DIE D S L - SOFTWARETECHNIK -

DARGESTELLT ANHAND EINER STEUERUNGS-AUFGABE

Hans-Jürgen Ehling

Berlin

AEG-TELEFUNKEN
Zentralabteilung Software-Technologie

Juli 1980

EINLEITUNG

Die DSL-Softwaretechnik ist ein Verfahren für systematische Software-Entwicklung, basierend auf

- einem Systembeschreibungs-Schema,
- einer Entwurfssprache: DSL -- design language --,
- einer Entwicklungs-Methodik
- sowie Entwicklungs-Hilfsmitteln.

Auf der Suche nach einem Vorgehen, das zu verständlichen und durchsichtigen Systemen führt, hat sich gezeigt, daß alle genannten Komponenten dazugehören.

Die DSL-Technik wird nachfolgend anhand eines Beispiels umrissen. Voranzustellen sind einige generelle Bemerkungen.

Ein durchgängiges methodisches Prinzip ist dasjenige der Zweckorientierung. Zum Beispiel bei der Auswahl der Grundbegriffe: soll ein Software-System als 'Programm' - also als Text - vorgestellt werden, ist ein System eine Summe von 'Moduln' (welcher Definition ?), sind die Bausteine Daten mit Operationen (woher diese kennen ?) ? Nach dem Z w e c k fragend, wird klar, daß ein Software-System

ein erwünschtes Verhalten in seiner Umgebung erzeugen soll.

Deshalb ist eine 'Einheit für Verhaltensbeschreibung' - 'Prozeß' genannt - der Grundbaustein des System-Beschreibungsschemas; die Betrachtung ist - 'menschgerecht' - auf zusammenhängende Vorgänge (in realer Zeit) gerichtet statt auf punktuell darin eingestreute 'Laufzeiten'. Wir stellen uns folglich Prozesse wie Personen, Systeme wie Organisationen vor.

Der Mensch ist ein optisches Wesen; die DSL-Sprache hat deshalb einen graphischen Einschlag, auf den keine etablierte Technik verzichtet. Natürlich muß die Grafik in der entscheidenden 'Bleistift- und Radiergummi-Phase' schnell produzierbar sein, und natürlich muß sie später mit wenig Aufwand maschinell erzeugbar und wartbar sein. Letzteres ist deshalb die erste Leistung des DSL-Entwicklungssystems, für dessen

Output das nachstehende Bild ein Beispiel ist.

Die Zweckorientierung hat beim Entwurf die Form des 'Output-Rückwärts'-Prinzips. Die Umgebung wird durch Outputs beeinflußt; von ihnen aus arbeitet man sich rückwärts zu den verfügbaren Inputs durch.

Für die Realisierung von DSL-Prozessen eignet sich die ADA-Rendezvoustechnik hervorragend [8]. Ein Betriebssystem mit Rendezvoustechnik für Mikroprozessoren ist deshalb als ein weiteres Hilfsmittel in Entwicklung.

Inspiriert wurde die DSL-Technik vor allem durch die M. Jackson-Entwurfstechniken [1] und durch die von J. D. Warnier [2] und K. Orr [3] vorgeschlagenen Darstellungsformen. Die Bedeutung der Zweckorientierung hat vor allem K. Orr verdeutlicht.

Klammerdiagramm, erstellt mit DSL-Entwicklungssystem.

ERLÄUTERUNGEN ZUM AUFBAU

Zur Darstellung: Als Diskussionsgrundlage dient eine einfache Steuerungsaufgabe, deren Bearbeitung im Umriß dargestellt wird. Dabei werden die nötigen sprachlichen und methodischen Elemente eingebracht. - Zur Verbindung mit der Gesamtkonzeption wird das Beispiel umrahmt von einer 'Kurzdarstellung' der DSL-Technik, deren Themen in einem 'Rückblick' am Ende des Beispiels reflektiert werden.

Die Darstellung erfolgt also in drei Stufen:

1) Eine "Kurzdarstellung" - auf dem nachfolgenden Blatt - beschreibt in Stichworten Bestandteile und Prinzipien der DSL-Technik.
2) Anschließend zeigt das "DSL-Beispiel: Steuergerät", wie die DSL-Mittel beschaffen sind und angewendet werden.
3) Am Ende des Beispiels ergänzt ein "Rückblick" einige Themen aus der "Kurzdarstellung" anhand der Entwurfs-Ergebnisse.

