Neue Funktionen für den Clipper-Compiler: Nantucket Tools II

Helmut Hoffmann

Nachfolgender Beitrag soll sich nicht mit den grundsätzlichen Unterschieden zwischen einem Interpreter und einem Compiler befassen. König hat in seinem an dieser Stelle veröffentlichten Beitrag „CLIPPER – Eine Alternative zu dBASE III Plus?"[1] anhand von dBASE III Plus einerseits und dem Clipper-Compiler andererseits diese Unterschiede sehr genau und zutreffend dargestellt, weshalb hierauf Bezug genommen werden kann. Seine Ausführungen zur Anlegung einer eigenen Rechtsprechungs- und Literatur-Datenbank unter Clipper[2] sollen hier vielmehr weitergeführt werden, und zwar aus Anlaß des Erscheinens der Nantucket Tools II – Funktionen für Clipper –; es geht also in erster Linie um die Verwendung von Funktionen für juristische Anwendungsprogrammierungen. Der Leser, der sich bisher mit anderen Programmiersprachen und noch nicht mit Clipper befaßt hat, wird einen Eindruck davon bekommen, wie weit fortgeschritten die Möglichkeiten von Clipper und der Toolbox sind. Nachdem König sich auf den Datenbank-Bereich konzentriert hat – was naheliegend ist, weil Clipper als Datenbank-Programmiersprache gilt –, sollen hier Clipper und die Nantucket Tools als umfassende Werkzeuge für eine juristische Anwendungsprogrammierung, nämlich für die Entwicklung flexibler Entscheidungsunterstützungs-Systeme mit Datenbank- und Berechnungsmodulen vorgestellt werden.

Wenn es nicht nur um den klassischen Bereich der reinen Datenbank geht, sondern um eine Integration mit den im juristischen Alltag häufig vorzufindenden Berechnungen[3], wird man eine Programmiersprache benötigen, die neben der selbstverständlichen Anforderung der Beherrschung der Grundrechenarten bis hin zur Potenzierung und Wurzelrechnungen folgende Anforderungen erfüllt:

- Unterstützung von Datums-Berechnungen,
- sehr komfortable Fehlerabfang-Möglichkeiten,
- Datenbank-Integration.

Nach der inzwischen gefestigten Überzeugung des Verfassers[4] ist Clipper in Verbindung mit den Tools, wenn professionelles Niveau angestrebt ist, absolut unerreicht. Die Herstellerfirma Nantucket ist längst darüber hinausgewachsen, lediglich einen dBASE-kompatiblen Compiler anzubieten. Sie hat sich vielmehr als besonders innovative Firma herausgestellt, die den Compiler weit über den Sprachumfang von dBASE hinaus entwickelt hat. Man gewinnt als Anwender, wenn man sich das neue dBASE IV anschaut, vielmehr den Eindruck, daß die gegenwärtig aktuelle Clipper-Version „Sommer 87" der erst 1989 mit offenbar zahlreichen Fehlern erschienenen neuen dBASE-Version nach wie vor weit überlegen ist. Einzelheiten hierzu – allerdings lediglich als Vergleich Clipper einerseits und dBASE III Plus andererseits – hat König bereits dargestellt und sollen hier nicht wiederholt werden. Hingewiesen sei allerdings darauf, daß auch dBASE IV bei etwa gleichem Preis wie Clipper nach wie vor keine selbständig ablauffähigen EXE-Files erzeugt. Der von König erwähnte prinzipielle Nachteil von Interpretern[5], daß etwaige Syntaxfehler erst in der praktischen Anwendung festgestellt werden[6], vermeidet natürlich der Compiler, weil jedes Programm auf richtige Syntax überprüft wird. dBASE IV überprüft die Syntax im Gegensatz zu dBASE III jetzt auch, weil neu geschriebene oder veränderte Programme kompiliert werden.

