TUSTEP-Version Oktober 1995: Neue Steuerbefehle für Editor-Makros

Wie in der letzten BI (95/11+12) S. 10 angekündigt, werden hier anhand von Anwendungsbeispielen die Neuerungen der TUSTEP–Version Oktober 1995 vorgestellt, die die Editor-Makros betreffen. Die genaue Beschreibung der neuen Steuerbefehle für die Editor-Makros findet man in den »Ergänzungen zum TUSTEP–Handbuch 1993« (Version Oktober 1995). Sie sind, wie in der letzten BI S. 8 beschrieben, allgemein zugänglich.

Ausdrucken des Bildschirminhalts im Editor

Ein Ausdrucken des Dateiinhalts war bisher nur über DRUCK-VORBEREITE möglich. Damit bekommt man zwar die Daten der Datei ausgedruckt, aber nicht den angezeigten Bildschirminhalt. Mit dem neuen Steuerbefehl `PRINT` ist es möglich, in der Funktion einer Hardcopy den Bildschirminhalt (ggf. mit Hervorhebungen, Fehlermeldungen, Makroleisten, Anweisungen etc.) in das Zweitprotokoll zu schreiben, um anschließend dieses Zweitprotokoll weiterzubearbeiten, z. B. auszudrucken, anzuschauen, zu kopieren, in eine andere Datei einzubinden etc.

Der Steuerbefehl `PRINT` kann mit der Tastenkombination `<Strg>+<P>` aufgerufen werden. Die alte Bedeutung der Tastenkombination, die Cursorposition anzuzeigen, steht nun ausschließlich über den neuen Steuerbefehl `SHW_CUR_POS` zur Verfügung.

Beispiel:
Der Steuerbefehl `PRINT` erzeugt keine Meldung, daß das Schreiben ins Zweitprotokoll erfolgt ist. Eingebunden in ein Editor-Makro kann eine Rückmeldung gegeben werden.

```
y,?mld=”PRINT ausgeführt!”
y,d=print, switch:abcxyz?mld
```

Nach Ausführung des Steuerbefehls `PRINT` schreibt das Editor-Makro *D* den mit `Y,?MLD` definierten Text in die Anweisungszeile und wartet auf eine Tasteneingabe. Die Meldung verschwindet bei der nächsten Tasteneingabe, und da es kein Editor-Makro mit der angegebenen Namensbasis *ABCXYZ* gibt, ist nichts weiter geschehen und der alte Zustand des Bildschirms ist wieder da.

Weitere Anwendung:
`PRINT` kann man verwenden, um beim strukturierten Suchen mit den Anweisungen *SA* etc. sich am Bildschirm angezeigte Fundstellen abzuspeichern. Mit `PRINT` kann man den angezeigten Text ins Zweitprotokoll kopieren, ohne daß dabei die Suche unterbrochen wird. Nach Beenden der Suche stehen alle ausgewählten Stellen im Zweitprotokoll zur Verfügung.

Merken und Verändern der *-Position

Im TUSTEP–Editor wird jeweils der zuletzt bearbeitete Datensatz als *-Position gemerkt, und man kann in Anweisungen durch einen * als Kurzform auf diese Stelle referieren. Mit den neuen Steuerbefehlen kann die *-Position auch dazu verwendet werden, sich eine Stelle in der Datei zu merken, um – nachdem man z. B. eine andere Stelle dieser Datei oder eine andere Datei im Editor bearbeitet hat – wieder an diese Stelle zurückzukehren, ohne daß die Satznummer dafür eingetippt werden muß. Die drei neuen Steuerbefehle sind:

`SAVE_REC_NR`	*-Position merken
`REST_REC_NR`	gemerkte *-Position setzen
`EXCH_REC_NR`	aktuelle und gemerkte *-Position tauschen

Beispiel:
Für die täglich Arbeit ist es hilfreich, sich zur Benutzung dieser Steuerbefehle drei Editor-Makros zu definieren:

```
y,m=save_rec_nr
y,p=confirm, rest_rec_nr, ”zu*”,
  enter
y,x=confirm, exch_rec_nr, ”zu*”,
  enter
```

Das Editor-Makro *M* merkt die aktuelle *-Position. Das Editor-Makro *P* setzt die gemerkte *-Position und zeigt diese Stelle der Datei an. Das Editor-Makro *X* macht beides gleichzeitig: Es merkt die aktuelle *-Position und zeigt die Stelle um die zuvor gemerkte *-Position in der Datei an.