Zum Entwurfs-Beispiel: es behandelt die Steuerung eines "Walzgerüstes". Die Aufgabe ist, ein codierbares Modell für das "Steuergerät" des Gerüstes zu entwerfen. Sie wird auf zwei Ebenen der Sprachverwendung behandelt:

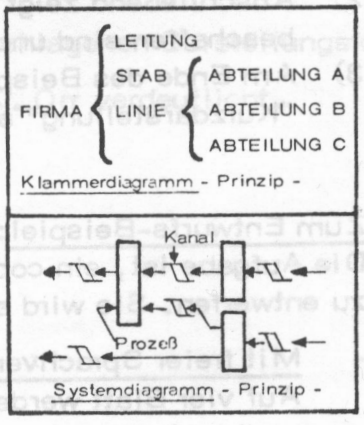
- Mit freier Sprachverwendung, ähnlich "Pseudocode".
  Auf vier Blatt werden betrachtet:
  die Aufgabenstellung - Seite 5,
  die Aufbaustruktur - Seite 6,) Typische Stufen im
  eine Ablaufstruktur - Seite 7,) Entwurfsgang
  die Gesamtstruktur - Seite 8.
  Der Entwurf ist - in den Grenzen dieser Sprachverwendung - auf Seite 8 abgeschlossen. Im "Rückblick" stehen weitere Angaben zu den Themen der Seiten 6 und 7.
- Mit strikter Sprachverwendung. Hier werden nochmals dargestellt:
  die Aufbaustruktur - Seite 9,
  - mit vollständiger Beschreibung des Synchronisationsverhaltens,
  eine Ablaufstruktur - Seite 10,
  - mit einer Code-Skizze.

Die DSL-SOFTWARETECHNIK besteht aus:
- einem System-Schema,
- einer Entwurfssprache: DSL - design language -,
- einer Entwicklungsmethodik - und aus
- dem DSL-Entwicklungs-System.

Das SYSTEM-SCHEMA ist ein generelles Schema für "System-Modelle":
- Ein System-Modell besteht aus "Schichten", zuoberst die "Nutz-Schicht".
- Eine Schicht besteht aus "Prozessen", die über "Kanäle" kommunizieren.
  - Ein Prozeß ist ein "Individuum", das "auf Lebenszeit" für bestimmte Output-Produktionen verantwortlich ist.
  - Ein Kanal ist eine "Rohrpost"; er verbindet und synchronisiert Prozesse.
- Untere Schichten interpretieren obere Schichten, z. B. Kanalzugriffe.

Die ENTWURFS-SPRACHE DSL beschreibt abstrakte System-Modelle.
- DSL ist eine formale Sprache. Als Hauptform benutzt sie
- Klammerdiagramme. Für diese gilt:
  - sie zeigen hierarchische Struktur als Form,
  - sie beschreiben Aufbau, Abläufe und Daten,
  - man kann sie auch in freier Form verwenden.

  Daneben werden
- Systemdiagramme gebraucht:
  - sie zeigen Netze aus Prozessen und Kanälen,
  - sie sind Anschauungshilfen; ihre Information steht auch in Klammerdiagrammen,
  - in freier Form - ohne Kanäle - werden sie zu "Blockdiagrammen".

