Inhalt

1. Zweck der Schrift

Das System.014 ist ein Softwarekomplex, der die umfassende Einfuehrung rechentechnischer Mittel zur Rationalisierung von Verwaltungs- und Produktionsprozesssen entscheidend unterstuetzen soll.
Im folgenden wird ein Einblick in die Funktionsweise dieses Systems gegeben.
Dabei geht es zunaechst darum, wie des System.014 im weiten Gebiet der Betriebs- und Anwendungssoftware einzuordnen ist. Weiterhin werden die grundlegenden theoretischen Voraussetzungen und Prinzipien erlaeutert. Schliesslich werden die wesentlichen Komponenten des System.014 in ihrer Funktionsweise dargestellt.
In dieser Schrift wird nicht ein fertiges System in seinen funktionellen Einzelheiten beschrieben, sondern es wird das Vorhaben, ein solches System zu schaffen, ueberblicksmaessig skizziert, wobei Einzelheiten von Implementierungen nicht betrachtet werden und wo auch nicht jener Grad an Systematik erwartet werden kann, der fuer die umfassende Beschreibung eines umfangreichen Softwarekomplexes an sich notwendig ist.
Diese Darstellung ist die zweite von 3 Schriften, in denen die Vorstellungen zum System.014 dargelegt sind.
Eine Diskussion von Implementierungs- Varianten, von technischen Problemen usw. ist in den "Technischen Anforderungen" TA-01/87 zu finden.
Die grundlegenden Begriffe sind in der Schrift "Glossarium und Literaturverzeichnis" GL-01/87 erlaeutert. Dort sind auch bedeutsame Literaturstellen angefuehrt, und zwar sowohl hinsichtlich der theoretischen Grundlagen als auch hinsichtlich jener Arbeiten, die auf die Vorstellungen zum System.014 entscheidenden Einfluss ausgeuebt haben.
Fuer die exakte Dokumentation des Systems ist die "Technische Spezifikation" zusammen mit der Beschreibung der Theorie ("Handbuch der Wirkprinzipien" einschliesslich Formalbeschreibung), dem "Programmierhandbuch" sowie "Implementierungshandbuch" und "Anwendungshandbuch" (beide implementierungsspezifisch) vorgesehen.

2. Uebersicht

Das System.014 ist ein "integriertes Anwendungssystem", das es ermoeglichen soll, komplexe Auskunftssysteme, Systeme zur vertragsgerechten Steuerung der Produktion usw. in breitem Umfang einzufuehren. Die Anwendung soll keine spezifischen Computer- oder Programmierkenntnisse voraussetzen, so dass die Rechentechnik als alltaegliches problemlos nutzbares Arbeitsmittel in den jeweiligen Prozessen wirksam werden kann.
Integrierte Anwendungssysteme fuer den kommerziellen Einsatz sind seit einiger Zeit auf dem internationalen Markt verfuegbar (z. B. "Lotus 1-2-3", "open access", "KnowledgeMan"). Sie haben in der Regel folgende wesentliche Bestandteile:
- ein Datenbasis- System
- ein "spreadsheet"- Programm (Tabellenkalkulation)
- ein Textverarbeitungssystem
- ein System zur Erzeugung graphischer Darstellungen.

Die verschiedenen Systeme unterscheiden sich hauptsaechlich dadurch, in welchem Masse die einzelnen Funktionen ausgebaut sind. So kann ein System ueber besonders hochentwickelte Moeglichkeiten zur Textverarbeitung verfuegen, aber der Gebrauchswert des Tabellenkalkulationsprogramms vergleichsweise gering sein; bei einem anderen System koennen die Datenbank- Funktionen in aussergewoehnlichem Masse ausgebaut sein usw. Diese Ausgestaltungen haengen massgeblich davon ab, fuer welche Zielgruppe von Anwendern das jeweilige System vorgesehen ist.
Das System.014 ist der selben Kategorie von Softwareprodukten zuzuordnen, und es unterscheidet sich durch manche Besonderheiten von den bekannten Systemen:

1.

Es ist anwendungsseitig fuer eine bestimmte Art von Einsatzfaellen vorgesehen, naemlich fuer die Rationalisierung von Leitungs-, Verwaltungs- und Produktionsprozessen unter den gegebenen volkswirtschaftlichen Bedingungen, d. h. abgestimmt auf die Anforderungen des Reproduktionsprozesses. Diese Orientierung betrifft nicht die theoretischen Grundlagen (- sie sind davon voellig unabhaengig und muessen es auch sein -), sondern sie kommt in den Aufwendungen und Rangordnungen zum Ausdruck, die fuer die einzelnen Komponenten hinsichtlich Funktionseigenschaften und zeitlicher Reihenfolge der Implementierung vorgesehen sind. So ist unmittelbar einsichtig, dass umfassend ausgebaute Datenbankfunktionen wichtiger sind als Vorkehrungen zur graphischen Darstellung von Geschaeftsergebnissen.

2.

Es ist von vornherein als Mehrplatzsystem vorgesehen, d. h. an einen Rechner koennen mehrere einfache Bildschirm- Terminals angeschlossen werden, und an jedem Terminal kann ein Benutzer unabhaengig seine spezifischen Probleme bearbeiten.
Diese Betriebsweise hat fuer kommerzielle Anwendungen unbestreitbare Vorteile gegenueber der Vernetzung von Personalcomputern (ueber LAN). Zum einen ist wesentlich, dass sich nur so im praktischen Rechenbetrieb die Integritaet der Daten gewaehrleisten laesst. Schliesslich ist es von entscheidender Bedeutung, dass sich alle Benutzer auf einheitliche Daten verlassen koennen. So darf es keinesfalls vorkommen, dass beispielsweise Einkauf, Produktion, Lagerhaltung und Verkauf mit unterschiedlichen Daten der Materialbestaende arbeiten oder dass sie zu bestimmten Daten nicht zugreifen koennen, weil der betreffende Rechner des Netzwerkes gerade abgeschaltet ist.
Zum anderen werden die (vergleichsweise knapp zugeteilten) Rechner auf diese Weise besser ausgenutzt.

3.

Es ist theoretisch fundiert. Dem System.014 liegt eine relationale Wissensbasis zugrunde, die so gestaltet ist, dass kuenftige Erweiterungen bis hin zu kommerziellen Expertensystemen moeglich sind.
Die wesentlichen Algorithmen fuer Zugriffe zur Wissensbasis beruhen auf allgemeingueltigen mathematischen Sachverhalten, und die Vorstellungen zur Implementierung sind durch Forschungsergebnisse seitens der nicht- numerischen Informationsverarbeitung beeinflusst, die sowohl Effizienz und als auch funktionelle Korrektheit erwarten lassen.

Es ist "offen" fuer Weiterentwicklungen und fuer die Implementierung auf anderen (kuenftigen) Rechnern.
Diese "Offenheit" ist bei existierenden Systemen (verstaendlicherweise) nicht gegeben.
Das System.014 wird aus zwei Gruenden "offen" sein:

1. durch die Tatsache der Eigenentwicklung an sich: nur das, was gewissermassen "von Grund auf" selbst entwickelt wurde, kann wirklich als "verstanden" gelten
2. durch die speziell auf "Offenheit" ausgelegte Entwicklungsmethodik, die eine exakte und umfassende Dokumentation aller Entwicklungsleistungen vorsieht.

Die wesentlichen Komponenten des System.014 sind:

- eine relationale Wissensbasis
- ein Zugriffssystem fuer diese Wissensbasis
- eine Realzeit- Executive
- eine Laufzeitumgebung
- ein kommerzieller Interpreter
- eine Benutzeroberflaeche (Anwendungsumgebung)
- eine Entwicklungsumgebung
- ein Wartungssystem.

Grundsaetzlich geht das System.014 von der relationalen Wissensbasis aus, die ein einheitlich organisiertes, allen Nutzern zugaengliches Mittel zur Daten- und Programm- Speicherung darstellt (alle Konzepte sind gleichsam "um die Wissensbasis herum" aufgebaut).
Die Funktionen der Textverarbeitung sind in den Komponenten des Systems.014 enthalten. Besonderer Wert wird darauf gelegt, Texte aus an sich vorhandenen Bausteinen effektiv zu erstellen und Information aus der Wissensbasis in die Texte einzubringen.
Elementare Formen der Tabellenkalkulation, die oft benoetigt werden, sind im Rahmen der Benutzeroberflaeche vordefiniert. Diese Funktionen sind ihrerseits flexibel ausgestaltet, und die Nutzung dieser Flexibilitaet ist im Rahmen normaler Bedienhandlungen (ohne Programmierung) moeglich.
Fuer Formen der Tabellenkalkulation, die darueber hinausgehen, bietet der kommerzielle Interpreter COIN.014 entsprechende Sprachkonstrukte an.
Es sind verschiedene Implementierungen des System.014 vorgesehen, wobei es auch Abstufungen im Leistungsvermoegen gibt. Die ersten Implementierungen beruhen auf 16- bit- Personalcomputern. Einzelheiten dazu sind aus den "Technischen Anforderungen" (TA-01/87) ersichtlich.
Die folgenden Darstellungen gehen von einer vollen Funktionsfaehigkeit der ersten Entwicklungsstufe aus, d. h. es wird die volle Funktionsfaehigkeit angenommen mit Ausnahme jener Funktionen, die fuer Expertensysteme charakteristisch sind (das betrifft Inferenzregeln, Ereignisse usw.).

