4 processor	2000	42,000,000

[back to top](#)

Gordon Moore made his famous observation in 1965, just four years after the first planar integrated circuit was discovered. The press called it "Moore's Law" and the name has stuck. In his [original paper](#), Moore observed an exponential growth in the number of transistors per integrated circuit and predicted that this trend would continue. Through Intel's relentless technology advances, Moore's Law, the doubling of transistors every couple of years, has been maintained, and still holds true today. Intel expects that it will continue at least through the end of this decade. The mission of Intel's technology development team is to continue to break down barriers to Moore's Law.



[back to top](#)

Publications

- ["Cramming More Components Onto Integrated Circuits"](#)
(Acrobat PDF file, 167 KB)
Author: Gordon E. Moore
Publication: Electronics, April 19, 1965
- ["Microprocessors Circa 2000"](#)
(Acrobat PDF file, 543 KB)
Authors: Patrick Gelsinger, Paolo Gargini, Gerhard Parker, Albert Yu
Publication: IEEE Spectrum, October 1989

Presentations

- ["No Exponential is Forever ... but We Can Delay 'Forever'"](#)
(Acrobat PDF file, 2005 KB)
Presenter: Gordon Moore
Event: [International Solid State Circuits Conference \(ISSCC\)](#) **
Date: February 10, 2003

Focus Research

- [Record Setting Processes](#)
- [Lithography](#)
- [300 mm Wafers](#)
- [Flash & Other Non-Volatile Memory Technologies](#)
- [Packaging](#)

Related Links

- [Moore's Law](#)
- [MEMS](#)
- [Lead-Free Solutions](#)
- [People](#)
- [International Technology Roadmap](#)
- [External Research Programs](#)

5.3 Quellcodes

5.3.1 table_edit.tex

```

#include <vcl.h>
#pragma hdrstop

#include "table_edit.h"
#include "main.h"
char FileName[MAX_PATH];
//-----
#pragma package(smart_init)
#pragma link "Output"
#pragma resource "*.dfm"
TMDIChild3 *MDIChild3;
// fetches the "mode" form ini-file: go right (col) or down (row)
int mode = GetPrivateProfileInt("Editor", "mode", 1, FileName);

//-----
__fastcall TMDIChild3::TMDIChild3(TComponent* Owner) : TMDIChild1(Owner)
{
// takes values for height, width, top and left from ini-file

GetModuleFileName(NULL, FileName, sizeof(FileName));
strcpy(strrchr(FileName, '\\')+1, "xplore.ini");

StringGrid1->Height = GetPrivateProfileInt("Editor", "Height", 600, FileName);
StringGrid1->Width = GetPrivateProfileInt("Editor", "Width", 800, FileName);
Height= StringGrid1->Height;
Width = StringGrid1->Width;

Top= GetPrivateProfileInt("Editor", "Top", 15, FileName);
Left= GetPrivateProfileInt("Editor", "Left", 15, FileName);

}
//-----
void __fastcall TMDIChild3::MemoAlignment(TObject *Sender)
{
StringGrid1->Width = ClientWidth;
StringGrid1->Height = ClientHeight;
}
//-----
void __fastcall TMDIChild3::StringGrid1Enter(TObject *Sender)
{
// when Grid is shown, number of cols and rows is taken from ini-file

StringGrid1->ColCount = GetPrivateProfileInt("Editor", "cols", 15, FileName)+1;
StringGrid1->RowCount = GetPrivateProfileInt("Editor", "rows", 15, FileName)+1;

StringGrid1Fill(Sender);

}
//-----

void __fastcall TMDIChild3::InsertRow1Click(TObject *Sender)
{ // function to insert rows before the current line

StringGrid1->RowCount+=1;