Verändern der *-Position mit dem Cursor

Normalerweise ist die *-Position der zuletzt angezeigte bzw. der zuletzt bearbeitete Satz. Mit dem neuen Steuerbefehl `XFER_REC_NR` kann der Satz an der aktuellen Cursorposition als *-Position definiert und anschließend in Anweisungen verwendet werden.

Beispiel:
Eine bestimmte Stelle in der Datei soll durch Positionieren des Cursors ausgewählt werden.

```
y,k=xfer_rec_nr, save_rec_nr,
  confirm, ″db″, enter,
  rest_rec_nr, ″ka,*″, enter
```

In einer Datei namens *A* wählt man einen Satz durch Positionieren des Cursors aus und startet das Editor-Makro *K*. Dieses merkt die *-Position, wechselt in eine Datei namens *B* und kopiert den in der Datei *A* ausgewählten Satz ans Ende der Datei *B*.

Cursor zum Eintragen positionieren

Ist man im Eintragemodus (nach der Anweisung `EE`) und will mit Editor-Makros neuen Text eintragen (z. B. Teile von Eingabemasken), so ist es schwierig, die letzte Textzeile zu finden, um dahinter den neuen Text einzutragen. Der neue Steuerbefehl `NEXT_LINE` bietet die Leistung, den Cursor im Eintragemodus an den Anfang der Zeile unter der letzten Textzeile zu positionieren und ggf. den vorhandenen Text dazu um eine Zeile nach oben zu schieben.

Ein Beispiel für `NEXT_LINE` steht im nächsten Abschnitt.

Umdefinieren der Return-Taste

Neben den bisherigen Modi *ENTER* und *SPLIT* gibt es für die Return-Taste eine weitere Bedeutung, den *MACRO*–Modus. Eingestellt werden können die Modi mit den Steuerbefehlen `CHG_CR` und `RST_CR`.

Im *MACRO*–Modus ruft die Return-Taste ein Editor-Makro auf. Steht der Cursor im Textfeld, wird das Editor-Makros *CR* (ggf. ergänzt um den aktuellen Leistennamen) aufgerufen; steht der Cursor in der Anweisungszeile, ruft die Return-Taste das Editor-Makro namens `CMD` (ggf. ergänzt um den aktuellen Leistennamen).

Beispiel:
Es soll ein Eintragen mit einer Eingabemaske ermöglicht werden. Die Feldkennungen bestehen aus einer laufenden Nummer. Um das Eintragen von längeren Feldern zu erleichtern, werden die Feldkennungen je einzeln durch die Return-Taste auf den Bildschirm geschrieben.

```
y,e=rst_cr, 2*chg_cr, 2*enter, ″1″,
  cur_le, rd_num, del, ″ee″, enter,
  lf, ″<″, wr_num, ″>″
y,cr=inc_num, next_line, ″<″,
  wr_num, ″>″
y,cmd=switch:cc_?frage
y,?frage=″Beenden / Weiter ?″
y,cc_b=2*enter, rst_cr
y,cc_w=switch:e
```

Mit dem Editor-Makro *E* beginnt das Eintragen. Für die Return-Taste wird der *MACRO*–Modus eingestellt; die laufende Nummer wird auf 1 gesetzt. Das Eintragen wird begonnen und die erste Feldkennung `<1>` hingeschrieben. Man trägt nun den Text für dieses Feld ein, schließt es durch Betätigen der Return-Taste ab und löst damit das Editor-Makro *CR* aus. Dieses Editor-Makro erhöht die laufende Nummer und schreibt in die Zeile hinter der letzten Textzeile (`NEXT_LINE`; dieser Steuerbefehl ist hier wichtig, da der Cursor z. B. infolge von Korrigieren weiter oben im Text stehen kann, hier aber nun unbedingt die letzte Textzeile gefunden werden muß) die neue Feldkennung. Um das Eintragen eines Datensatzes zu beenden, stellt man den Cursor in die Anweisungszeile und betätigt dann die Return-Taste, wodurch das Editor-Makro `CMD` ausgelöst wird. Dies stellt die mit `Y,?FRAGE` definierte Frage und wartet auf eine Eingabe. Durch die Eingabe `B` wird das Eintragen beendet, und die Bedeutung der Return-Taste wird in den *ENTER*–Modus zurückgestellt; die Eingabe von `W` ruft das Editor-Makro *E* zum Eintragen einer neuen Maske.