3. Einblick in Theorie und Wirkungsweise

Um das System.014 umfassend beschreiben zu koennen, muessen grundlegende Begriffe eingefuehrt werden. Diese Begriffe muessen exakt genug sein, um Eindeutigkeit und Genauigkeit der Beschreibung zu gewaehrleisten, und sie muessen allgemein genug sein, um der angestrebten Universalitaet des System.014 gerecht werden zu koennen. Solche grundlegenden Begriffe werden als "Fundamentalkategorien" bezeichnet ("Kategorien" sind oberste Aussageformen ueber das Seiende, also maximal umfassende Begriffe). Zu den Fundamentalkategorien gehoeren:

- Begriffe (Terme)
- Relationen (Praedikate)
- semantische Kategorien
- Zugriffsregeln (invocation rules; Invocationsregeln)
- Schlussregeln (inference rules; Inferenzregeln)
- Quantifizierer (numerische Bestimmungen)
- Dimensionierungen (Masseinheiten der Quantifizierer)
- Qualifizierer (alphanumerische Bestimmungen)
- Texte
- Benutzer
- logische Terminals
- Ereignisse
- Aktionen.

Allgemein heissen die Namen, die Elementen der Wissensbasis zukommen, "Bezeichner" oder "Designatoren".
Es wird grundsaetzlich angenommen, dass die Designatoren im Sinne der Verstaendlichkeit gewaehlt werden und dass jeder Anwendung die (deutsche) Umgangssprache zugrunde gelegt wird.
Zur Bildung von Designatoren ist das Alphabet (Klein- und Grossbuchstaben einschliesslich Umlauten), das Zeichen "Unterstreichung" (_) sowie das Leerzeichen vorgesehen. Designatoren koennen also aus mehreren Worten bestehen, sofern zwischen den einzelnen Worten nicht mehr als ein Leerzeichen vorkommt.
Fuer jeden Designator ist in der Wissensbasis die Fundamentalkategorie vermerkt. Die Beziehungen zwischen Designatoren heissen Praedikate bzw. Relationen.
Eine Relation der Mengen M,N,O,P... ist eine Abbildung der Menge M*N*O*P*... in die Menge der Wahrheitswerte (0,1 bzw. "falsch"/"wahr"). (Das Zeichen "*" symbolisiert das Kreuzprodukt.) Relationen werden prinzipiell durch Aufzaehlung der n- Tupel dargestellt, denen der Wahrheitswert 1 zukommt (d. h. durch Angabe jener n- Tupel, die die Relation erfuellen).
Um die Bedeutung der Designatoren der rechentechnischen Auswertung zugaenglich zu machen, sind die semantischen Kategorien vorgesehen. Es gibt Designatoren, die die Fundamentalkategorien bezeichnen. Damit sind sie eindeutig erklaert. Alle anderen Designatoren sind semantischen Kategorien zugeordnet. Auch diese semantischen Kategorien werden mit Designatoren benannt. Jede semantische Kategorie drueckt einen Universalbegriff aus, der gewissermassen die maximal umfassende Verallgemeinerung darstellt. Solchen Universalbegriffen kommt selbst keine Gegenstaendlichkeit zu. Beispiel:
Es wird eine semantische Kategorie "Personal" eingefuehrt.
Dieser Universalbegriff ist durch Begriffe mit gegenstaendlicher Bedeutung genauer zu bestimmen, beispielsweise durch

- "Name"
- "Vorname"
- "Geburtstag"

- "Anschrift"
- "Status"
- "Telephonnummer".

Das sind alles Designatoren der Wissensbasis, die zur semantischen Kategorie "Personal" gehoeren. Diese Designatoren sind selbst weiter zu bestimmen - und das ist eine pragmatische Angelegenheit, eine Frage der Zweckmaessigkeit fuer die jeweilige Anwendung. So kann beispielsweise spezifiziert werden:
- "Name" als Qualifizierer mit 16 Zeichenpositionen
- "Vorname" als Qualifizierer mit 12 Zeichenpositionen
- "Geburtstag" als Quantifizierer (numerischer Wert) mit der Dimensionierung "Datum"
- "Anschrift" als Qualifizierer mit 80 Zeichenpositionen usw.

Dies entspricht ueblichen Datenbanksystemen.
Fuer die folgenden Erlaeuterungen seien einige Funktionen des System.014 vorausgesetzt, die in Abschnitt 4. naeher beschrieben werden:
- Das blosse Hinschreiben eines Designators bewirkt das Abrufen gespeicherten Information.
- Es koennen zwecks genauerer Bestimmung Ketten von Designatoren angegeben werden. Die einzelnen Designatoren sind durch Punkte voneinander zu trennen.
- Zu jedem Designator gibt es eine textliche Beschreibung. Diese ist durch 2 nachgestellte Fragezeichen abrufbar.
- Die beschreibende (metasprachliche strukturell- deskriptive) Information ist durch 3 nachgestellte Fragezeichen abrufbar.

So fuehrt das blosse Eingeben von "Personal" (Anfuehrungszeichen koennen weggelassen werden !) dazu, dass die gesamte Relation ausgegeben wird (bzw. ab einer gewissen Groesse zunaechst zur Ausgabe angeboten wird).

"Personal.Name" veranlasst nur die Ausgabe der Namen.
"Personal.Name=Meier" veranlasst die Ausgabe aller n- Tupel der Relation, in denen der Name "Meier" vermerkt ist.

Nun ist wesentlich, dass das "Wissen" des System.014 auf den gespeicherten Designatoren und den Relationen zwischen ihnen beruht. "Meier" ist aber kein Designator ! Die Anfrage "Meier" wuerde mit einer Fehlmeldung beantwortet werden. Man muesste wenigstens "Name=Meier" angeben - und das wuerde nur dann nicht zu Mehrdeutigkeiten fuehren, wenn es den Begriff "Name" nicht auch in anderen Relationen gaebe, etwa im Rahmen einer semantischen Kategorie "Firmen" als Betriebsnamen.
Soll eine solche unspezifizierte Anfrage moeglich sein, so muessen die Namen ebenfalls als Designatoren in die Wissensbasis eingetragen werden, d. h.:
"Name" ist ein Praedikat (eine 1-stellige Relation) der semantischen Kategorie "Personal". "Meier", "Schulze", "Huber" usw. sind Terme (Begriffe) der semantischen Kategorie "Personal", denen das Praedikat "Name" zukommt. Genauer gesagt sind "Meier", "Schulze" usw. Individuen der semantischen Kategorie "Personal" (sie haben in dieser semantischen Kategorie die Ordnungszahl 1), das Praedikat "Name" hat die Ordnungszahl 2. Sinngemaess liesse sich etwa der Designator "Status" als Praedikat fassen; die zugehoerigen Individuen- Designatoren sind z. B. "Grundarbeiter", "Hilfsarbeiter", "Angestellter", "Rentner" usw.
So reicht es aus, das Wort "Rentner" einzugeben, und das System wird die Daten aller Rentner der Personaldatei liefern. Der

Benutzer braucht sich nicht darum zu kuemmern, in welcher Datei der Begriff zu suchen ist.
Eine Eingabe "Rentner??" liefert die textliche Beschreibung, die zu diesem Designator gehoert (sie ist anwendungsspezifisch auszufuellen).
Eine Eingabe "Rentner???" liefert die systeminterne metasprachliche Beschreibung des Designators, also
- "Rentner" ist ein Term der semantischen Kategorie "Personal"
- er hat die Ordnungszahl 1
- er ist in einer einzigen Relation enthalten, und zwar in der Relation "Status".

Solche Anfragen an die Wissensbasis sind sowohl unmittelbar durch den Benutzer als auch im Rahmen von Programmen moeglich.
Prinzipiell wird bei jeder Anfrage das gesamte "Wissen" zugaenglich, soweit es durch die konkrete Struktur der Anfrage spezifiziert ist. Jeder so abgerufene Ausschnitt des "Wissens" ist fuer die Weiterverarbeitung verwendbar, wobei die zulaessige Art und Weise der Weiterverarbeitung durch die zugeordnete semantische Kategorie und Fundamentalkategorie bestimmt wird. So sind Rechenoperationen nur mit numerischen Werten (Quantifizierern) zugelassen, die der selben semantischen Kategorie und der selben Ordnung angehoeren. So ist es unmoeglich, "Tonnen Aepfel" und "Tonnen Birnen" zu addieren, man kann aber "Tonnen Obst" bilden, sofern es ein Praedikat "Obst" gibt, dass "Aepfel" und "Birnen" umfasst (handelt es sich um Praedikate, so wird auf identische Dimensionierungen hin geprueft - d. h. die entsprechenden Quantifizierer werden sinngemaess verrechnet).
Die bisher illustrierten Zugriffe zur Wissensbasis laufen faktisch auf Auswahloperationen hinaus. Diese sind als elementare Invocationsregeln implementiert. Des weiteren koennen Programme an sich beliebiger Art als anwendungsspezifische Invocationsregeln in das System eingebracht werden. Auch diese Programme werden durch Designatoren aktiviert. So koennte beispielsweise eine Anfrage JAHRESERGEBNIS.X_DORF ein Programm starten, dass eine Auswertung von Produktionsergebnissen eines Jahres vornimmt. Im konkreten Fall wuerde diese Programm die Daten des Werkes in X_DORF verarbeiten.