```

5 Anhang

```
// takes care of right copying
for (int i=1;i<StringGrid1->ColCount;i++){
    for (int j=StringGrid1->RowCount-1;j>StringGrid1->Row;j--){
        StringGrid1->Cells[i][j] = StringGrid1->Cells[i][j-1];
    }
}

// inserts "" to inserted row
for (int i=1;i<StringGrid1->ColCount;i++){
    StringGrid1->Cells[i][StringGrid1->Row] = "";
}
// fill grid
StringGrid1Fill(Sender);

}
//-----

void __fastcall TMDIChild3::InsertColumn1Click(TObject *Sender)
{ // insert column and copy correctly

StringGrid1->ColCount+=1;

for (int i=1;i<StringGrid1->RowCount;i++){

    for (int j=StringGrid1->ColCount;j>StringGrid1->Col;j--){
        StringGrid1->Cells[j][i] = StringGrid1->Cells[j-1][i];
    }
}

for (int i=1;i<StringGrid1->RowCount;i++){
    StringGrid1->Cells[StringGrid1->Col][i] = "";
}
// fill grid
StringGrid1Fill(Sender);
}
//-----

void __fastcall TMDIChild3::StringGrid1Fill(TObject *Sender)
{
/*
This function fills the first row and first column with index.
*/

for (int i=0; i<(StringGrid1->ColCount+1);i++)
for (int j=0; j<(StringGrid1->RowCount+1);j++)
{
    if (i==0) StringGrid1->Cells[i][j] = "[_"+IntToStr(j)+",_]";
    else if (j==0) StringGrid1->Cells[i][j] = "[_."+IntToStr(i)+",_]";
}

if (mode==0)
    StringGrid1->Cells[0][0] = "Row";
else // mode = 1
    StringGrid1->Cells[0][0] = "Col";

} // end method
//-----
void __fastcall TMDIChild3::StringGrid1MouseUp(TObject *Sender,
TMouseButton Button, TShiftState Shift, int X, int Y)
{ /*
This function is used for switching between row and col mode. it checks
whether we are inside the topleft cell
*/
```

5 Anhang

```
    if ( (X<StringGrid1->ColWidths[0]) && (Y<StringGrid1->RowHeights[0])) {
        if (StringGrid1->Cells[0][0]=="Row") {
            StringGrid1->Cells[0][0]="Col";
            mode=1;
        }
        else{
            StringGrid1->Cells[0][0]="Row" ;
            mode=0;
        }
    }
}
//-----

void __fastcall TMDIChild3::ClearGrid1Click(TObject *Sender)
{ // empties the Stringgrid and asks before

    if (MessageDlg("Clear_Grid?", mtConfirmation, TMsgDlgButtons() << mbYes << mbNo,0) == mrYes)
    {
        for (int i=0; i<(StringGrid1->ColCount+1);i++)
        for (int j=0; j<(StringGrid1->RowCount+1);j++)
        {
            StringGrid1->Cells[i][j] = "";
        }
        StringGrid1Fill(Sender);
    }
}
//-----

void __fastcall TMDIChild3::DeleteRow1Click(TObject *Sender)
{ //deletes a row, if there are more than two rows left

    if (StringGrid1->RowCount>2){
    if (MessageDlg("Delete_row?", mtConfirmation, TMsgDlgButtons() << mbYes << mbNo,0) == mrYes)
    {
        for (int i =0;i<StringGrid1->ColCount;i++){
            for (int j= StringGrid1->Row;j<StringGrid1->RowCount-1;j++){

                StringGrid1->Cells[i][j] = StringGrid1->Cells[i][j+1];
            }
        }
        StringGrid1->RowCount-=1;
        StringGrid1Fill(Sender);
    } // end dialog

    } // end if
}
//-----

void __fastcall TMDIChild3::DeleteColumn1Click(TObject *Sender)
{ // deletes a column
if (StringGrid1->ColCount>2){
if (MessageDlg("Delete_column?", mtConfirmation, TMsgDlgButtons() << mbYes << mbNo,0) == mrYes)
{

for (int i=1;i<StringGrid1->RowCount;i++){

        for (int j=StringGrid1->Col;j<StringGrid1->ColCount;j++){
            StringGrid1->Cells[j][i] = StringGrid1->Cells[j+1][i];
        } // end for 1
    } // end for 1

    StringGrid1->ColCount-=1;
    StringGrid1Fill(Sender);

} // End Dialog
```