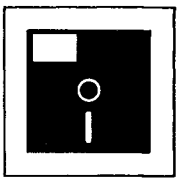
Herausragend bei Clipper ist wohl nicht so sehr der Befehls-Umfang, sondern einerseits die Textverwaltung in MEMO-Feldern[7], in erster Linie aber die Funktionen. Der Sprachumfang von Clipper ist gegenüber dBASE insofern erweitert. Die Tools fügen zahlreiche teils außerordentlich leistungsfähige Funktionen hinzu. Außerdem gibt Clipper dem Programmierer die Möglichkeit, eigene Funktionen zu definieren, was eine große Hilfe sein kann, weil ein Programmiersprachen-Entwickler nicht an jede spezielle Anwendersituation denken und diese berücksichtigen kann.

Syntax und Bedeutung von Funktionen

Clipper-Funktionen bauen auf der allgemeinen Syntax auf: <Funktionsname> <(Ausdruck)>, wobei allerdings in der runden Klammer nicht stets ein Argument zur Funktion stehen muß. Bekanntestes Beispiel: DATE() liest das aktuelle Systemdatum ein; wollen Sie hieraus jedoch den Tag, den

Helmut Hoffmann ist Richter am Amtsgericht Ulm

1 Jur-PC 1989, 23.
2 In Erwiderung zur Darstellung des Verfassers in IuR 1988, 253 ff., 310 ff.
3 Man denke an Zinsberechnungen und überhaupt den finanzmathematischen Bereich sowie Unfallberechnungen, Prozeßkostenberechnungen usw.
4 Deren Richtigkeit im streng wissenschaftlichen Sinne natürlich nicht in ihrer Richtigkeit „beweisbar" sein kann.
5 A.a.O. Seite 24.
6 Man denke bei stark verschachtelten IF- oder CASE-Schleifen daran, daß ein schließendes ENDIF oder ENDCASE vergessen wurde.
7 Vgl. hierzu König a.a.O. Seite 26 einerseits, Verfasser IuR 1988, 253, 258 andererseits.



Monat als Ziffer bzw. als Wort oder das Jahr extrahieren, benötigen Sie Funktionen, die das Datum als Ausdruck angeben, aus dem der Teil extrahiert werden soll. Am Beispiel des aktuellen Datums 01.09.1989:

```
HEUTE = DATE()
Ergebnis: 01.09.89
TAG = DAY(HEUTE)
Ergebnis: 1
MONAT = MONTH(HEUTE)
Ergebnis: 9
MONAT = CMONTH(HEUTE)
Ergebnis: September
JAHR = YEAR(HEUTE)
Ergebnis: 1989
```

Verschachtelungen von Funktionen sind möglich:

```
MONAT = CMONTH(DATE())
Ergebnis: September
```

Die Beispiele zeigen, daß das Ergebnis von Funktionen in eine Variable gespeichert werden kann. Der Typ der Variablen muß nicht mit dem Typ des Ausdrucks übereinstimmen: In obigen Beispielen ist der Ausdruck stets ein Datum gewesen, das Ergebnis teilweise eine numerische Variable, bei CMONTH dagegen eine String-Variable. Die meisten Funktionen erwarten einen Ausdruck eines speziellen Typs. Es gibt aber auch Funktionen, die beliebige Ausdruck-Typen verarbeiten können. Das nebenstehende kleine Beispielprogramm zur Erläuterung von Clipper-Funktionen innerhalb einer Anwendungsumgebung[8] benutzt die Funktion EMPTY(). Diese Funktion versteht als Ausdruck sowohl Strings wie numerische, Datums- und sogar logische Werte. Die Funktion ermöglicht es also, beliebige Eingabefelder oder auch Datenfelder dahingehend zu überprüfen, ob etwas eingetragen ist. Die Funktion ist also sehr flexibel. Oben wurde schon erläutert, daß Funktionen in aller Regel Werte zurückgeben.

Wer sich das Beispielprogramm anschaut, wird bemerken, daß hier offenbar weder ein String noch eine Zahl als Ergebnis der Funktion EMPTY() zurückgegeben wird, sondern ein logisches „Wahr" oder „Falsch". Die Programmzeile „IF EMPTY(datum)" prüft also ab, ob das Eingabefeld datum mit Zeichen belegt ist oder nicht, und gibt ein logisches Wahr oder Falsch zurück, das mit einer gängigen IF-Schleife abgefragt werden kann.