4. Die Komponenten des System.014

4.1. Wissensbasis RKB.014

4.1.1. Ueberblick

Die Wissensbasis nimmt alle Datenstrukturen des System.014 auf. Jeder Eintrag stellt im allgemeinen Sinne ein "Objekt" dar, das zu einer bestimmten Fundamentalkategorie gehoert. Alle Objekte werden durch Designatoren benannt. Die Wissensbasis selbst wird als geordnete Menge aller Objekte aufgefasst, und intern werden die Objekte durch ihre Ordinalzahlen identifiziert.
Anmerkung:
Grundsaetzlich wird jede Ordinalzahl binaer mit 24 bit codiert. In manchen Implementierungen ist der Bereich der Ordinalzahlen (bzw. gleichbedeutend: die maximale Anzahl der Designatoren) eingeschraenkt; trotzdem wird konzeptionell stets die 24- bit- Struktur vorausgesetzt.
Es wird erwartet, dass die Designatoren sinnvoll aus dem Bereich der Umgangssprache gewaehlt werden. So sollte jeder Designator mindestens 3 Buchstaben umfassen. Buchstaben und Ziffern duerfen

nicht zusammengeschrieben werden. Designatoren duerfen aus Worten und Zifferngruppen bestehen, die untereinander durch ein Leerzeichen bzw. durch ein Unterstreichungssymbol getrennt sind (weitere Einzelheiten s. Abschnitt 4.6.). Synonyme und Mehrfachbezeichnungen sind vom Prinzip her zulaessig (mehrere Designatoren bezeichen ein Objekt bzw. ein Designator bezeichnet mehrere Objekte - eine solche Nutzung kann allerdings zur Folge haben, dass bei Eingaben weitere Spezifikationen angefordert werden).

4.1.2. Uebersicht ueber die Fundamentalkategorien

4.1.2.1. Semantische Kategorien

Der Zweck der semantischen Kategorien besteht darin, die Bedeutung der Objekte fuer die rechentechnische Behandlung zugaenglich zu machen. Semantische Kategorien druecken Universalbegriffe aus, denen keine gegenstaendliche Bedeutung zukommt und die eine jeweils maximal umfassende Verallgemeinerung darstellen. Jede semantische Kategorie ist durch die Gesamtheit der Objekte gekennzeichnet, die ihr zugeordnet sind. Diese Zuordnung ist eine Angelegenheit der Anwendung; sie setzt eine genaue Analyse der anwendungsseitigen Probleme und Prozesse voraus - und sie muss mit einer gewissen Verantwortung gehandhabt werden: der Rechner selbst kann die semantischen Kategorien lediglich nach den grundlegenden Zugriffsregeln mechanisch interpretieren.
Zur Charakterisierung eines Designators gehoert neben der semantischen Kategorie als solcher noch eine Ordnungszahl innerhalb dieser Kategorie, die wie folgt eingefuehrt wird:
- ein Objekt, das nicht aus weiteren Objekten zusammengesetzt ist (d. h. das nicht eine Menge aus anderen Objekten darstellt), hat die Ordnungszahl 1
- ein Objekt, das nur Objekte der Ordnungszahl 1 enthaelt, hat die Ordnungszahl 2
- allgemein hat ein Objekt, das Objekte der Ordnungszahlen 1...n enthaelt, die Ordnungszahl n + 1.

Objekte mit der Ordnungszahl 1 repraesentieren mithin die nicht weiter zerlegbaren Individuen.
Aus praktischen Gruenden sind die Ordnungszahlen in relativer Form angebbar, d. h. zu jeder semantischen Kategorie gibt es eine Basis- Ordnungszahl, die der Ordnung 1 (also den Individuen) zukommt. Damit ist es moeglich, Objekte, die beim Aufbau der Wissensbasis als Individuen angesehen wurden, spaeter weiter zu zerlegen; es wird dann lediglich die Basis- Ordnungszahl entsprechend vermindert.
Mithin ist jede semantische Kategorie gekennzeichnet durch:
- ihren Eigennamen (Designator)
- die Menge der ihr zugehoerenden Objekte
- die Basis- Ordnungszahl
- die aktuelle hoechste Ordnungszahl.

Den Objekten folgender Fundamentalkategorien sind semantische Kategorien zugeordnet:
- Terme
- Relationen
- Qualifizierer
- Quantifizierer
- Dimensionierungen
- Texte
- Invocationsregeln.

Folgende Fundamentalkategorien haben keine semantischen Kategorien:
- die semantischen Kategorien selbst
- Benutzer
- logische Terminals.

Folgende Fundamentalkategorien werden hier nicht im einzelnen betrachtet:
- Inferenzregeln
- Ereignisse
- Aktionen.

4.1.2.2. Begriffe (Terme)

Diese Objekte sind die nicht weiter zerlegbaren Einheiten der Wissensbasis (Individuen). Sie haben in der jeweiligen semantischen Kategorie die Basis- Ordnungszahl.

4.1.2.3. Relationen (Praedikate)

Diese Objekte repraesentieren die Beziehungen zwischen den Komponenten der Wissensbasis. Das grundlegende Modell einer Relation als Abbildung aus einer Menge N*M·P... in die Menge der Wahrheitswerte ist eine Liste jener n-Tupel des Kreuzproduktes, die die Relation erfuellen (d. h. in den Wahrheitswert "1" abgebildet werden). Die Objekte, die die Mengen N, M, P... als solche repraesentieren, heissen konstituierende Objekte der Relation. Die Objekte, die die einzelnen n-Tupel bilden, sind der Inhalt der Relation.
Bei grundsaetzlich gleicher Interpretation werden unterschieden:
- Aufzaehlungsrelationen
- Zuordnungsrelationen
- Systemrelationen.

Alle einstelligen Relationen (Praedikate) sind Aufzaehlungsrelationen. Es handelt sich dabei um Listen aller Objekte, die die Relation erfuellen. Eine solche Relation beschreibt mithin eine Menge durch Aufzaehlung ihrer Elemente. Diese Elemente muessen zur selben semantischen Kategorie gehoeren und dieselbe Ordnungszahl haben. Die Relation selbst hat dann die um 1 erhoehte Ordnungszahl.

Zuordnungsrelationen entsprechen direkt dem genannten Modell einer Liste der n-Tupel. Sie werden zum einen durch die konstituierenden Objekte und zum anderen durch den Inhalt bestimmt. Die Objekte des Inhaltes muessen zur selben semantischen Kategorie gehoeren wie das jeweils zugehoerige konstituierende Objekt, und deren Ordnungszahl muss niedriger sein.

Systemrelationen sind Aufzaehlungs- bzw. Zuordnungsrelationen, die mit beliebigen Objekten der Wissensbasis gebildet sind. Sie werden vom System.014 intern verwendet, um das universelle Konzept der Relationen fuer die eigene Verwaltung zu nutzen.

Grundsaetzlich werden alle Relationen in dicht gepackter binaer codierter Form gespeichert. Die prinzipielle Beschraenkung fuer alle Implementierungen besteht darin, dass die binaere Repraesentation eines n-Tupels hoechstens 4095 bit belegen darf.

Relationen koennen mit folgenden Fundamentalkategorien gebildet werden:
- Terme
- weitere Relationen
- Qualifizierer
- Quantifizierer
- Dimensionierungen
- Text.

Systemrelationen koennen mit Objekten aller Fundamentalkategorien aufgebaut werden.

Um die konstituierenden Objekte bei der Darstellung von Relationen (z. B. auf einem Bildschirm) zu benennen, sind zwei Moeglichkeiten waehlbar:
1. Angabe des Designators des jeweiligen konstituierenden Objekts
2. Angabe eines speziellen Titels (in Textform; der Titel ist Bestandteil des Objekts, das die Relation enthaelt).

4.1.2.4. Qualifizierer

Diese bezeichnen die Struktur von Erklaerungen, Kennzeichnungen, Bestimmungen usw., die ihrerseits nicht als Designatoren zur Wissensbasis gehoeren. Solche Objekte werden bestimmt durch die maximal zulaessige Zeichenzahl sowie durch weitere Angaben, wie z. B.:
- Gross/Klein- Schreibung ignorieren oder nicht
- kuerzere Zeichenfolgen auffuellen oder nicht
- Sonderzeichen zulassen oder nicht.

Qualifizierer haben in der jeweiligen semantischen Kategorie eine Ordnungszahl, die um Eins groesser ist als die Basis-Ordnungszahl; sie bezeichnen mithin Mengen von Individuen. Diese Individuen selbst sind jedoch keine Komponenten der Wissensbasis.

Qualifizierer koennen 1...63 Zeichen umfassen.

4.1.2.5. Quantifizierer

Diese Objekte kennzeichnen numerische Angaben. Bestimmende Groessen sind:
- Stellenzahl
- Typ der Darstellung (binaer, dezimal usw.)
- Vorzeichen
- Position des Kommas

- Wertebereich
- Reaktion bei nicht adaequater Eingabe
- Dimensionierung.

Es ist eine maximale Stellenzahl von 30 zugelassen (zuzueglich Komma und Vorzeichen). Beispiele fuer Darstellungen von Quantifizierern sind:
- dezimal (festes Komma an beliebiger Stelle)
- binaer 15 bit
- binaer 31 bit
- binaer Gleitkomma.

Manche Dimensionierungen koennen eine implizite Formatierung des Quantifizierers bewirken (z. B. die Dimensionierung "Datum").

Quantifizierer haben eine Ordnungszahl, die mindestens um Eins hoeher ist als die Basis- Ordnungszahl der jeweiligen semantischen Kategorie.
Quantifizierer hoeherer Ordnungszahl sind praktisch Listen von Quantifizierern niederer Ordnungszahl. Solche Listen ermoeglichen verschiedene Formen der Angabe, z. B.:
- verschiedene Bezeichner, verschiedene Struktur
- verschiedene Bezeichner, gleiche Struktur
- ein Bezeichner (der der Liste), Quantifizierer mit verschiedener Struktur
- ein Bezeichner, gleiche Struktur.

4.1.2.6. Dimensionierungen

Diese kennzeichnen die jeweilige Masseinheit eines Quantifizierers. Dimensionierungen der Ordnungszahl 1 sind durch ihren Designator selbst gekennzeichnet. Dimensionierungen hoeherer Ordnungszahl sind Listen von Dimensionierungen, die eine bestimmte Art der Bemessung kennzeichnen. So kann z. B. Geld in verschiedenen Waehrungseinheiten gemessen werden, fuer Gewichts-Angaben sind kg oder Tonnen moeglich usw. Derartige Listen koennen auch Umrechnungsregeln enthalten.