5 Anhang

```
} // end if
}
//-----

void __fastcall TMDIChild3::StringGrid1KeyPress(TObject *Sender, char &Key)
{
// after pressing enter/return goes one cell further, depending on MODE
if (Key==015){

    // check for missing values
    for (int i=1; i<(StringGrid1->Col+1); i++) {
        for (int j=1; j<(StringGrid1->Row+1); j++){
            if (StringGrid1->Cells[i][j] == ""){
                StringGrid1->Cells[i][j]=".";
            }
        }
    }

    if (mode==0){
        if (StringGrid1->Row<StringGrid1->RowCount-1){
            StringGrid1->Row=StringGrid1->Row+1; }
    }
    else {
        if (StringGrid1->Col<StringGrid1->ColCount-1){
            StringGrid1->Col=StringGrid1->Col+1;
        }
    }
} // end of first IF
} // end method
//-----

void __fastcall TMDIChild3::FormActivate(TObject *Sender)
{ // sets the menu back to mainmenu1, if t.editor is activated
MainForm->Menu = MainForm->MainMenu1;
}
//-----

void __fastcall TMDIChild3::HideColumn1Click(TObject *Sender)
{ // hides current Column
StringGrid1->ColWidths[StringGrid1->Col] = 0;
}
//-----

void __fastcall TMDIChild3::ShowallColumns1Click(TObject *Sender)
{ // sets all widths to one value
for (int i=1; i<StringGrid1->ColCount; i++){
    StringGrid1->ColWidths[i] = 50;
}
}
```

5.3.2 unit1.cpp

```
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
#include "table_edit.h"

//-----
#pragma package(smart_init)
#pragma resource "*.dfm"
TDimension *Dimension;
//-----
__fastcall TDimension::TDimension(TComponent * Owner)
: TForm(Owner)
```

5 Anhang

```
{
// just the constructor for the setdimension-form
}
//-----
void __fastcall TDimension::FormShow(TObject *Sender)
{
/*
insert current values as default for the two fields in the form
*/
Edit1->Text = MDIChild3->StringGrid1->RowCount-1;
Edit2->Text = MDIChild3->StringGrid1->ColCount-1;
}
//-----
void __fastcall TDimension::Button1Click(TObject *Sender)
{
/* Function does not accept negative values, sets then row and/or column to 2
(including the first row/col)
*/

float temp;
temp = StrToFloat (Dimension->Edit1->Text)+1;
if (temp<0) temp=2;
MDIChild3->StringGrid1->RowCount = temp;

temp = StrToFloat (Dimension->Edit2->Text)+1;
if (temp<0) temp=2;
MDIChild3->StringGrid1->ColCount = temp;

MDIChild3->StringGrid1Fill (Sender);
// hide the dimension form!
Dimension->Visible = false;

ModalResult = mrOk;
}
```

5.3.3 Output.cpp

```
#include <vcl.h>
#pragma hdrstop

#include "Output.h"
//-----
#pragma package(smart_init)
#pragma link "ChildWin"
#pragma resource "*.dfm"
TMDIChild1 *MDIChild1;
char FileName[MAX_PATH];
//-----
__fastcall TMDIChild1::TMDIChild1(TComponent * Owner)
: TMDIChild(Owner)
{ // Class of Outputwindow

GetModuleFileName (NULL, FileName, sizeof(FileName));
strcpy( strchr(FileName, '\\')+1, "xplora.ini");

Height= GetPrivateProfileInt("Output", "height", 600, FileName);
Width = GetPrivateProfileInt("Output", "width", 500, FileName);
Top= GetPrivateProfileInt("Output", "Top", 50, FileName);
Left= GetPrivateProfileInt("Output", "Left", 25, FileName);

MemoAlignment (MDIChild1);
Position = poDesigned;
```