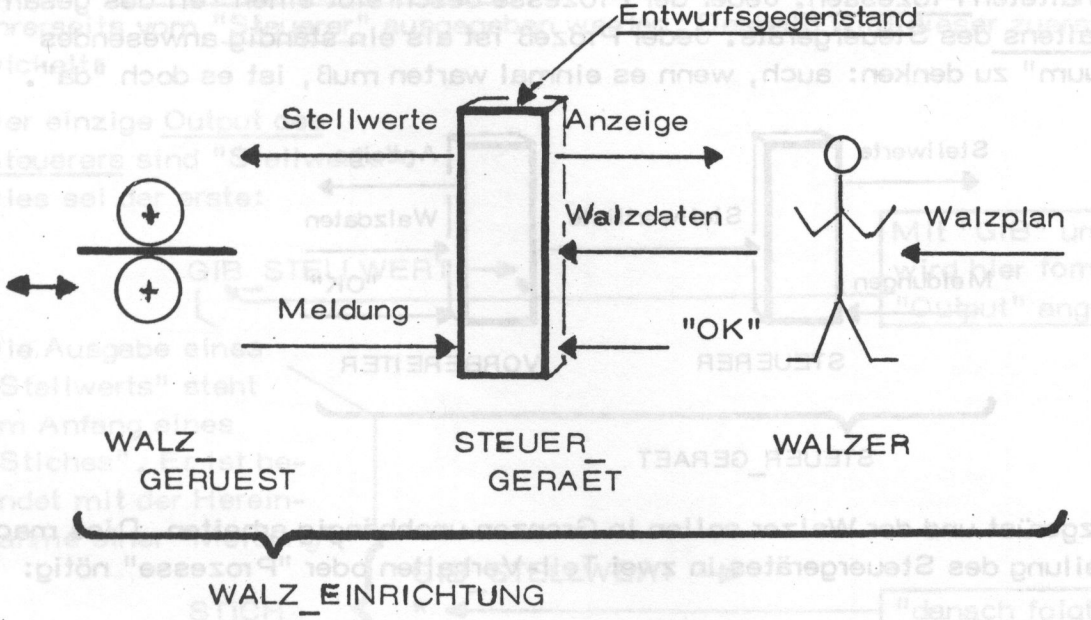


Die ENTWICKLUNGS-METHODIK geht aus von:
- der Entwurfsmethodik. In dieser gelten zuerst:
  - Zielorientierung und
  - Datenorientierung. Die Maxime ist: "Vom Output rückwärts schreiten"
  - Daraus hervor geht oft: die Daten s t r o m-Orientierung (M.A. Jackson).
  - "Entwurf" verläuft von der Nutzschicht in die "Sub"-Schichten, jeweils vom Output zum Input.
- Die Zielfindung (Anforderungsanalyse) ist ebenfalls "Entwurf", nur gröber.
- Die Realisierung hält sich an das Modell. Zuerst sind
  - die Repräsentationen zu bestimmen; dann wird das Modell "übersetzt".
  - Die Testdaten werden aus Input/Output-Datenströmen abgeleitet.
  - "Realisierung" verläuft "invers" zum Entwurf: von den Subschichten in die Nutzschicht, jeweils vom Input zum Output.

Das DSL-ENTWICKLUNGS-SYSTEM [1]:
- hält und führt System-Modelle,
- erzeugt - projektbegleitend - Dokumente und
- unterstützt die Programm-Erzeugung.

1) in Entwicklung befindlich.

Dieses Bild zeigt eine "Walzeinrichtung", für die ein "Steuergerät" entworfen werden soll:



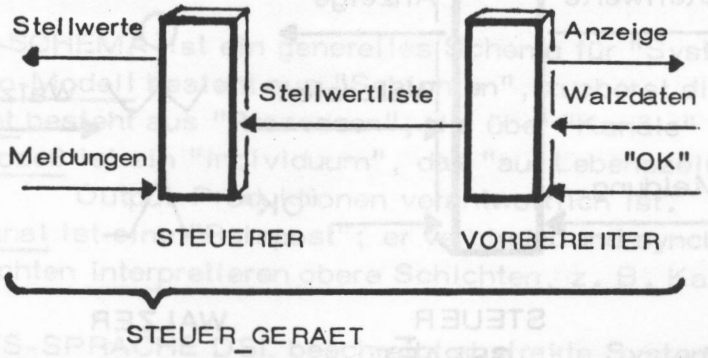
Hier folgen nähere Angaben:

- Das WALZGERUEST walzt ein Materialstück in 5 Durchgängen - "Stiche" genannt -, die abwechselnd hin- und hergerichtet sind.
- Ein Stich wird begonnen bei Eintreffen eines Stellwerts. Bei seinem Ende gibt das Walzgerüst eine Meldung.
- Ein Bedienungsmann, der WALZER, gibt die Walzdaten vor, u. a. die Endabmessungen des gewalzten Materialstücks.
- Das STEUERGERAET setzt die Walzdaten um in eine Folge von Stellwerten, die es nacheinander an das Walzgerüst weitergibt.
- Dies geschieht jedoch erst, nachdem der Walzer eine Anzeige der errechneten Stellwertfolge begutachtet und sein "OK" gegeben hat. - Erkennt der Walzer einen Irrtum, gibt er anstelle des "OK" neue Walzdaten ein.
- Der Walzer kann im Verlauf einer Walzung die Daten für die nächste vorbereiten; aber nicht mehr!

Das Bild zeigt die "Walzeinrichtung" als ein System aus mehreren, ständig dazugehörigen "Teilen"; diese sind durch Signale miteinander verbunden. Auch, wenn z. B. der "Walzer" einmal nichts zu tun hat, gehört er - wartend - doch dazu.

Ähnlich wird zunächst im folgenden Bild das "Steuergerät" betrachtet.

Dieses Blockdiagramm zeigt den Aufbau des "Steuergeräts" aus zwei hintereinander geschalteten Prozessen. Jeder der Prozesse beschreibt einen Teil des gesamten Verhaltens des Steuergeräts. Jeder Prozeß ist als ein ständig anwesendes "Individuum" zu denken: auch, wenn es einmal warten muß, ist es doch "da".