4.1.2.7. Texte

Texte sind bestimmt durch:
- Format- und Darstellungsattribute (Zeichenzahl pro Zeile, Zeilenzahl pro Seite, Schriftart, Seitennumerierung, besonderes Layout, etwa mit Platz fuer Zeichnungen usw.)
- die Gliederung
- funktionelle Abhaengigkeiten
- den eigentlichen Inhalt.

Texte koennen konstituierende Objekte in Relationen sein. Im Unterschied zu Qualifizierern ist es aber nicht moeglich, in Relationen nach dem Inhalt solcher Text- Objekte zu suchen.

Text- Inhalte koennen funktionelle Abhaengigkeiten enthalten, und zwar sowohl als direkte Bezugnahmen auf die Wissensbasis als auch in Form funktioneller Abkuerzungen, die in der zugehoerigen Funktionsliste zu erklaeren sind. Damit koennen von einem grundlegenden Text aus eine Vielzahl von Texten mit konkretem Inhalt abgeleitet werden. Diese Texte sind wieder separat speicherbar, wobei

die funktionellen Abhaengigkeiten stets mitgefuehrt werden.

Texte ohne funktionelle Abhaengigkeiten haben die Ordnungszahl 1. Ansonsten ist die Ordnungszahl des Textes um 1 hoeher als die hoechste Ordnungszahl der funktionellen Abhaengigkeiten. Werden solche Abhaengigkeiten durch konkrete Inhalte ersetzt, so vermindert sich die Ordnungszahl des Textes entsprechend.

4.1.2.8. Invocationsregeln

Invocationsregeln sind die Algorithmen fuer Zugriffe zur Wissensbasis. Jede Invocationsregel ist durch die spezifische Operation sowie die zugehoerigen Parameter bestimmt.
Elementare Invocationsregeln sind im Rahmen der Komponente ASK.014 vorgesehen (s. Abschnitt 4.2.).
Komplexe Invocationsregeln koennen mit dem Interpreter COIN.014 aufgebaut werden (s. Abschnitt 4.5.) bzw. sie koennen als an sich beliebige Programme (nur bestimmte Anschlussbedingungen sind einzuhalten) auf einem Entwicklungssystem erstellt und in das Anwendungssystem eingebracht werden.

4.1.2.9. Benutzer

Jeder Benutzer ist im System gekennzeichnet durch:
- seinen Namen
- seine Kennung
- seine Berechtigungen
- eine persoenliche Signatur, die er selbst jederzeit aendern kann.

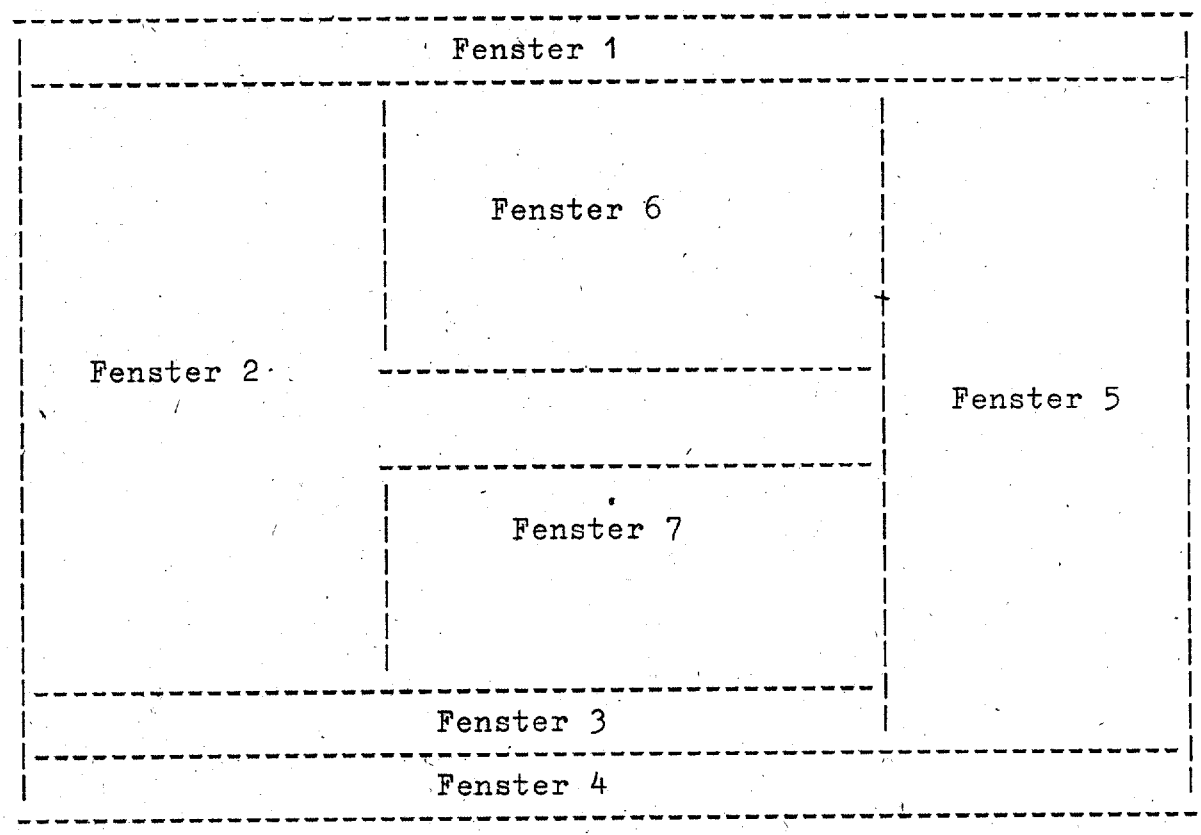
Diese Bestimmungen werden von der Laufzeitumgebung verwaltet (s. Abschnitt 4.3.).
Im allgemeinen ist fuer jeden Benutzer in einer Systemrelation angegeben, zu welchen Objekten der Wissensbasis er in welcher Weise Zugriff hat.

4.1.2.10. Logische Terminals

Im System.014 ist das logische Terminal durch bestimmte funktionelle Eigenschaften hinsichtlich Funktionsauswahl, Eingabe und Informationsdarstellung charakterisiert.
Wesentlich ist, dass eine gewisse Anzahl von Fenstern vorgesehen ist, die auf dem physischen Terminal einzeln oder in verschiedenen Kombinationen dargestellt werden koennen. Das Wechseln und Ueberblenden der Anzeigen wird durch CASE.014 gesteuert; es kann teilweise manuell durch Funktionsauswahl beeinflusst werden.
In COIN.014 sind Anweisungen vorgesehen, um Fenster zu erzeugen, darzustellen, Eingaben anzufordern usw.
Normalerweise ist ein ueblicher Bildschirm mit 24 Zeilen zu 80 Zeichen wie folgt aufgeteilt:
Fenster 1: 1. Zeile fuer Funktionszustand und aktuelle Ueberschrift

Fenster 2: 21 Zeilen zu 64 Zeichen fuer allgemeine Darstellungen
Fenster 3: 1 Zeile (die 23.) zu 64 Zeichen fuer Eingaben (dieses Fenster kann, falls noetig, nach oben bis zu 15 Zeilen expandieren und dabei Fenster 2 teilweise ueberblenden
Fenster 4: 1 Zeile (die 24.) zu 80 Zeichen fuer Fehleranzeigen, Bedienhinweise usw. (ist normalerweise leer)
Fenster 5: 22 Zeilen zu 16 Zeichen am rechten Rand fuer die Erklaerung der Funktionstasten
Fenster 6: 16 Zeilen zu 32 Zeichen fuer Hilfs- Information. Damit wird bedarfsweise die rechte obere Ecke von Fenster 2 ueberblendet.
Fenster 7: 10 Zeilen zu 32 Zeichen zur Auswahlsteuerung bei umfangreichen Darstellungen (Auswaehlen von Teil- Anzeigen, Verschieben usw.). Damit wird bedarfsweise die rechte untere Ecke von Fenster 2 ueberblendet.

Erscheinungsbild:



4.1.2.11. Inferenzregeln, Ereignisse, Aktionen

Diese Fundamentalkategorien werden zunaechst nicht bzw. nur in elementarer Form implementiert. Sie sind erforderlich, um das System.014 zum Expertensystem zu erweitern. Inferenzregeln ermoeglichen beliebige logische Verknuepfungen von Relationen und deren Auswertung. Ereignisse kennzeichnen das Auftreten bestimmter Sachverhalte, und zwar sowohl in der realen Laufzeitumgebung des Systems (z. B. Meldungen peripherer Einrichtungen - dies ist die

Art von Ereignissen, die von Anfang an verarbeitet werden-) als auch in der Wissensbasis. Aktionen stellen aktive automatisch ausgeloeste Einwirkungen auf die reale Laufzeitumgebung bzw. auf die Wissenbasis dar.

4.2. Zugriffssystem ASK.014

Dieser Komplex umfasst die elementaren Invocationsregeln fuer die Wissensbasis RKB.014 sowie die grundlegenden Funktionen fuer den Zugriff zu den Designatoren und deren Verwaltung.
Alle internen Repraesentationen der Datenstrukturen des System.014 beruhen auf dicht gepackten binaeren Codierungen. Im besonderen werden die Objekte intern nicht durch ihre Designatoren, sondern durch binaer codierte Ordinalzahlen bezeichnet.
Eine Komponente von ASK.014 gewaehrleistet die Wandlung von Designatoren in Ordinalzahlen und umgekehrt.
Eine weitere Komponente dient zur Verwaltung der Wissensbasis. Wesentliche Funktionen sind:
- allgemeinen Status anzeigen/modifizieren
- Objekte zufuegen
- Objektbeschreibungen (strukturell- deskriptive Angaben ueber die Objekte) anzeigen/ergaenzen/modifizieren
- Objekte entfernen (Speicherplatz bleibt belegt)
- Objekte loeschen (Speicherplatz wird freigegeben).