5 Anhang

```
}  
  
void __fastcall TMDIChild1::MemoAlignment(TObject *Sender)  
{  
    Memo1->Height = ClientHeight;  
    Memo1->Width = ClientWidth;  
}  
//
```

```
void __fastcall TMDIChild1::FormClose(TObject *Sender,  
    TCloseAction &Action)  
{  
    // is called when form is closed!  
}  
}
```

5.3.4 openform.cpp

```
#include <vcl.h>  
#pragma hdrstop  
  
#include "openform.h"  
//  
#pragma package(smart_init)  
#pragma resource "*.dfm"  
  
#include <list.h>  
#include <string>  
#include "table_edit.h"  
TForm2 *Form2;  
long *checkconditions (String , long *);  
//  
__fastcall TForm2::TForm2(TComponent* Owner)  
    : TForm(Owner)  
{  
    // Constructor for Openform  
}  
//  
void __fastcall TForm2::Button1Click(TObject *Sender)  
{  
    // Opendialog chooses the file  
    if (OpenDialog1->Execute()){  
  
        //insert the file-name a c-string to edit field  
        Edit1->Text= OpenDialog1->FileName.c_str();  
  
    }  
}  
//  
  
void __fastcall TForm2::Button2Click(TObject *Sender)  
{  
    // starts the copying of datafile to Stringgrid  
  
    HINSTANCE hDll = LoadLibrary("ReadXplAscii.dll");  
    typedef void __cdeclspec(dllimport) read(char *, char *, string **,  
        double **, long *, string *, long*, long*, long, long);  
    read *function;  
  
    string **strData = new string*;  
    double **dData = new double*;  
    long* rows = new long;
```

5 Anhang

```
string *strTypeOfCols = new string("");
char* sep=NULL,*datei = NULL;
long *selrowsptr = NULL, *selcolsptr = NULL;
long selrowslen = 0, selcolslen=0;

//selection of rows and columns
*rows = 0;
*dData = NULL;
*strData = NULL;

if (hDll)
{
    function = (read*) GetProcAddress(hDll, "_read");
    if (function)
    {
        sep = new char[Edit2->Text.Length()+5];
        strcpy(sep, (Edit2->Text+(CheckBox1->Checked?"_": "")+
        (CheckBox2->Checked?"\t": "")+"\\n").c_str());
        datei = new char[Edit1->Text.Length()+1];
        strcpy(datei, Edit1->Text.c_str());

        selrowsptr = checkconditions(Edit3->Text,&selrowslen);
        selcolsptr = checkconditions(Edit4->Text,&selcolslen);

        function (datei, sep, strData, dData, rows, strTypeOfCols,
        selrowsptr, selcolsptr, selrowslen/2, selcolslen/2);

        delete[] sep;
        delete[] datei;
    }
    else
        ShowMessage("There is no function read in dll!");

    MDIChild3->StringGrid1->ColCount=(*strTypeOfCols).length()+1;
    MDIChild3->StringGrid1->RowCount=(*rows)+1;

    unsigned int j, i, dcounter = 0, strcounter=0;
    for (i=0; i<=(*strTypeOfCols).length(); i++)

    dcounter = strcounter=0;
    for (j=0;j<(*strTypeOfCols).length(); j++)
    {
        if ((*strTypeOfCols)[j]=='d')
        {
            for (i=0;i<(*rows); i++)
                MDIChild3->StringGrid1->Cells[j+1][i+1] =
                ((*dData)+(*rows)*dcounter + i);
            dcounter++;
        }
        else
            for (i=0;i<(*rows); i++)
                MDIChild3->StringGrid1->Cells[j+1][i+1] =
                ((*strData)+(*rows)*(j-dcounter)+i).c_str();
    }

    FreeLibrary(hDll);
} //if hDll
else
    ShowMessage("ReadAscii.dll could not be found");

Form2->Close();
}
//-----
long *checkconditions (String text, long *len)
{
    long *temp = new long[(unsigned int)(text.Length()+1)];
```