Makrokontrolle bei erfolgloser Suche

In TUSTEP gibt es mehrere Möglichkeiten, Zeichenfolgen im Editor zu suchen: mit den Zeige- und Such-Anweisungen in der ganzen Datei oder mit den Steurbefehlen `JMP_UP`, `JMP_DN` oder `MRK_FND` innerhalb eines Bildschirms. Alle diese Suchen können in Editor-Makros eingebunden werden. Dabei muß unterschieden werden können, ob eine Zeichenfolge gefunden wurde oder nicht, damit die Editor-Makros auf sinnvolle Weise die Kontrolle behalten können.

Der Steuerbefehl `NO_MATCH:makroname` kann innerhalb der Editor-Makros hinter eine Suche geschrieben werden. Wird nichts gefunden, wird zum angegebenen Editor-Makro verzweigt. Wird eine Zeichenfolge gefunden, so wird `NO_MATCH:makroname` ignoriert und das Editor-Makro hinter diesem Steuerbefehl fortgesetzt.

Ein Beispiel für `NO_MATCH:makroname` steht im nächsten Abschnitt.

Zeicheneingabe unter Makrokontrolle

Für manche Anwendungen ist es sinnvoll, nur bestimmte Operationen im Editor zuzulassen. Sollen dabei auch Text oder Anweisungen eingegeben werden können, mußte dazu bisher die Kontrolle des Programms abgegeben werden. Der neue Steuerbefehl `CHAR` bietet in Kombination mit dem Steuerbefehl `SWITCH:xx?` die Möglichkeit, die Kontrolle vollständig bei den Editor-Makros zu belassen und trotzdem die Eingabe von Text zu ermöglichen.

Der Steuerbefehl `SWITCH:xxx?` unterbricht die Abarbeitung eines Editor-Makros und wartet auf genau eine Tasteneingabe, die zusammen mit der Basis `xxx` als Name eines Editor-Makros interpretiert wird, das dann aufgerufen wird. In diesem Zustand des Wartens ist die Kontrolle über den Editor beim Editor-Makro, denn die Tasteneingabe wird in jedem Fall zunächst vom Editor-Makro weiterverarbeitet. An der Stelle setzen die neuen Steuerbefehle an. Der Steuerbefehl `CHAR` gibt die Möglichkeit, die erfolgte Tasteneingabe – sofern es sich um ein ASCII–Zeichen der Schreibtastatur handelt – im Ablauf eines folgenden Editor-Makros auf den Bildschirm zu schreiben. Damit man nun nicht für alle Zeichen ein Editor-Makro definieren muß, steht für ASCII–Tasteneingabe die allgemeine Namensergänzung *CHAR* zur Verfügung, die eingesetzt wird, falls kein Editor-Makro mit dem speziellen Namen (Namensbasis + Tastenname) definiert ist. Eine kontrollierte Texteingabe sieht im Minimalfall so aus:

```
y,s=switch:s_?
y,s_char=char, switch:s_?
```

Mit dem Editor-Makro *S* wird das Eingeben gestartet. Das Editor-Makro wartet dann auf eine Tasteneingabe. Wird ein ASCII–Zeichen eingegeben, so wird das Editor-Makro `S_CHAR` ausgeführt. Dieses schreibt durch den Steuerbefehl `CHAR` das eingegebene Zeichen auf den Bildschirm und wartet anschließend erneut auf eine Tasteneingabe. Werden Sonderzeichen eingegeben, so werden Editor-Makros mit dem entsprechenden Namen für das Sonderzeichen gesucht, z. B. `S_DEL` bei Betätigen der `<ENTF>`–Taste. Im vorliegenden Minimalbeispiel führt die Eingabe von Sondertasten zum Abbruch des Editor-Makros, da keine entsprechenden Editor-Makros definiert sind. Um das Verfahren abzusichern, ist hinter `SWITCH:S_?` der Steuerbefehl `RETRY` anzubringen und dahinter ein kontrollierter Ausgang.