Es ist völlig ausgeschlossen und auch nicht Sinn dieses Beitrags, auch nur annähernd alle Funktionen des Clipper-Sprachumfangs und der Tools II zu erläutern. Erwähnt sei hier nur, daß einige sehr leistungsfähige Funktionen nicht nur einen Ausdruck verarbeiten können, sondern eine Vielzahl von Argumenten. So kann die Funktion MEMOEDIT(), die dazu bestimmt ist, ein MEMO zu editieren oder auch nur anzuzeigen, nicht weniger als 13 Argumente verarbeiten.

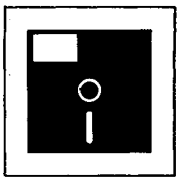
Offenbar ausgehend davon, daß die Funktionen eine wesentliche Stärke von Clipper darstellen und im Wettbewerb mit dBASE und sogenannten dBASE-Clones ein wichtiges Verkaufsargument sein können, entwickelte die Herstellerfirma seit 1987 eine Toolbox, und zwar nicht etwa in den USA, sondern in der deutschen Filiale in Köln. Die zweite Version ist soeben erschienen und liefert den aktuellen Anlaß für diesen Beitrag. Dem Käufer fällt zunächst rein optisch auf, daß die Beschreibung statt in einem Paperback nunmehr in einem stabilen Ringordner ähnlich dem Handbuch vorliegt und wesentlich umfangreicher geworden ist. Die Umstellung ist sehr zu begrüßen, weil man sich wie bei Handbüchern eigene Zettel mit privaten Anmerkungen, Beispiel-Listings usw. leicht einheften kann. Das Ganze ist auch wesentlich stabiler als das arg schnell zerfledderte Paperback. Als Zweites fällt sofort auf, daß jetzt drei und nicht nur eine Diskette beiliegen. Die eigentlichen Funktionen sind jedoch auf einer Diskette abgespeichert. Sinn und Inhalt der beiden anderen Disketten werden weiter unten beschrieben.

Mit den Tools II liegen nunmehr über 500 Clipper-Funktionen vor, die im Handbuch in Funktions-Gruppen übersichtlich aufgeteilt und in ihrer Syntax meist ausführlich dargestellt sind. Das Handbuch geht – realistischerweise – davon aus, daß der Käufer kein absoluter Anfänger ist, sondern das Prinzip der Clipper-Funktionen bereits kennt und deshalb Syntax-Beschreibungen nachvollziehen kann. Als Hilfe für den Käufer hat Nantucket auf der Diskette übersichtliche Beispiels-Programme für eine Reihe von Funktionen im Quellentext zur Verfügung gestellt, die den Zugang zur Syntax und den Sinn der Funktion zu verstehen erleichtern können. Erfreulicherweise sind auch eine Reihe von benutzerdefinierten Funktionen[9] im Quellentext als Beispiele dafür wiedergegeben, wie der Programmierer Funktionen selbst erstellen und in seine Entwicklungen integrieren kann. Wer Erfahrungen mit der Programmiersprache C hat, wird Ähnlichkeit des Programmierstils feststellen.

Der Programmierer, der kein „Vollprofi" mit vollständiger Beherrschung der verwendeten Programmiersprache ist, sieht sich oft mit der Situation konfrontiert, nicht zu wissen, ob für ein bestimmtes Problem eine Funktion vorhanden ist oder nicht. Bei über 500 verschiedenen Funktionen ist das Problem auch mit einem alphabetischen Inhaltsverzeichnis schwer lösbar. Nantucket hat sich deshalb erfreulicherweise entschlossen, ein Volltext-Datenbankprogramm auf zwei Disketten den

8 Berechnung von Zeiträumen.

9 UDFs - User Defined Functions. Die Möglichkeit, eigene Funktionen zu definieren und in die Programme einzubinden, stellt einen der wesentlichen Vorteile von Clipper dar.