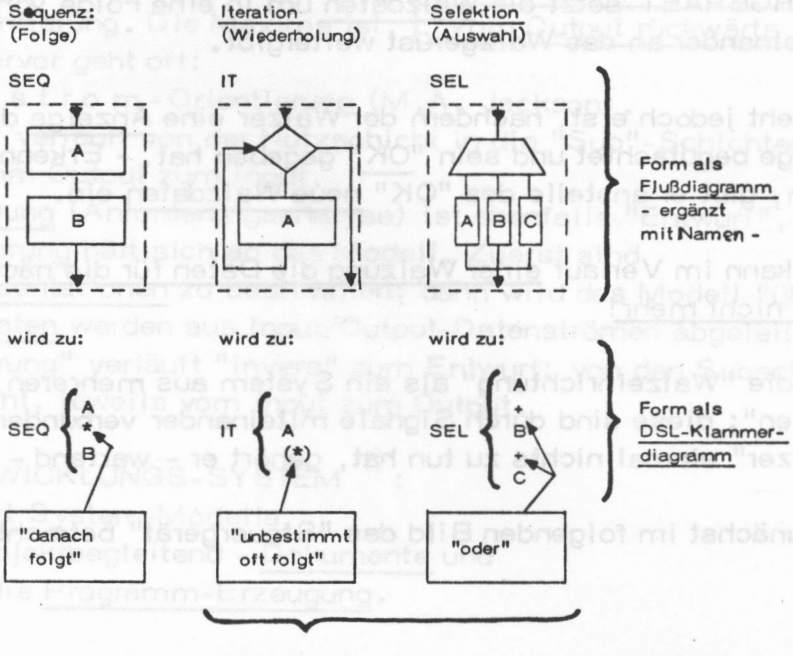


Das Walzgerüst und der Walzer sollen in Grenzen unabhängig arbeiten. Dies macht die Aufteilung des Steuergerätes in zwei Teil-Verhalten oder "Prozesse" nötig:

- den STEUERER, der das WALZGERUEST betreut,
- den VORBEREITER, der mit dem WALZER zusammenarbeitet.

Das Bindeglied zwischen "Vorbereiter" und "Steuerer" sind die "Stellwertliste(n)".

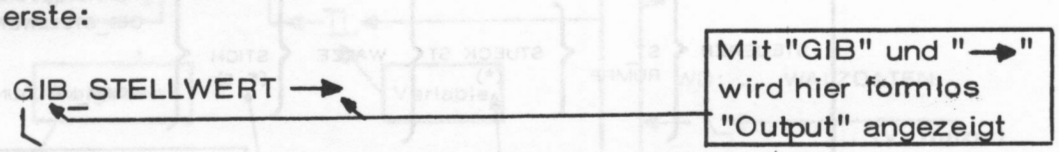
Ein hinreichend einfacher Prozeß kann sequentiell, als schrittweiser Ablauf, beschrieben werden. Wir nehmen an, daß dies für obige Prozesse zutrifft. In den folgenden Bildern werden dann DSL-Klammerdiagramme auftreten, mit denen auch Abläufe beschrieben werden. Dabei treten folgende Formen auf, die nachstehend ihren Entsprechungen in Flußdiagrammen gegenübergestellt werden:



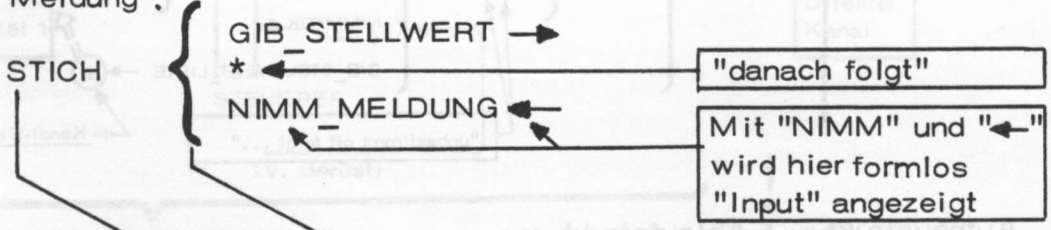
Bedingungen kann man angeben; sie werden bei Bedarf eingeführt werden.

Wir nehmen an, daß sich "Steuerer" und "Vorbereiter" als Abläufe beschreiben lassen. Vom Walzgerüst rückwärts treffen wir zunächst auf die "Stellwerte", die ihrerseits vom "Steuerer" ausgegeben werden. Deshalb wird dieser zuerst entwickelt:

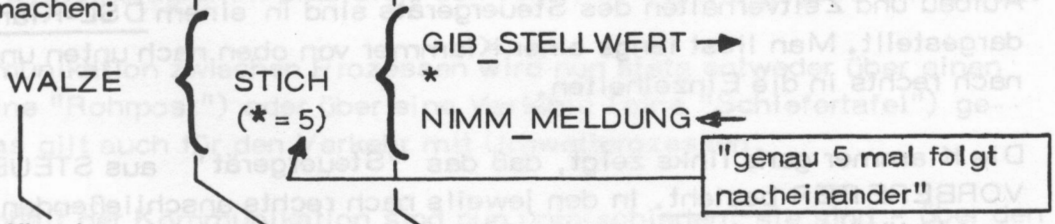
- Der einzige Output des Steuerers sind "Stellwerte". Dies sei der erste:



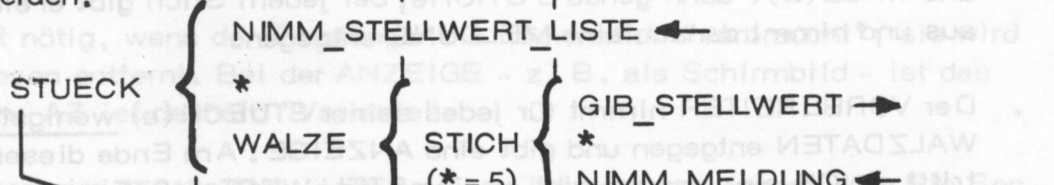
- Die Ausgabe eines "Stellwerts" steht am Anfang eines "Stiches". Er ist beendet mit der Hereinnahme einer "Meldung":