5 Anhang

```
long *ret;
std::string number = "";
char *c, ch;
int j = 0, k=0;

*len = 0;

for (int i = 1; i<=text.Length(); i++)
{
    ch = text[i];
    switch (ch){
        case '_':
            if (number != "" && isdigit(text[i+1]) )
                return NULL;
            break;
        case '\t':
            if (number != "" && isdigit(text[i+1]) )
                return NULL;
            break;
        case '~':
            if (number == "")
                return NULL;
            else
            {
                temp[k++]=strtol(number.c_str(),NULL,0);
                number = "";
                break;
            }
        case '|':
            if (number == "")
            {
                return NULL;
            }
            else
            {temp[k++]=strtol(number.c_str(),NULL,0);
                number = "";
                break;
            }
        case '-':
            if (number != "")
                return NULL ;
            else
                number=number.append("-");
            break;
        case '0':
            number=number.append("0");
            break;
        case '1':
            number=number.append("1");
            break;
        case '2':
            number=number.append("2");
            break;
        case '3':
            number=number.append("3");
            break;
        case '4':
            number=number.append("4");
            break;
        case '5':
            number=number.append("5");
            break;
        case '6':
            number=number.append("6");
            break;
        case '7':
```

5 Anhang

```
        number=number.append("7");
        break;
    case '8':
        number=number.append("8");
        break;
    case '9':
        number=number.append("9");
        break;
    default:
        return NULL;
} // switch
} //for

temp[k++]=strtol(number.c_str(),NULL,0);

if (k%2==1)
return NULL;

ret = new long[k];
for (int i = 0;i<k/2;i++)
{
    ret[i] = temp[2*i] ;
    ret[k-1-i] = temp[k-1-2*i];
}

delete [] temp;

*len = k;
return ret;
}
```

5.3.5 mdiapp.cpp

```
#include <vcl.h>
#pragma hdrstop
//-----
USEFORM("main.cpp", MainForm);
USEFORM("ChildWin.cpp", MDIChild);
USEFORM("about.cpp", AboutBox);
USEFORM("Output.cpp", MDIChild1);
USEFORM("Input.cpp", MDIChild2);
USEFORM("table_edit.cpp", MDIChild3);
USEFORM("Unit1.cpp", Dimension);
USEFORM("openform.cpp", Form2);
//-----
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
    Application->Initialize();
    Application->Title = "XploRe";
        Application->CreateForm(__classid(TMainForm), &MainForm);
        Application->CreateForm(__classid(TAboutBox), &AboutBox);
        Application->CreateForm(__classid(TMDIChild1), &MDIChild1);
        Application->CreateForm(__classid(TMDIChild2), &MDIChild2);
        Application->CreateForm(__classid(TMDIChild3), &MDIChild3);
        Application->CreateForm(__classid(TDimension), &Dimension);
        Application->CreateForm(__classid(TForm2), &Form2);
        Application->Run();

    return 0;
}
```