Schliesslich enthaelt ASK.014 Funktionen fuer Zugriffe zu den Komponenten der Objekte.
Die Selektion von Komponenten ist auf allen Ebenen moeglich, vom gesamten Objekt bis hin zu jenen einzelnen Komponenten, die nicht weiter zerlegbar sind.
Solche Zugriffen liegt folgendes Prinzip zugrunde:
Die binaeren Repraesentationen der Objekte werden als Binaervektoren im BOOLEschen Raum bzw. gleichbedeutend: als Loesungsmengen BOOLEscher Gleichungen aufgefasst. Damit ist die Isomorphie zwischen Mengen- und Binaeroperationen rechentechnisch ausnutzbar. Im einzelnen entsprechen:
- Vereinigung und Disjunktion
- Durchschnitt und Konjunktion
- symmetrische Differenz und Antivalenz
- Komplement der symmetrischen Differenz und Aequivalenz
- Komplement und Negation.

Durch Kombination solcher Operationen mit Umbaufunktionen (wie Selektieren, Mischen, Zusammensetzen usw.), die sowohl fuer einzelne Binaervektoren als auch fuer Listen gelten, koennen alle Anforderungen an relationale Datenbasis- Systeme erfuellt werden.
Als besonders zweckmaessig erweist es sich, vom Prinzip der ternaeren Speicherung auszugehen.
Dabei ist fuer jede binaere Variable eine dreiwertige Codierung vorgesehen (0, 1, -; letztere kennzeichnet einen "don't care"-Wert). Binaervektorlisten koennen in Ternaervektorlisten (TVL) ueberfuehrt werden, und solche Listen liegen - zumindest konzeptionell - allen Mengenoperationen im System.014 zugrunde. Allgemein ist von der ternaeren Codierung eine deutliche Reduktion des Speicherbedarfs zu erwarten: es werden zwar 2 Bit pro Ternaervariable benoetigt, aber die Anzahl der Vektoren wird deutlich verringert (ein Vektor mit 2 Strich- Variablen ersetzt 4 Binaervektoren, benoetigt aber nur den Speicherplatz von 2 Binaervektoren).

Fuer typische Listen, die n-Tupel von Relationen repraesentieren, gilt dies jedoch nicht immer, besonders dann nicht, wenn die Listen von Natur aus orthogonal sind, d. h. wenn beim Vergleich zweier beliebiger Vektoren einer Liste in jeweils wenigstens einer Variablenposition die Wertekombination 1,0 bzw. 0,1 auftritt.
In solchen Faellen wird im System.014 die Speicherung mit Binaervariablen beibehalten, die Algorithmen sind aber so ausgelegt, dass alle Ternaer- Operationen sinngemaess angewendet werden koennen. Beispielsweise laeuft eine elementare Abfrage einer Relation wie folgt ab:

- Es werden 2 weitere Relationen (1 Abfragerelation, 1 Ergebnisrelation) mit den selben konstituierenden Objekten wie die abzufragende Relation vorbereitet. Diese neuen Relationen haben zunaechst keinen Inhalt.
- Aus den angegebenen Objekten der Abfrage wird die Abfragerelation aufgebaut (das wird oft nur ein einziges n-Tupel sein). Diese Relation wird ternaer codiert; freie Positionen (fuer die in der Abfrage keine Objekte angegeben wurden), werden mit Strich- Belegungen aufgefuellt.
- Mit beiden Listen wird die Durchschnittsoperation ausgefuehrt; dadurch entsteht der (ternaer codierte) Inhalt der Ergebnisrelation.
- Aus dem Resultat werden die gewuenschten Komponenten selektiert (diese sind dann binaer codier verfuegbar).
- Die beiden Hilfs- Relationen werden geloescht.

Als Beispiel sei eine Relation mit Personaldaten angefuehrt:

Name	Vorname	Status	Einkommen
Maier	Anna	Hilfsarbeiter	800,00
Meier	Franz	Grundarbeiter	1100,00
Meier	Johann	Angestellter	750,00
Nolte	Margot	Angestellter	790,00

Es sollen die Daten aller Angestellten ausgegeben werden. Dementsprechend werden an allen Positionen der Abfragerelation ausser an der des konstituierenden Objekts "Status" Striche eingetragen:

----------Angestellter----------

Die Durchschnittsoperation laeuft auf ein Durchmustern hinaus, wobei alle nichtorthogonalen Zeilen gesucht werden (Kombinationen 0,1 bzw. 1,0 kommen nicht vor). Da die Ordinalzahlen aller Designatoren voneinander verschieden sind, ist die Nichtorthogonalitaet nur bei den Zeilen der abzufragenden Relation gegeben, die den Designator "Angestellter" enthalten. Diese Zeilen werden in die Resultatliste aufgenommen (Belegungen 0,1 dominieren bei der Erzeugung des Durchschnittsvektors ueber Striche). Zur Ausgabe des Resultats werden alle konstituierenden Objekte mit Ausnahme von "Status" selektiert, so dass folgende Anzeige erscheint:

Angestellter

Name	Vorname	Einkommen
Meier	Johann	750,00
Nolte	Margot	790,00

Zur weiteren Demonstration sei eine Anfrage nach allen Angestellten angegeben, deren Name mit "M" beginnt.

Die Anfrage selbst lautet:

Angestellter.Name = M..

Daraus wird die Abfragerelation erzeugt:

M----------Angestellter----------

Wesentlich ist, dass ASK.014 alle Anfragen dem Sinn nach auf dieses Modell zurueckfuehrt. Die Implementierung kann von Fall zu Fall verschieden sein, ebenso koennen die Algorithmen Modifikationen zur Leistungssteigerung enthalten (so koennen Positionen, in denen konstituierende Objekte voll mit Strichelementen belegt sind, beim Durchmustern von vornherein ausgeschlossen werden). In jedem Fall muss das Resultat gleich jenem sein, dass bei der (gedanklichen) Ueberfuehrung in "reine" TVL und Anwendung der wohldefinierten Mengenalgorithmen entstehen wuerde. Relationen, die Relationen enthalten, werden in diesem Zusammenhang als TVL- Systeme behandelt.
Fuer die Auswertung von Quantifizierern oder Qualifizierern werden erforderlichenfalls weitere Hilfsrelationen erzeugt, die dann binaere Resultate von arithmentischen Vergleichsoperationen o. dergl. enthalten.
Grundsaetzlich erhaelt ASK.014 die Parameter fuer die jeweilige Invocation von der Benutzeroberflaeche CASE.014 oder vom kommerziellen Interpreter COIN.014.
Die Parameter werden zunaechst in eine wohldefinierte Reihenfolge gebracht, die im einzelnen von den Fundamentalkategorien, semantischen Kategorien und Ordnungszahlen abhaengt.
Dann wird gepruefet, ob diese Abfrage einem der in ASK.014 vorgesehenen Muster ("templates") entspricht (manche Implementierungen enthalten nur eine Auswahl an "templates"). Falls ein gueltiges Muster gefunden wurde, wird die jeweilige Invocationsregel angewendet, die ihrerseits aus elementareren Operationen aufgebaut ist, wie Selektionen, Aufbau von Relationen, Mengenoperationen (z. B. Durchschnitt, Vereinigung) usw.

4.3. Realzeit- Executive RTX.014

Dieser Komplex ermoeglicht es, dass eine Vielzahl von Benutzern gleichzeitig mit dem System.014 arbeiten kann.
In manchen Implementierungen koennen die Funktionen von RTX.014 vollstaendig von einem Standard- Betriebssystem ausgefuehrt werden, etwa wenn es sich um ein 1- Platz- System handelt oder wenn Rechner und Betriebssystem an sich den leistungsmaessigen und funktionellen Anforderungen des System.014 gerecht werden (in diesem Sinne ist die Definition von RTX.014 gewissermassen als Checkliste aufzufassen, um die Eignung gegebener Betriebbsysteme zu pruefen).
Typische Funktionen sind:
- Ereignissteuerung
- Laufzeitverwaltung
- Objektverwaltung.