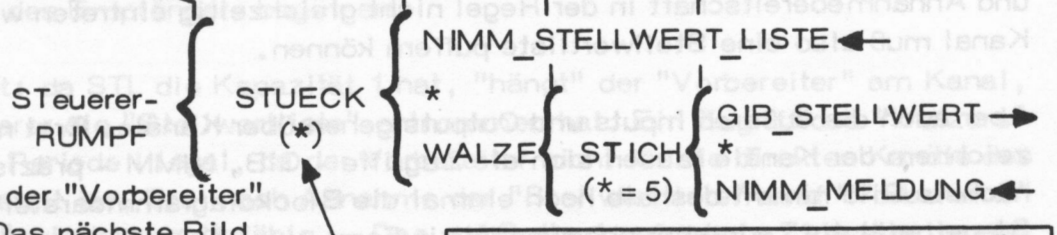
- Zu "Walzen" heißt, nacheinander 5 Stiche zu machen:



- "Walze" gilt für ein "Stück"; zuvor muß eine "Stellwert-Liste" empfangen werden:

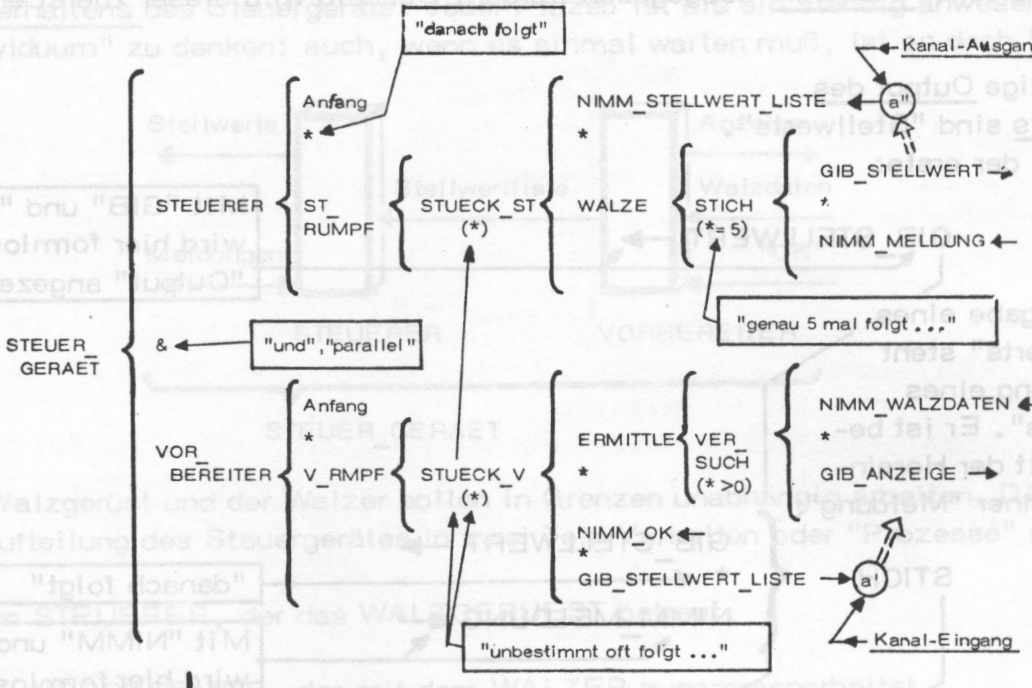


- "Stücke" werden unbestimmt viele gewalzt:



Ähnlich wird der "Vorbereiter" entwickelt. Das nächste Bild zeigt beide zusammen:

Das nachstehende DSL-Klammerdiagramm beschreibt die Arbeitsweise des "Steuergeräts" mit Aufbau- und Ablaufstrukturen:



Aufbaustruktur | Ablaufstrukturen

Aufbau und Zeitverhalten des Steuergeräts sind in einem DSL-Klammerdiagramm dargestellt. Man liest längs einer Klammer von oben nach unten und geht jeweils nach rechts in die Einzelheiten.

Die Klammer ganz links zeigt, daß das "Steuergerät" aus STEUERER und ("&") VORBEREITER besteht. In den jeweils nach rechts anschließenden Bild-Teilen sind Abläufe dargestellt, erkennbar am "danach folgt" ("*"). Diese Teile besagen:

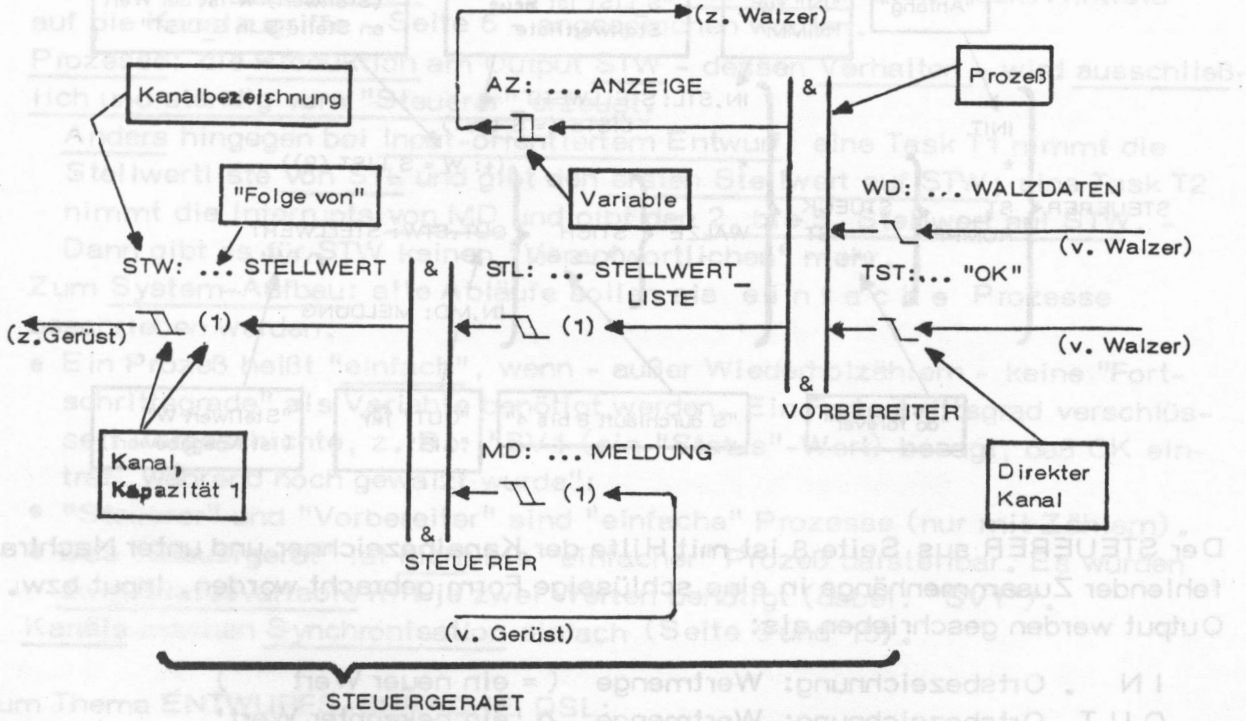
- Der STEUERER nimmt für jedes seiner STUECKe eine STELLWERT_LISTE und WALZ(E)t dann genau 5 STICHe; bei jedem Stich gibt er einen STELLWERT aus und nimmt danach eine MELDUNG entgegen.

- Der VORBEREITER nimmt für jedes seiner STUECK(e) wenigstens einmal WALZDATEN entgegen und gibt eine ANZEIGE. Am Ende dieser VERSUCH(e) trifft ein OK ein; danach gibt er eine STELLWERT_LISTE aus.

Die Übergabe der Stellwertliste erfolgt über einen "Kanal", da Ausgabe- und Annahmebereitschaft in der Regel nicht gleichzeitig eintreten werden. Dieser Kanal muß also eine Stellwertliste puffern können.

Aber auch die äußeren Inputs und Outputs gehen über Kanäle. Erst mit den Bezeichnern der Kanäle lassen sich die Zugriffe - GIB, NIMM - präzisieren. Das nächste Bild nimmt deshalb noch einmal die Blockdiagrammdarstellung für das Steuergerät auf, nun aber abgewandelt in Form eines DSL-Systemdiagramms:

Dieses Bild zeigt die Aufbaustruktur des "Steuergeräts" in der Form eines DSL-Systemdiagramms. Das ist ein Netzwerk aus Prozessen und Kanälen:



Dieses DSL-Systemdiagramm präzisiert den schon in Seite 6 gegebenen Aufbau:

- Die Kommunikation zwischen Prozessen wird nun stets entweder über einen Kanal (eine "Rohrpost") oder über eine Variable (eine "Schiefertafel") geleitet. Das gilt auch für den Verkehr mit Umweltprozessen.

- Ort und Inhalt der Kommunikation sind nun unterschieden; sie sind - über den Symbolen - dargestellt in der Form:

  Ortsbezeichnung: Wertmengen - Folge.