Beispiel:
Ermöglicht werden soll die interaktive Eingabe einer Suchzeichenfolge, die anschließend im Text gesucht wird. Das initialisierende Editor-Makro *S* kann man sich dabei in ein größeres System von Editor-Makros eingebunden denken.

```
y,s=clear, "Bitte Suchzeichenfolge
  eingeben: 〉", ignore, switch:za_?,
  5*retry
y,za_char=switch:zf_char
y,zf_char=char, switch:zf_?, 5*retry
y,za_tilde=switch:za_?til, 5*retry
y,zf_tilde=switch:zf_?til, 5*retry
y,?til="Tilde ist nicht erlaubt!!!"
y,zf_bsp=mrk_ini, cur_pos:1;35,
  mrk_del_del, switch:zf_?
y,zf_enter=save_cur, cur_pos:1;35,
  mrk_ini, rest_cur, mrk_rep, clear,
  "zu,,,~", mrk_ins, "~", enter,
  no_match:su_end, switch:su_?next,
  5*retry
y,zf_cr=switch:zf_enter
y,su_w=enter, no_match:su_end,
  switch:su_?next, 5*retry
y,?next="Treffer! Weitersuchen
  oder Beenden? W / B ?"
y,su_b=switch:su_?neu, 5*retry
y,?neu="Neue Suche: J / N ?"
y,su_end=switch:su_?ende, 5*retry
y,?ende="Nichts mehr gefunden!
  Neue Suche: J / N ?"
y,su_j=switch:s
y,su_n=clear, "b", enter
```

Das Editor-Makro *S* schreibt auf den Bildschirm einen Eingabeprompt für die Suchzeichenfolge und wartet auf eine Eingabe, die durch die Namensbasis *ZA_* ergänzt wird, so daß bei Eingabe eines ASCII–Zeichens das Editor-Makro *ZA_CHAR* aufgerufen wird (hinter `RETRY` sollten noch Steuerbefehle zur Absicherung folgen).

ZA_CHAR wird nur für die Eingabe des ersten Zeichens verwendet, um das Absenden einer leeren Suchzeichenfolge zu verhindern. Es ruft das Editor-Makro *ZF_CHAR* auf, mit dem die eingegebenen Zeichen je auf den Bildschirm geschrieben werden. Neben den ASCII–Zeichen ist die Verwendung folgender Sondertasten vorgesehen:

Die Tilde führt zum Editor-Makro *ZF_TILDE*, das die Eingabe dieses Zeichens verbietet, da die Tilde als Begrenzungszeichen gebraucht wird und deswegen in der Suchzeichenfolge nicht vorkommen darf.

Mit der Backspace-Taste (Editor-Makro *ZF_BSP*) wird die eingegebene Suchzeichenfolge wieder gelöscht. Der Eingabeprompt

bleibt dabei erhalten. (Position 1;35 ist die Stelle hinter dem Prompt).

Mit der Enter- oder der Return-Taste wird die Eingabe abgeschlossen und die Suche gestartet (Editor-Makro *ZF_ENTER* bzw. *ZF_CR*). Die Suchzeichenfolge wird am Bildschirm durch Markieren abgeholt, wobei das Markieren an der bekannten Position 1;35 beginnt und bis zur aktuell erreichten Position geht. Danach wird die Zeige-Anweisung für die Suche zusammengebaut, die abgeholte Suchzeichenfolge zwischen die Begrenzungszeichen eingesetzt und die Suche gestartet. Der Steuerbefehl `NO_MATCH:su_end` gibt an, daß das Editor-Makro *SU_END* aufgerufen wird, falls keine Zeichenfolge gefunden wird. Wird eine Zeichenfolge gefunden, so wird sie angezeigt und gleichzeitig die mit `Y,?NEXT` definierte Meldungszeile mit der Namensbasis *SU_* aufgerufen. Damit bleibt auch während der Suche die Kontrolle bei den Editor-Makros, und das Anzeigen der nächsten Zeichenfolge erfolgt durch Eingabe von `W` (ruft Editor-Makro *SU_W*). Alternativ kann man die Suche beenden, was letztlich durch das Editor-Makro *SU_N* zum Beenden des Editors führt.

Dieses vollständige, aber immer noch einfache Beispiel soll die Möglichkeiten andeuten, um im Editor ein geschlossenes System zu realisieren, das z. B. auch zur Datenerfassung mit Masken eingesetzt werden kann.

Bei den vorgestellten Beispielen muß man die beiden typischen Einsatzgebiete von Editor-Makros im Blick behalten: Zum einen dienen Editor-Makros dazu, sich selbst die Arbeit zu erleichtern, wiederkehrende Arbeitsabläufe zu vereinfachen und manche Leistung erst zugänglich zu machen. Zum anderen kann man mit Editor-Makros begrenzte Anwendungen so zur Verfügung stellen, daß sie ohne Einarbeitung von anderen sicher bedient und beherrscht werden können.

Winfried Bader
bader@zdv.uni-tuebingen.de

BI 96/1 + 2, S. 14–17