5.3.6 main.cpp

```

#include <vcl.h>
#pragma hdrstop

#include "Main.h"
#include "About.h"
#include "Unit1.h"
#include "openform.h"
//-----
#pragma resource "*.dfm"
TMainForm *MainForm;
char FileName[MAX_PATH];
//-----

__fastcall TMainForm::TMainForm(TComponent *Owner): TForm(Owner)
{
/*
  Constructor of Main project-form
*/

// retrieves the full path and filename for the executable
GetModuleFileName(NULL,FileName,sizeof(FileName));
// cuts exe-name and adds name of ini
strcpy(strchr(FileName,'\\')+1,"xplore.ini");

// gets layout settings from ini-file
Width = GetPrivateProfileInt("Program","Width",800,FileName);
Height= GetPrivateProfileInt("Program","Height",600,FileName);
Top= GetPrivateProfileInt("Program","Top",0,FileName);
Left= GetPrivateProfileInt("Program","Left",0,FileName);

  Application->Restore();
}
//-----

void __fastcall TMainForm::CreateMDIChild(String Name)
{
  TMDIChild *Child;

  //--- create a new MDI child window---
  Child = new TMDIChild(Application);
  Child->Caption = Name;
  if (FileExists(Name))
    Child->Memo1->Lines->LoadFromFile(Name);
}
//-----

void __fastcall TMainForm::FileNew1Execute(TObject *Sender)
{
  CreateMDIChild("Editor" + IntToStr(MDIChildCount));

  // changes menu
  if (MDIChildCount >=2) Menu = MainMenu2;
}
//-----

void __fastcall TMainForm::FileOpen1Execute(TObject *Sender)
{
  if (MDIChildCount >=1) Menu = MainMenu2;
  if (OpenDialog->Execute())
    CreateMDIChild(OpenDialog->FileName);
}
//-----

```

5 Anhang

```
void __fastcall TMainForm::HelpAbout1Execute(TObject *Sender)
{
    // shows the aboutbox in modal form
    AboutBox->ShowModal();
}
//-----

void __fastcall TMainForm::FileExit1Execute(TObject *Sender)
{
    // closes application, original is overwritten later
    Close();
}

void __fastcall TMainForm::CheckMenubar(TObject *Sender)
{
    if (MDIChildCount<3) Menu = MainMenu1;
}
//-----

void __fastcall TMainForm::Positioning(TObject *Sender)
{
    // sets position of mdichilds
    MDIChild1->Left=500;
    MDIChild1->Top =0;
}
//-----

void __fastcall TMainForm::Exit1Click(TObject *Sender)
{
    Close();
}
//-----

void __fastcall TMainForm::FormClose(TObject *Sender, TCloseAction &Action)
{ // this is the overwritten close()-routine
    if (MessageDlg("Close_XploRe?", mtConfirmation, TMsgDlgButtons()
        << mbYes << mbNo,0) == mrYes) {Action = caFree;}
}
//-----

//-----

void __fastcall TMainForm::SetDimensions1Click(TObject *Sender)
{ // if "Set Dimensions" is clicked, this is called
    Dimension -> ShowModal();
}
//-----

void __fastcall TMainForm::Open1Click(TObject *Sender)
{
    // on "open" shows Form2
    Form2->ShowModal();
}
//-----

void __fastcall TMainForm::APSSHelp1Click(TObject *Sender)
{
    // fetches URL of APSS from ini-file
    char www[255];
    DWORD dwhandle = GetPrivateProfileString("APSS", "xpl4nethome",
        "http://www.xploRe-stat.de", www, 255, FileName);
    // ShowMessage(AnsiString(www));
}
```

5 Anhang

```
ShellExecute(Handle, "open",www, NULL, NULL, SW_SHOW);  
}
```

5.3.7 Input.cpp

```
#include <vcl.h>  
#pragma hdrstop  
  
#include "Input.h"  
//-----  
#pragma package(smart_init)  
#pragma link "ChildWin"  
#pragma resource "*.dfm"  
TMDIChild2 *MDIChild2;  
char FileName[MAX_PATH];  
//-----  
__fastcall TMDIChild2::TMDIChild2(TComponent* Owner)  
    : TMDIChild(Owner)  
{ // derived from MDICHILD, this is the inputbox-class  
  
    // fetch dimensions and position from xplore.ini  
    GetModuleFileName(NULL,FileName, sizeof(FileName));  
    strcpy(strchr(FileName, '\\')+1, "xplore.ini");  
  
    Height= GetPrivateProfileInt("input", "height", 600, FileName);  
    Width = GetPrivateProfileInt("input", "width", 500, FileName);  
    Top= GetPrivateProfileInt("input", "Top", 50, FileName);  
    Left= GetPrivateProfileInt("input", "Left", 25, FileName);  
  
}  
//-----  
void __fastcall TMDIChild2::Sizing(TObject *Sender)  
{  
    ListBox1->Width = ClientWidth;  
    ListBox1->Height = ClientHeight - 20;  
    Edit1->Width = ClientWidth;  
    Edit1->Top = ClientHeight - 30;  
}  
//-----  
void __fastcall TMDIChild2::FormClose(TObject *Sender,  
    TCloseAction &Action)  
{  
    // on close!  
}
```