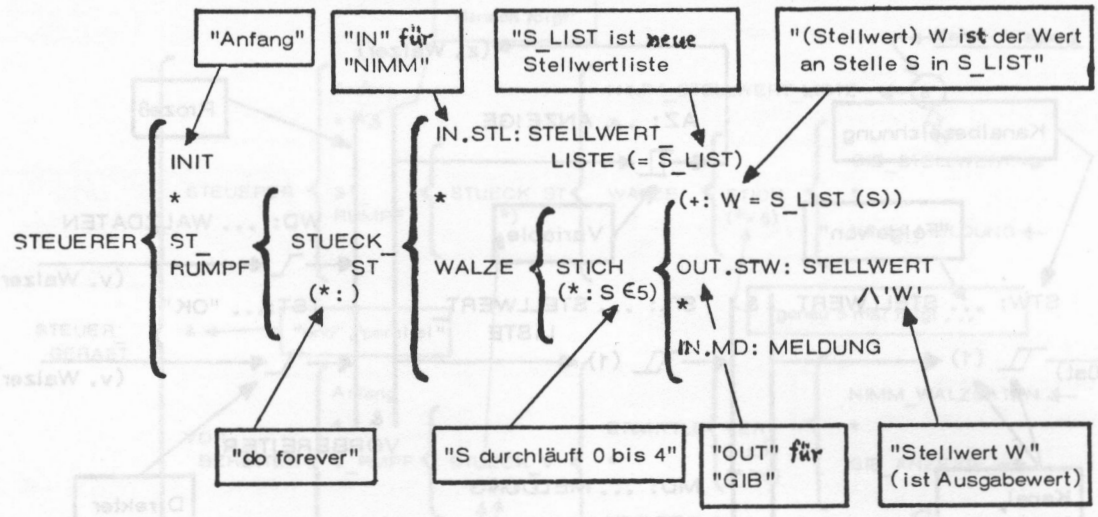
  Kanäle und Variable "befinden" sich an abstrakten "Orten"

Ein Kanal ist nötig, wenn der Empfänger die Information "verbraucht"; sie wird dann beim Lesen entfernt. Bei der ANZEIGE - z. B. als Schirmbild - ist das nicht möglich; AZ ist deshalb "Variable".

Ein Empfänger wird angehalten, wenn er einen leeren Kanal lesen will. Ein Sender wird angehalten, wenn nach seinem Eintrag der Kanal voll ist. - Ein "direkter Kanal" erzwingt alternierenden Betrieb der gekoppelten Prozesse; dieser muß mit Bereitschaft des Empfängers beginnen.

Das bedeutet: da STL die Kapazität 1 hat, "hängt" der "Vorbereiter" am Kanal, bis der Steuerer die "Stellwertliste" entnommen hat. Ein Zugriff des "Walzers" ist in dieser Periode illegal, da der Vorbereiter nicht für die direkten Kanäle des "Walzers" bereit ist. Erst nach Abnahme der "Stellwertliste" sind "Vorbereiter" und "Walzer" wieder arbeitsfähig. - Das auf Seite 1 geforderte Zeitverhalten ist damit erfüllt.

Dieses Bild zeigt ein formalisiertes DSL-Klammerdiagramm für den "Steuerer", nämlich seine "Prozeßbeschreibung"; sie schließt den Entwurf ab:



Der STEUERER aus Seite 8 ist mit Hilfe der Kanalbezeichner und unter Nachtrag fehlender Zusammenhänge in eine schlüssige Form gebracht worden. Input bzw. Output werden geschrieben als:

I N . Ortsbezeichnung: Wertmenge (= ein neuer Wert)
O U T. Ortsbezeichnung: Wertmenge /\ 'ein bekannter Wert'

Weil die Wertmenge in der Nähe steht, ist die Bedeutung eines Wertes überall schnell festzustellen.

Der Entwurf ist mit der Prozeßbeschreibung beendet. Für die anschließende Codierung sind Repräsentationen für Wertmengen und Kanäle zu wählen. Ein Code für eine sehr einfache Wahl ist nachstehend gegeben:

Code mit Strukturnamen als Kommentar - Prinzip -

```
/* STEUERER */
   /* Initialisierung, hier: leer */
   /* ST_RUMPF */
STEUR: REPEAT
      S_LIST: = EMPFANG (STL)
      /* WALZE */
      FOR S FROM 1 TO 5 REPEAT
         /* STICH */
         WRITE (STW) S_LIST (S)
         CALL EMPFANG (MD)
            /* kein "Datum" vom Interrupt MD */
         /* STICH-Ende */ END
      /* WALZE-Ende */
      END
   /* ST_RUMPF-Ende */
/* STEUERER-Ende */
```

Code ohne Kommentar - Prinzip -

```
STEUR: REPEAT
   S_LIST: = EMPFANG (STL)
   FOR S FROM 1 TO 5 REPEAT
      WRITE (STW) S_LIST (S)
      CALL EMPFANG (MD)
   END
END
```

Kanalzugriff - Prinzip -
(für Kanalkapazität 1; beide Semaphore mit 0 initialisiert)

```
EMPFANG:
   REQUEST  Semaphor-Empfang-STL
   S_LIST: = Pufferplatz-von-STL
   RELEASE  Semaphor-Senden-STL

SENDEN:
   Pufferplatz-von-STL: = "aktueller Wert"
   RELEASE  Semaphor-Empfang-STL
   REQUEST  Semaphor-Senden-STL
```