Tools II beizufügen. Dies stellt einen großen Gewinn für den praktischen Einsatz dar, zeigt aber auch, wie außerordentlich leistungsfähig Clipper als Datenbankprogramm ist. Das Retrieval-Programm ist selbstverständlich in Clipper geschrieben. Die Benutzeroberfläche ist in mancher Hinsicht sehr ähnlich derjenigen der NJW-Leitsatzkartei auf CD-ROM.[10] Man wird damit eigentlich sofort zurecht kommen. Die Geschwindigkeit der Suche im Volltext ist außerordentlich groß, was daran liegt, daß jedes sinntragende Wort in den verwendeten MEMO-Feldern indexiert worden ist.[11]

Weitere Funktionen

Hier können nicht alle Funktionen erläutert, sondern nur einige für den juristischen Anwendungsprogrammierer möglicherweise besonders interessante in ihren Grundzügen exemplarisch vorgestellt werden. Die Tools II stellen umfangreiche Funktionen zur Fenstertechnik zur Verfügung. Der Pfiff liegt darin, daß mit den Cursortasten, PageUp, PageDown, Home und End die so erzeugten Fenster auf dem Bildschirm und sogar aus dem Bildschirm heraus verschoben werden können. Auf Tastendruck wird das Fenster wieder an die Ausgangsposition zurückgeschoben. Das Beispielprogramm zeigt eine kleine Anwendung der Fenstertechnik in der Funktion Help. Es kann nicht etwa nur ein Fenster, sondern es können bis zu 255 Fenster gleichzeitig verwaltet und mit einer Funktion per Tastendruck aufgerufen werden. Cursor-Positionen können innerhalb der einzelnen Fenster relativ zu den Begrenzungen des Fensters definiert werden, so daß ein Text in einem Fenster, wenn dieses bewegt wird, sich zusammen mit dem Rahmen des Fensters auf dem Bildschirm bewegt. Bewegt man das Fenster, so wird ein eventuell „darunter" stehender Text wieder sichtbar. Die mit Hilfe der Tools II erzeugten Fenster können Bildschirme bis 255 Zeilen mal 255 Spalten unterstützen, auch den 50-Zeilen-Modus des VGA-Standards und die im Bürobereich hin und wieder eingesetzten Ganzseiten-Bildschirme.

Diese Funktionen sind wie sämtliche anderen der Toolbox in Assembler geschrieben. Es dürfte wohl kaum einen juristischen Anwendungsprogrammierer geben, der in der Lage wäre, solche Funktionen selbst zu erstellen. Die Window-Funktionen werden von den Treiber-Funktionen ergänzt. Clipper entwickelt sich zusammen mit den Tools zu einer Programmiersprache, die starke Fortschritte hinsichtlich der Graphikfähigkeit macht, wie auch die Funktionsgruppe Video zeigt. Die unterschiedlichsten Graphikkarten – auch VGA-Graphik – werden jetzt unterstützt, auch der 43-Zeilen-Modus der EGA-Karte. Funktionen ermöglichen die Überprüfung, ob eine EGA- oder VGA-Karte installiert ist, so daß mit IF-Schleifen die unterschiedlichste Hardware individuell mit ihren jeweiligen Möglichkeiten eingesetzt werden kann. Zum Clipper-Sprachumfang gehört demgegenüber nur die Funktion ISCOLOR(), die überprüft, ob überhaupt eine Farbgraphik vorhanden ist – vgl. die Funktionen COLOR1() und COLOR2() des bereits mehrfach erwähnten beiliegend abgedruckten Beispielprogramms. Am Ende des Listings sehen Sie zunächst die benutzerdefinierte Funktion COLOR1(), die in einer IF-Schleife abfängt, ob ein Farbsystem installiert ist. Die Funktion COLOR3() überprüft wesentlich weitergehend mit Hilfe einer Reihe von Funktionen aus den Tools II ganz genau, welche Grafik-Karte vorhanden ist bzw. emuliert werden kann. Man kann hieran erkennen, daß benutzerdefinierte Funktionen mit solchen des Clipper-Sprachstandards und der Tools beliebig vermischt und ineinander verschachtelt werden können.