Im allgemeinen koennen so viele Benutzer gleichzeitig mit dem System arbeiten wie Terminals angeschlossen sind. Jedem Terminal ist dabei eine Task zugeordnet. Die Aufgaben der Ereignissteuerung besteht darin, auf Ereignisse zu reagieren, die entweder physisch (durch Interrupts in der Hardware) oder logisch (durch programmseitige Aktivitaeten) ausgeloest wurden. Ein Ereignis fuehrt im allgemeinen zu einem Programmstart, dem Verlassen eines Wartezustandes, einem Eintrag in eine Warteschlange usw.
Die Laufzeitverwaltung muss den einzelnen Tasks angemessene Laufzeit zur Verfuegung stellen. Dazu reichen einige einfache Prinzipien vollauf aus:
Eine Task im Ruhezustand erhaelt sofort Laufzeit, wenn ein entsprechender Programmstart auseloest wurde. Die Task, der die Laufzeit entzogen wurde, wird am Ende einer Laufzeitwarteschlange eingeordnet. Gelangt eine Task in den Ruhezustand (z. B. durch eine TERMINATE- Anweisung), so erhaelt die jeweils erste Task aus der Laufzeitwarteschlange Laufzeit. Damit wird ein zyklisches Weiterschalten zwischen den aktiven Tasks ("round robin") gewaehrleistet. Ist die Laufzeit des einzelnen Programms relativ lang, so ist ein zeitgesteuertes zyklisches Weiterschalten (z. B. im Rhythmus von 30 ms) vorgesehen ("time slicing"). Fuer Sonderzwecke kann veranlasst werden, dass eine Task nicht am Ende, sondern am Anfang der Laufzeitwarteschlange eingeordnet wird. Weiterhin ist es programmseitig steuerbar, ob ein Ereignis in einer bereits aktiven Task einen Programmstart erzwingt oder ob die Programmstart- Information in eine Auftragswarteschlange der Task eingetragen wird.
Die Objektverwaltung muss die reale Struktur der technischen Speichermittel (RAM, Festplatte, Floppy disc usw.) auf die abstrakte Objektstruktur abbilden, auf die sich alle anderen Komplexe des System.014 beziehen. Grundsaetzlich arbeiten diese Komplexe mit Objektidentifiern (bzw. Ordinalzahlen), und die Objektverwaltung muss die betreffenden Objekte zur Verarbeitung bereitstellen, den Datenaustausch zwischen RAM und Externspeichern organisieren usw.
Der RAM ist grundsaetzlich so aufgeteilt, dass neben gemeinsam nutzbaren Bereichen (COMMON- Bereichen) je Task fest formatierte Bereiche reserviert sind. Stack- und Arbeitsbereiche werden fuer jede Task separat gefuehrt. Daten und Programme befinden sich entweder in gemeinsam nutzbaren oder in Task- spezifischen Bereichen. Im allgemeinen werden Objekte der Benutzer- Umgebung in Task- spezifische Bereiche geladen und Objekte der eigentlichen Wissensbasis in COMMON- Bereiche. (Da die gemeinsame Nutzung der Wissensbasis ein charakteristisches Merkmal der .014- Anwendungen

ist, kann erwartet werden, dass viele der RAM- residenten Objekte von mehreren Tasks benoetigt werden. Daraus ergibt sich eine bessere Nutzung des RAM als bei Verzicht auf COMMON- Bereiche zugunsten grosser Speicherkapazitaeten fuer die einzelnen Tasks.) Die Implementierung der Objektverwaltung haengt von der jeweiligen technischen Ausstattung und dem zugrunde gelegten Standard-Betriebssystem ab (auf das RTX.014 gewissermassen "aufgesetzt" ist).

4.4. Laufzeitumgebung RTE.014

Dieser Komplex organisiert den praktischen Betrieb des System.014. Dazu gehoeren beispielsweise das Zu- und Abschalten von Benutzern, das Pruefen von Berechtigungen usw.
Der Zugang zur gespeicherten Information wird einerseits durch entsprechende globale Angaben in den Objekten selbst (genauer: in den Objektdeskriptoren) und andererseits durch eine spezifische Systemrelation zwischen den betreffenden Benutzern, Objekten und individuellen Berechtigungen geregelt.
Fuer jedes Objekt kann angegeben werden:
- ungehinderter Zugriff oder eingeschraenkter Zugriff
- bei eingeschraenktem Zugriff die Berechtigung, und zwar entweder als globale Berechtigung in Bezug auf Benutzerklassen oder als spezifische Berechtigung im Rahmen der genannten Systemrelation.

Die Berechtigungen werden getrennt gefuehrt fuer blosses Lesen, fuer das Aendern des Inhaltes und fuer strukturelle Aenderungen. Es sind folgende Benutzerklassen vorgesehen:

- Systemexperte (Master of System)
- Wissensexperte (Knowledge Expert)
- Wissensverwalter (Knowledge Supervisor)
- Betriebsverwalter (Operation Supervisor)
- Professioneller Programmierer (Professional Programmer)
- Anwender 1...4.

Fuer die Anwender 1...4 gibt es noch die Moeglichkeit der Priorisierung (Anwender 1P...4P). Diese werden vorrangig mit Rechenzeit bedient. Weiterhin kann fuer jeden Benutzer festgelegt werden, ob er an bestimmte Terminals gebunden ist oder nicht.

Im einzelnen sind den Benutzerklassen folgende Berechtigungen zugeordnet:

Systemexperte: Nutzung saemtlicher Moeglichkeiten des System.014

Wissensexperte: strukturelle Aenderungen der Wissensbasis (Erweitern, modifizieren usw.)

Wissensverwalter: Einbringen von Vorschlaegen zu strukturellen Aenderungen der Wissensbasis; diese Vorschlaege werden gesammelt und im Stapelbetrieb in die Wissensbasis ueberfuehrt

Betriebsverwalter: An- und Abschalten von Benutzern, Aendern von Berechtigungen, Eingabe von Uhrzeit und Datum, Ausloesen des Stapelbetriebes (etwa um Wissensvorschlaege in die Wissensbasis aufzunehmen), Erstellen von Sicherheitskopien usw.

Professioneller Programmierer: Einbringen von global nutzbaren COIN.014- Programmen unter Nutzung aller Moeglichkeiten von COIN.014

Anwender 1: Nur Abfragen der Wissensbasis

Anwender 2: Abfragen der Wissensbasis mit Erlaubnis, Inhalte zu aendern

Anwender 3: wie Anwender 1, aber mit Nutzung privater Objekte und COIN.014 (fuer private Programme)

Anwender 4: wie Anwender 2, aber mit Nutzung privater Objekte und COIN.014 (fuer private Programme).

Jeder Benutzer wird in einer Systemrelation gefuehrt, die u. a. enthaelt:
- den Namen
- implementierungsspezifisch weitere Personal- Information (Struktureinheit, Telephonnummer usw.)
- die Benutzerklasse
- ein Kennwort, das nur der Betriebsverwalter aendern darf
- eine Signatur, die der Benutzer selbst beliebig aendern kann (dazu sind Kennwort und alte Signatur einzugeben).

Bei Aenderungen in Struktur und Inhalt der Wissensbasis werden Zeit, Benutzer und Signatur sinngemaess vermerkt.

Im Rahmen von RTE.014 sind verschiedene Registrierungs- und Statistikfunktionen anschaltbar, z. B. das Registrieren aller Zugriffe, das Ermitteln von Zugriffshaeufigkeiten, die zeitliche Auslastung des gesamten Systems usw.).

Weiterhin gehoert zu RTE.014 ein Notizsystem, das ueber die Benutzeroberflaeche CASE.014 erreichbar ist und das es ermoeglicht, Hinweise zum Systembetrieb an die Benutzer zu senden sowie Beschwerden, Wuensche, pauschale Fehleraussagen usw. zu erfassen und auszuwerten.

In Mehrplatz- Konfigurationen werden die Terminals von der Systemconsole aus aktiviert bzw. deaktiviert. Der Betriebsverwalter ist dabei in der Lage, Berechtigungen bestimmter Benutzer (an bestimmten Terminals) temporaer zu aendern.

Schliesslich obliegt RTE.014 das Erstellen von Sicherheitskopien sowie der Datenaustausch mit anderen .014- Installationen und mit Fremdsystemen.

4.5. Kommerzieller Interpreter COIN.014

Dieser Komplex ist vorgesehen, um Anfragen an die Wissensbasis zu formulieren und Anwendungsprogramme zu erstellen.
In manchen (einfachen) Implementierungen kann die Moeglichkeit zum Programmieren fehlen; es handelt sich dann um schluesselfertige Systeme, die nicht fuer die Programmierung durch Anwender vorgesehen sind. Zwecks Programmaenderungen oder Erweiterungen ist dann stets auf die der Implementierung zugrunde liegende Programmiersprache (etwa C, BUSINESS BASIC usw.) zurueckzugreifen.

Mit COIN.014 erstellte Programme sind ueblicherweise private Programme des jeweiligen Benutzers. Sie koennen aber auch anderen Benutzern zur Verfuegung gestellt werden.
Benutzer mit der Berechtigung "Professioneller Programmierer" koennen globale Programme in das System einbringen; sie duerfen dazu alle Moeglichkeiten von COIN.014 ausnutzen.

Die sprachlichen Mittel von COIN.014 koennen auch zur Formulierung von Anfragen an die Wissensbasis, zu Berechnungen usw. verwendet werden, ohne dass ausdruecklich ein Programm geschrieben werden muesste (vergleichbar mit Anweisungen ohne Statement- Nummern in BASIC).

Das grundlegende Modell eines Programms in COIN.014 ist die Prozedur, die allgemein als Invocationsregel fuer die Wissensbasis aufgefasst wird. Jede Prozedur ist durch ihren Bezeichner sowie die Parameterliste gekennzeichnet. Prozeduren, die nur ein einziges Resultat- Objekt haben, heissen Funktionen. Die Prozedur selbst ist entweder ein Objekt der Wissensbasis oder ein privates Objekt des jeweiligen Benutzers. Das blosse Angeben einer vorhandenen Prozedur ist fuer deren Ausfuehrung hinreichend.
Die uebliche Schreibweise fuer eine Prozedur ist die Praefix-Notation P(a,b...), und eine Zuweisung hat die Form

x:= P(a,b...).

Fuer bestimmte elementare Operationen ist die gewohnte Infix-Notation vorgesehen, z. B. fuer die Addition als x:= a + b.

Dies betrifft im einzelnen:
- die Addition: +
- die Subtraktion: -
- die Multiplikation: *
- die Division: /
- die Bildung der Potenz: **
- Vergleiche: =; <; >; <=; >=; <> (ungleich)
- logische bzw. Mengenoperationen: AND; OR; NOT; XOR; EQU
- Verzweigungen: if... then... else...
- Laufanweisungen: for... to... step... next.

Fuer die genannten Funktionen ist auch die Praefix- Notation unter Nutzung der systeminternen Designatoren angebbar; z. B. statt x:= a + b x:= ADD (a,b).