5.3.8 Childwin.cpp

```
#include <vcl.h>  
#pragma hdrstop  
  
#include "main.h"  
#include "ChildWin.h"  
#include "About.h"  
//-----  
#pragma resource "*.dfm"  
//-----  
__fastcall TMDIChild::TMDIChild(TComponent *Owner)  
    : TForm(Owner)  
{ // base class for mdichildren  
}  
//-----
```

5 Anhang

```
void __fastcall TMDIChild::FormClose(TObject *Sender, TCloseAction &Action)
{
    Action = caFree;
}
```

5.3.9 About.cpp

```
#include <vcl.h>
#pragma hdrstop

#include "About.h"
//-----
#pragma resource "*.dfm"
TAboutBox *AboutBox;
//-----
__fastcall TAboutBox::TAboutBox(TComponent *Owner)
    : TForm(Owner)
{
    // constructor for AboutBox Form
}
//-----

void __fastcall TAboutBox::Label1Click(TObject *Sender)
{
    ShellExecute(Handle, "open", "http://localhost", NULL, NULL, SW_SHOW);
}
```

Literaturverzeichnis

- [Arnush 1996] ARNUSH, Craig: *Borland C++ 5 in 21 Tagen*. SAMS, 1996
- [Booch u. a. 2002] BOOCH, Grady ; RUMBAUGH, James ; JACOBSON, Ivar: *Die UML-Kurzreferenz für die Praxis*. Oldenbourg, 2002
- [Eichinger 2001] EICHINGER, Armin: *Usability*.
2001. – URL <http://pcptpp030.psychologie.uni-regensburg.de/student2001/Skripten/Zimmer/u-definition.html>
- [Feuerhake 2002] FEUERHAKE, Jörg: *Optimierung client-/serverbasierte Statistiksyste-me*.
Diplomarbeit. 2002
- [Hollingworth u. a. 2003] HOLLINGWORTH, Jarrad ; SWART, Bob ; CASHMAN, Mark ; GUSTAVSON, Paul: *C++ Builder 6 Developer Guide*. SAMS, 2003
- [Intel 2003] INTEL: *Moore's Law*.
2003. – URL <http://www.intel.com/research/silicon/mooreslaw.htm>
- [Kaiser 2003] KAISER, Richard: *C++ mit dem Borland C++ Builder*. Springer, 2003
- [Leisch 2000] LEISCH, Friedrich: *R: Ein freies Paket für Datenanalyse und statistische Graphik*. January 11 2000. – Invited talk at Universität Dortmund, Germany
- [Louis 1997] LOUIS, Dirk: *Das Borland C++ 5 Kompendium*. Markt und Technik, 1997
- [MathSoft 1999] MathSoft (Veranst.): *S-Plus 2000 User's Guide*. 1999
- [Oestereich 2002] OESTEREICH, Bernd: *Die UML-Kurzreferenz für die Praxis*. Oldenbourg, 2002
- [R Development Core Team 2003] R DEVELOPMENT CORE TEAM: *R Data Import/Export*.
2003. – URL <http://cran.r-project.org/doc/manuals/R-data.pdf>

Erklärung zur Urheberschaft

Hiermit erkläre ich, dass ich die vorliegende Arbeit allein und nur unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Uwe Ziegenhagen
Berlin, 12. September 2003