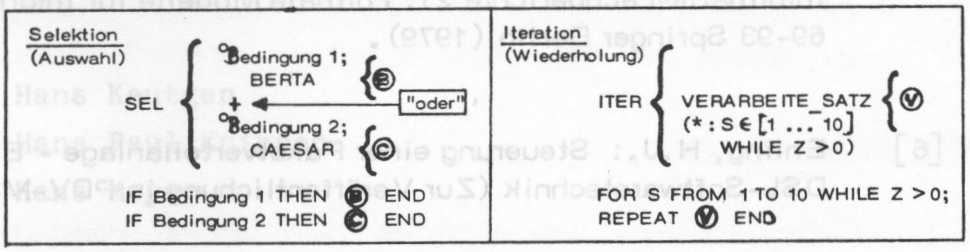
Rückblickend einige Hinweise zu Themen der DSL-Softwaretechnik:

Zum Thema SYSTEM-SCHEMA:

- Es wurde die Nutzschicht behandelt. Eine Subschicht ist mit dem Hinweis auf die Kanalzugriffe - Seite 6 - angesprochen worden.
- Prozesse: die Produktion am Output STW - dessen Verhalten - wird ausschließlich und ständig vom "Steuerer" erzeugt.
  Anders hingegen bei Input-orientiertem Entwurf: eine Task T1 nimmt die Stellwertliste von STL und gibt den ersten Stellwert auf STW; eine Task T2 nimmt die Interrupts von MD und gibt den 2. bis 5. Stellwert auf STW. - Dann gibt es für STW keinen "Verantwortlichen" mehr.
- Zum System-Aufbau: alle Abläufe sollen als e i n f a c h e Prozesse beschrieben werden.
  - Ein Prozeß heißt "einfach", wenn - außer Wiederholzählern - keine "Fortschrittsgrade" als Variable benötigt werden. Ein Fortschrittsgrad verschlüsselt Vorgeschichte, z. B.: "SV1 (ein "Status"-Wert) besagt, daß OK eintraf, während noch gewalzt wurde":
  - "Steuerer" und "Vorbereiter" sind "einfache" Prozesse (nur mit Zählern).
  - Das "Steuergerät" ist nicht als "einfacher" Prozeß darstellbar. Es würden zwei Statusvariable mit je zwei Werten benötigt (dabei: "SV1").
- Kanäle machen Synchronisation einfach (Seite 9 und 10).

Zum Thema ENTWURFS-SPRACHE DSL:

- Klammerdiagramme erschließen die E i n p r ä g s a m k e i t graphischer Form für abstrakte Texte. Man kann sie schnell schreiben und ändern.
  - Die strikte Form erlaubt exakte Beschreibungen jenseits von Programmiersprachen. Die informelle Schreibweise wird schnell verstanden.
  - Beispiele zur Schreibweise von Bedingungen in Klammerdiagrammen.



Zum Thema ENTWICKLUNGS-METHODIK:

- Eine Schicht wird vom Output her rückwärts mit größtmöglichen einfachen Prozessen aufgebaut. - Würden Vorbereiten und Steuern sich abwechseln, wäre das "Steuergerät" insgesamt ein einfacher Prozeß!
- Ein Prozeß wird von seinen Outputdaten her rückwärts aufgebaut, wobei jeweils die gerade benötigten Inputs hinzugenommen werden.
  - Funktionen werden dann zwischen ihren Output und Input eingefügt.
- Es wurde ein sehr einfaches Modell entworfen. Dies könnte zum Ergebnis einer Zielfindung (Anforderungsanalyse) gehören.

Zum Thema DSL-ENTWICKLUNGS-SYSTEM:

- Klammerdiagramme können bereits mechanisch erstellt werden; ihre Textanordnung wird errechnet. [1)]

1) weitere Leistungen in Entwicklung.

LITERATUR

[1] Jackson, M.A.: Principles of Program Design.
Academic Press, London (1975).

[2] Warnier, J.D.: Logical Construction of Programs.
Stenfort Kroese BV (1974).

[3] Orr, K.: Structured Systems Development.
New York (1977).

[4] Ehling, H.J.: Datenstrukturierter Software-Entwurf für Echtzeitsysteme.
PDV-Berichte: Verfahren und Hilfsmittel für Spezifikation und Entwurf von Prozeßautomatisierungssystemen, Kernforschungszentrum Karlsruhe, KfK-PDV 154 6 (1978).

[5] Ehling, H.J.: Evolutionärer System-Entwurf.
Informatik-Fachberichte 21: Formale Modelle für Informationssysteme 69-93 Springer Berlin (1979).

[6] Ehling, H.J.: Steuerung einer Paketverteilanlage - Ein Entwurf in der DSL-Softwaretechnik (Zur Veröffentlichung in PDV-Mitteilung).

[7] Ehling, H.J.: Kurze Einführung in die Entwurfssprache DSL.
AEG/Interner Bericht Z5612 SB 020/79.

[8] Ichbiah, Heliard u. a.: Rationale for the Design of the ADA Programming Language.
acm Sigplan Notices 14 6 (1979) Part B.