COIN.014 stellt fuer alle Aspekte des Verarbeitungsprozesses Mittel bereit, die eine "abstrakte" Formulierung erlauben. Die grundlegenden Abstraktionen sind im folgenden angefuehrt:

1. die Operation
   Eine Operation ist lediglich durch ihre Wirkung gekennzeichnet, d. h. durch die aus den Argumenten gebildeten Resultate. Dabei ist es voellig belanglos, auf welche Weise die Argumente bereitgestellt und wie die Resultate weiterverwendet werden.

2. die Invocation (Beschaffung von Argumenten)
   Das Resultat einer Invocation ist, das das betreffende Objekt (bzw. eine Komponente des Objekts) zur weiteren Verarbeitung bereitsteht.

3. die Zuordnung von Resultaten (Assignment)
   Im Ergebnis einer Zuordnung wird ein Resultat einem Objekt zugewiesen. Die uebliche Form der Zuweisung ist die im Sinne des Lambda- Kalkuels:
   Es wird nicht der neue Inhalt abgespeichert, sondern der Ablauf zusammen mit den aktuellen Objektreferenzen. Bei jedem neuen Zugriff zum Resultat- Objekt wird deshalb der Wert mit den dann aktuellen Parametern neu ermittelt.
   Diese Zuordnung wird mit dem Zeichen := symbolisiert.
   Das Zeichen "::" veranlasst hingegen das Kopieren des Resultats in das betreffende Objekt.

4. die Selektion (Auswahl von Komponenten eines Objekts)
   Das Ergebnis einer Selektion ist, dass die selektierte Komponente fuer die Verarbeitung bereitsteht. Wie diese Komponente ermittelt wird, bestimmt der jeweilige Selektor. Haeufig gebrauchte Selektoren sind vordefiniert. Die uebliche Selektion besteht in der Angabe des Zugriffspfades. Dabei sind die einzelnen Objekte durch Punkte zu trennen (handelt es sich bei den Objekten um Mengen, so steht der Punkt als Abbreviatur fuer die Durchschnittsoperation). So sind etwa
   - Gehalt.Meier
   - Gehalt AND Meier
   - AND (Gehalt, Meier)

   zulaessige Ausdruecke fuer ein und dieselbe Selektion.

5. die Iteration
   Die Iteration dient zum Bereitstellen von Objekten bzw. Komponenten innerhalb von Schleifen (z. B. for- loops). Wie das jeweilige naechste Objekt bzw. die jeweilige naechste Komponente ermittelt werden, bestimmt der betreffende Iterator. Einige sind vordefiniert, es koennen aber auch Iteratoren in Anwendungsprogrammen formuliert werden.
   Beispielsweise stellt das Sprachkonstrukt

   <u>for</u> A <u>to</u> B <u>step</u> C
   .....
   .....
   <u>next</u> A

   einen vordefinierten Iterator dar.

   Der Formulierung eigener Iteratoren liegt folgendes Schema zugrunde:

   - die Deklaration des Iterators:

     <u>iterator</u> A (Parameter x,y...) Ausdruck

(Prozedur, die den Iterationsablauf beschreibt)

end

- die Anwendung des Iterators:

A (e,f,g..)	Aufruf des Iterators (mit aktuellen Parametern)
....	
....	
....	Ausdruecke des loop body
....	
yield A	Auswertung des Iterators

6. die Konstruktion (Aufbau von Objekten)
   Aus gegebenen Objekten (die Komponenten anderer Objekte sein koennen) lassen sich neue Objekte aufbauen.
   Dabei ist es unwesentlich, auf welche Weise die Argumentobjekte bereitgestellt werden.

7. die Ausnahmen (Exceptions)
   Spezifische Bedingungen, die waehrend des Programmablaufs auftreten, fuehren zur Signalisierung von Ausnahmen.
   Einige Ausnahmebedingungen sind vordefiniert, andere koennen in Anwendungsprogrammen formuliert werden.

COIN.014 wirkt interpretativ, setzt aber die eingegebenen Statements in kompakt codierte Befehlsfolgen um. Eine COIN.014-Prozedur ist im allgemeinen Fall ein heterogenes zusammengesetztes Objekt, das aus einer Zugriffstabelle (ACCESS REFERENCE TABLE ART) und wenigstens einer Befehlsfolge besteht. Die ART enthaelt die Objektidentifier fuer alle Parameter, Konstanten, Erweiterungen fuer bestimmte Befehle, Steuerworte usw. Beim Aufruf der Prozedur wird die ART auf den Stack der jeweiligen Task kopiert (sie bildet den "stack frame" bzw. "activation record" des Prozeduraufrufs) und mit den aktuellen Parametern versorgt. Der freie Bereich des Stack (ab Ende des "activation record") bildet dann den aktuellen Arbeitsbereich fuer die Prozedur.
Die ART kann maximal 4096 32- bit- Worte aufnehmen.
Die Befehle beziehen sich ausschliesslich auf die ART (bzw. - genauer gesagt - den aktuellen "activation record").
Alle Befehle sind 16 bit lang und bestehen aus einem 4 - bit-Operationscode und einem 12- bit- Direktwert (finite Ordinalzahl).
Jede Befehlsfolge kann maximal 4096 Befehle umfassen.

Es gibt folgende Befehlstypen:

Vordefinierte Systemoperationen (PREDEFINED SYSTEM OPERATOR; FC 0)

Es wird eine Operation aus der Menge der vordefinierten allgemeinen Systemoperationen ausgefuehrt. Beispiele: Starten und Beenden von Ablaeufen, Speicherverwaltung, Eintritt in Programme, Verlassen von Programmen usw.).
Dazu gehoeren auch die ueblichen arithmetischen und logischen Operationen, das Testen von Bedingungen usw. sowie jene elementaren Invocationsregeln, die in ASK.014 vorgesehen sind.
Des weiteren werden ueber solche Befehle die Funktionen der Realzeit- Executive RTX.014 aufgerufen.

Anwendungs- Prozeduren (PROCEDURAL OPERATOR; FC 1)

Es wird eine Operation aus der Menge der global verfuegbaren anwendungsspezifischen Prozeduren ausgefuehrt.

Zugriffe zu Argumenten (INVOCATION; FC 2)

Die Ordinalzahl im Befehl bezeichnet eine Position im activation record. Handelt es sich um ein Objekt, das seinem Typ nach als Argument fuer Operationen in Frage kommt, so wird dessen Descriptor auf den Stack transportiert. Handelt es sich um einen Iterator, so wird dieser fuer die nachfolgende Anwendung initialisiert.

Zuordnung (ASSIGNMENT; FC 3)

Die Ordinalzahl im Befehl bezeichnet eine Position im activation record. Diese Position wird mit dem Objektidentifier geladen, der dem Objekt in der ersten Position des Stack entspricht. War bereits ein Objektidentifier in dieser Position des activation record vorhanden, so wird er dereferenziert.

Modifikation (MODIFICATION; FC 4)

Die Ordinalzahl im Befehl bezeichnet eine Position im activation record. Das damit identifizierte Objekt (bzw. die Komponente eines Objekts) wird mit dem Wert ueberladen, der durch die erste Position des Stack gegeben ist.

Finiter Direktwert (FINITE IMMEDIATE; FC 5)

Die 12- Bit- Ordinalzahl im Befehl wird als Direktwert auf den Stack gebracht.

Transfiniter Direktwert (TRANSFINITE IMMEDIATE; FC 6)

Die Ordinalzahl im Befehl bezeichnet eine Position im activation record. Deren Inhalt wird als 32- bit- Direktwert interpretiert und auf den Stack gebracht.

Auswertung (YIELD; FC 7)

Die Ordinalzahl im Befehl bezeichnet eine Position im activation record. Die betreffende Position enthaelt entweder ein Steuerwort oder einen Objektidentifier fuer eine Steuerabstraktion (Selektor, Iterator, Lambda- Ausdruck). Es wird der entsprechende aktuelle Wert ermittelt.

Ausfuehrung (EXECUTE; FC 8)

Die Ordinalzahl im Befehl bezeichnet eine Position im activation record. Diese enthaelt entweder einen Objektidentifier oder ein Steuerwort, wodurch eine Operation beschrieben wird. Die Operation wird ausgefuehrt.

Bedingte Ausfuehrung (EXECUTE IF TRUE; FC 9)

Die Wirkung gem. FC 8 wird ausgefuehrt, wenn fuer die erste Position im Stack eine (zuvor selektierte oder durch Testoperationen ermittelte) Bedingung den Wert TRUE hat. Ansonsten wird der folgende Befehl abgearbeitet.

Bedingte Ausfuehrung (EXECUTE IF FALSE; FC 10)

Wirkung analog zu FC 9; Ausfuehrung dann, wenn die Bedingung den Wert FALSE hat.

Verzweigung (BRANCH; FC 11)

Die Ordinalzahl im Befehl waehlt den naechsten Befehl in der aktuellen Befehlsfolge des Programms aus.

Bedingte Verzweigung (BRANCH IF TRUE; FC 12)

Die Verzweigung gem. FC 11 wird nur ausgefuehrt, wenn die Bedingung den Wert TRUE hat. Ansonsten wird der folgende Befehl abgearbeitet.

Bedingte Verzweigung (BRANCH IF FALSE; FC 13)

Wirkung analog zu FC 12; Verzweigung dann, wenn die Bedingung den Wert FALSE hat.

Ausnahmebedingung (RISE EXCEPTION; FC 14)

Die Ordinalzahl im Befehl bezeichnet eine Ausnahmebedingung aus der Menge aller Ausnahmebedingungen (sowohl der im System vordefinierten als auch der zusaetzlich von Anwendern vorgesehenen). Fuer diese Bedingung wird der "innerste" Behandler gesucht und gestartet. Die Suche beginnt im activation record der aktuellen Prozedur; sie setzt sich - falls dort erfolglos - fort im activation record der aufrufenden Prozedur bis hin zum System selbst.