Eine eigene Funktionsgruppe der Tools II befaßt sich mit der Kommunikation über die serielle Schnittstelle. Bis zu vier serielle Schnittstellen können gleichzeitig angesteuert werden. Für den Datenbank-Bereich äußerst leistungsfähige Stringfünktionen schließen sich an. Der Leser hat den Eindruck, daß geradezu alle String-Manipulationen hiermit realisiert werden können. Hierzu gehören auch Funktionen, die das Vorhandensein bestimmter Zeichen durch den gesamten String überprüfen. Eine praktische Anwendung liegt darin, die DOS-Trennzeichen wie „:." aufzufinden und so eine Zeichenkette, die z.B. den vollen Pfad eines Dateinamens angibt, in ihre Einzelteile zu zerlegen und auf Richtigkeit zu überprüfen. Eine Richtigkeits-Überprüfung ist im Gegensatz zu den Tools I jetzt auch mit der Funktion FILEVALID() dahingehend möglich, ob ein vom Benutzer vergebener Dateiname unter DOS zulässig ist oder nicht.

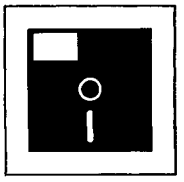
Viele Tips zur Verwendung der Funktionen findet man in den Dateien LIES.DAS und vor allem TOOLDEMO.PRG, die sich auf der Diskette befinden und die man sich sinnvollerweise gleich nach dem Erwerb der Tools ausdruckt.

Eine Reihe von Druckfunktionen erlauben es, die Anzahl der vorhandenen Druckerschnittstellen festzustellen und einen Druckbefehl auf einen beliebigen Drucker zu geben.

Wie oben sowie anhand des abgedruckten Beispielprogramms bereits ersichtlich, unterstützt Clipper das korrekte Rechnen mit dem Datum, und zwar nach den Regeln der Gregorianischen Kalenders über die Jahrhundert-Grenzen hinweg.

10 Vgl. hierzu Hoffmann NJW-CoR 4/89, 10 sowie Jur-PC 1989, 195.

11 Zu den Vor- und Nachteilen der Indexierung in Volltext-Retrieval-Programmen vgl. Herberger Jur-PC 1989, 192, 193.



Die Tools liefern zusätzliche Möglichkeiten: So kann mit EOY() das Jahresende, mit ISLEAP() ein Schaltjahr (beide Funktionen sind im Beispielprogramm verwendet), mit entsprechenden Funktionen das Ende eines Monats oder Quartals festgestellt und in eine Variable gespeichert werden. Die Umwandlung von beliebigen Zeiträumen in Sekunden ist mit einer neuen Funktion möglich. Die Funktion TIMEVALID() überprüft, ob eine Variable eine gültige Zeitangabe darstellt. Hiermit kann man also Fehleingaben bei Uhrzeiten ebenso gut abfangen wie es bei Datumswerten bereits mit dem Clipper-Sprachumfang möglich ist.
Jeder dBASE- oder Clipper-Programmierer kennt die zahlreichen SET-Befehle. Die Tools erlauben es, den aktuellen Status jedes SET-Befehls in eine Variable zu speichern und dann nach dem Lauf eines anderen Programms den früheren Zustand wiederherzustellen. Dies wird man beispielsweise bei den Hilfefunktionen einsetzen. Wenn die Clipper-eigene Hilfe-Funktion über die Taste F1 ausgelöst wird, wird man Hilfetexte stets in einer Standard-Farbe anzeigen und den Cursor ausschalten. Am Anfang eines Hilfe-Programms wird man deshalb die aktuellen Farbeinstellungen und den Cursor-Status sichern und nach Ende des Hilfe-Programms den alten Status wiederherstellen, was dem Programm einen ausgesprochen professionellen Anstrich gibt. Diese Möglichkeiten werden im Beispielsprogramm im Rahmen der Funktion HELP exemplarisch vorgestellt. Angemerkt sei, daß man die Hilfe-Funktionen in einem „richtigen" Anwenderprogramm wesentlich komfortabler und besser programmieren wird; hier soll nur das Prinzip anhand eines einfachen Beispiels vorgestellt werden.

Mathematische Funktionen

Daß Clipper weit über eine Datenbank-Programmiersprache hinausgeht, zeigt der Umstand, daß viele Funktionen im mathematischen Bereich vorhanden sind, um einerseits auch komplexere mathematische Berechnungen vornehmen zu können, andererseits aber auch die Leistungsfähigkeit der installierten Hardware hierbei auszunutzen. So kann sogar festgestellt werden, ob ein mathematischer Coprozessor installiert ist, ebenso wie zahlreiche andere System-Informationen. Allerdings sei zur Vorsicht bei der Benutzung von Funktionen geraten, die finanzmathematische Operationen betreffen. Denn die Funktionen legen nicht offen, nach welchen Formeln und aufgrund welcher rechtlichen Grundlage die Berechnung z.B. eines Jahreszinses oder einer Annuität erfolgt.
Ein Clou für den fortgeschrittenen Clipper-Progammierer stellt die jetzt erstmals in den Tools II vorhandene Möglichkeit dar, verschachtelte GET zu verwenden. So kann jetzt erstmals innerhalb eines GET-Feldes über eine Funktionstaste ein anderes Programm mit GET-Feldern aufgerufen werden. Dies führt nicht mehr, wie es bisher zwangsläufig der Fall war, zu Fehlern, sondern die neuen Funktionen sichern die aktiven GET und restaurieren sie später wieder, sobald die Hilfe-Prozedur abgelaufen ist.
Angesichts dieses Leistungsumfangs ist sicherlich plastisch geworden, daß es bei Clipper mitnichten „nur" um Datenbank-Applikationen geht. Vielmehr ist nach der Überzeugung des Verfassers Clipper in Verbindung mit den Tools II das geradezu ideale Werkzeug zur Entwicklung beliebiger Anwendungsprogramme, die sich dadurch auszeichnen, daß sie sowohl eine professionelle Benutzeroberfläche beinhalten als auch aus einer Kombination von Rechenwerk einerseits und Datenverwaltung andererseits bestehen auch und gerade für Juristen. Solche Informationssysteme in dieser Kombination sind das, was der Jurist in der Praxis neben der reinen Datenbank immer wieder benötigt, während jedoch die entsprechenden Programmentwicklungen oft erst am Anfang stehen.
Mit Clipper und den Tools ist ein Programm-Niveau auch für den Programmierer, der nicht vollprofessionell Programme entwickelt, mit vertretbarem Lern- und Entwicklungsaufwand darzustellen, das nach dem gegenwärtigen Stand der Programmiersprachen-Entwicklung mit keinem anderen System erreichbar sein wird.
Das alles hat natürlich auch seinen Preis: Clipper kostet 2.395 DM, die Tools 1.590 DM, jeweils unverbindliche Preisempfehlungen zuzüglich Mehrwertsteuer. Wer nicht nur kleine Gelegenheits-Programme für sich selbst entwickelt, sondern ein höheres Niveau erreichen will, für den ist dies gut angelegtes Geld; insbesondere wenn man die Preise damit vergleicht, daß die Entwickler-Version von dBASE IV ca. 3.800 DM kostet und dort sehr viel fehlt, was Clipper kann.
Nach neuesten Informationen (PC Welt 9/89) wird der innovative Vorsprung bald ausgebaut werden: Im Herbst soll in den USA die Version 5.0 von Clipper erscheinen. Die heute bekannten wesentlichsten Weiterentwicklungen: Die Tendenz, von dBASE im Sprachumfang abzukommen und den Vorsprung zu vergrößern, verstärkt sich. Durch einen neuen Linker werden Overlays auch in großen Anwendungen überflüssig. Es wird – einem aktuellen Modetrend im Sprachenmarkt folgend – eine „objektorientierte" Version geben, die unter DOS, Windows und OS/2 sowie auf dem Macintosh läuft. Ein Datenaustausch zwischen allen dBASE-kompatiblen und Structured-Query-Language-Datenbanken wird möglich. Neue Aufgaben für juristische Anwendungsprogrammierer und neue Tests für Jur-PC sind deshalb zu erwarten.