Reserviert (RESERVED; FC 15)

Dieser Operationscode ist fuer kuenftige Erweiterungen reserviert.

Hinweise zu Befehlen und typischen Ablaeufen:

1. YIELD- Befehl
Enthaelt der acitvation record in der betreffenden Position einen Objektidentifier, so wird dieser wie folgt ausgewertet:
- Objekt mit zugewiesenem Wert: keine Wirkung (NO OP)
- Lambda- Ausdruck: Berechnung des Wertes
- Prozedur zur Iteration bzw. Selektion: Aufruf; berechnete Werte werden zurueckgegeben.

Enthaelt der activation record ein Steuerwort fuer einen intern definierten Iterator bzw. Selektor, so wird dieses ausgewertet, und die aktuellen Werte werden zurueckgegeben.

2. EXECUTE- Befehle
Enthaelt der activation record in der betreffenden Position einen Objektidentifier, der eine Prozedur bezeichnet, so wird diese im Sinne eines Unterprogramms ausgefuehrt.
Alternativ dazu kann der activation record ein Steuerwort enthalten. Dieses kann beschreiben:
- eine Verzweigung zu einem Befehl in einer anderen Befehlsfolge des Programms
- den Code des auszufuehrenden Befehls ("Befehlssubstitution")
- einen erweiterten Befehl, der aus einem Operationscode und weiteren Steuerbits besteht.

3. BRANCH- Befehle
Es kann nur zu Befehlen der aktuellen Befehlsfolge verzweigt werden (andernfalls ist EXECUTE mit einem entsprechenden Steuerwort zu verwenden).
Die Position des Folgebefehls wird bei der Verzweigung gerettet, so dass eine Rueckkehr (im Sinne eines Unterprogrammaufrufs) moeglich ist. Dies wird durch die angesprungenen Befehlsfolge gesteuert:
- ein Befehl RETURN (FC 0) veranlasst die Rueckkehr zum Folgebefehl
- ein Befehl BEGIN loescht die Rueckkehrmoeglichkeit (Uebergang zum Ausgangsniveau)
- ein Befehl END veranlasst das Beenden des gesamten Programms.

4. Iteration
Diese wird in ueblichen Programmiersprachen durch "for- loops" beschrieben, so etwa (Sprache Ada) in der Form

<u>for</u> X <u>do</u> A <u>end</u>

Dabei ist X ein Iterator, der den Schleifenkoerper (loop body) A mit aktuellen Argumenten versorgt und ueber das Verlassen der Schleife entscheidet. Ein solcher Ausdruck wird in einem COIN.014-Programm wie folgt codiert:

```
invoke  X        -- Initialisieren des Iterators
execute A        -- Ausfuehren des "loop body"
yield   X        -- Auswerten des Iterators
```

Der YIELD- Befehl hat dabei folgende Wirkung: Ist die Iteration noch nicht beendet, so wird zum vorhergehenden Befehl verzweigt, andernfalls zum nachfolgenden.
Fuer X und A gibt es jeweils eine Position im activation record.

Jede dieser Positionen kann wahlweise enthalten:
- einen Objektidentifier
- ein spezifisches Steuerwort (Befehlssubstitution, vordefinierte elementare Iteratoren, erweiterte Befehle)
- ein Steuerwort, das den Eintritt in eine andere Befehlsfolge des Programms bewirkt (Iterator bzw. "loop body" sind innerhalb des Programms separat codiert).

5. Verzweigungen if...then...else
Die uebliche Verzweigung der Form

```
if A then B
  C                    -- Folgeanweisung
```

wird wie folgt codiert:

```
Testen der Bedingung
EXECUTE IF TRUE    -- Befehlsfolge fuer B
C
```

Die Befehlsfolge fuer B endet mit einem RETURN- Befehl.

Die erweiterte Verzweigung der Form

```
if A then B else C
  D                    -- Folgeanweisung
```

wird wie folgt codiert:

```
Testen der Bedingung A
EXECUTE IF TRUE B
EXECUTE IF FALSE C
D
```

Die Befehlsfolge fuer B endet mit einem Befehl RETURN & SKIP (der zum uebernaechsten Befehl der aufrufenden Befehlsfolge zurueckkehrt); die Befehlsfolge fuer C endet mit einem RETURN-Befehl.

Ineinandergeschachtelte Konstrukte (if...then...else if... usw.) werden durch Ineinanderschachteln entsprechender Befehlsfolgen codiert.

4.6. Benutzeroberflaeche CASE.014

Dieser Komplex gewaehrleistet den Bedienungskomfort des System.014 fuer die Benutzer.

Unter Nutzung der funktionellen Eigenschaften des logischen Terminals werden folgende Moeglichkeiten geboten:

- elementare Anfragen
- komplexe Anfragen unter Nutzung von COIN.014
- Programmierung mit COIN.014
- Funktionsausloesung
- Hilfe
- selektive Darstellung umfangreicher Information
- Sonderfunktionen (beschleunigte Anfragen u. a.).

Diese Moeglichkeiten werden im folgenden naeher erklaert:

1. Elementare Anfragen

Anfragen koennen sowohl die Wissensbasis als auch private Objekte des Benutzers betreffen.
Objekte der Wissensbasis werden durch Angabe ihrer Designatoren bezeichnet. Aufeinanderfolgende Objekte sind durch jeweils einen Punkt voneinander zu trennen (Doppelpunkt bei privaten Objekten). Private Objekte werden durch einen vorangestellten Doppelpunkt von den Objekten der Wissensbasis unterschieden.
Die Bezeichner fuer Objekte der Wissensbasis duerfen bis zu 63 Zeichen lang sein. Sie muessen aus wenigstens einem Wort bestehen und duerfen nur aus Worten und Zahlen im Sinne der ueblichen Umgangssprache aufgebaut werden. Es sind alle Buchstaben und Ziffern zugelassen. Die Gross- bzw. Kleinschreibung hat bei der Eingabe keine Bedeutung (CASE.014 wandelt Kleinbuchstaben in Grossbuchstaben), der Ausgabe liegt jedoch stets eine Darstellung mit Kleinbuchstaben und (bei Substantiven) grossen Anfangsbuchstaben zugrunde. Als Trennzeichen sind Leerzeichen (eines pro Trennung) und Unterstreichungszeichen zugelassen.
Qualifizierer und private Objekte koennen auch andere Zeichen enthalten; solche Angaben sind in Anfuehrungszeichen zu setzen. (auch Bezeichner im o. g. Sinne duerfen in Anfuehrungszeichen gesetzt werden.) Bezeichner privater Objekte duerfen maximal 15 Zeichen umfassen, wobei das erste Zeichen ein Buchstabe sein muss.

Beispiele:

Korrekte Bezeichner fuer die Wissensbasis sind:

Einkommen

STATUS

STATUS.meier

Erzeugnis 234

ROHR_Durchmesser_120

Falsch sind hingegen:

ST12x

Erzeugnis 234

ROHR/Durchmesser 120

Korrekte Bezeichner fuer private Objekte sind:

:lohn

:112a

:ERZEUGNIS 13

:"a/b?? #"

:Geh:Ber:Name=Meier

Falsche Bezeichner fuer private Objekte sind:

:12

:LOHN PRO ERZEUGNIS

Die Reihenfolge der Angaben ist an sich beliebig; sie werden von ASK.014 in eine definierte Reihenfolge gebracht, die durch Fundamentalkategorie, semantische Kategorie und Ordnungszahl bestimmt wird. Diese Reihenfolge wird ausgegeben.

Es ist weiterhin moeglich, in Bezeichnern unbestimmte Positionen anzugeben:
- ein Fragezeichen bedeutet, dass an der betreffenden Position alle Buchstaben bzw. Ziffern einzusetzen sind
- zwei aufeinanderfolgende Punkte bedeuten, dass die nachfolgenden Buchstaben oder Ziffern beliebig zu ergaenzen sind.

Beispiele:

Status.M??er — liefert den Status aller Personen, deren Name mit "M" beginnt und mit "er" endet und dazwischen 2 weitere Buchstaben enthaelt. Konkret waere das z. B. "Maier", "Mayer", "Meier", "Meyer" usw.

Status.Me.. — liefert den Status aller Personen, deren Name mit "Me" beginnt, also z. B. "Meier", "Meyer", "Meseritscher" usw.

In den Anfragen koennen logische und arithmetische Operationen enthalten sein. Im einzelnen kennzeichnen:

- "+"; "-"; "*"; "/"; "**" Addition, Subtraktion, Multiplikation, Division, Erhebung zur Potenz
- "="; "<"; ">"; "<="; ">="; "<>" die ueblichen Vergleichsoperationen
- ":=" die Zuweisung im Sinne des Lambda- Kalkuels
- "::" die Zuweisung als Kopie (von Anwendern nur fuer private Objekte angebbar)
- "<aaa..bbb>" ein geschlossenes Intervall (alle Werte zwischen aaa und bbb einschliesslich der Grenzen)
- ">aaa..bbb<" ein offenes Intervall (alle Werte ausserhalb des angegbenen Intervalls
- "--" den Beginn eines Kommentars, der an sich beliebig sein kann.
- "," das Komma in Zahlenangaben.

Qualifizierer und Quantifizierer sind jeweils als Kombination von Bezeichner, relationalem Operator und aktuellem Wert anzugeben, z. B.:

Einkommen.Angestellter < 934,00

Anschrift.Ort = Falkenstein

Lieferbezeichnung = "a12 ef/18-5".preis_pro_tonne

Arbeiter. Abteilung = 23. Leistung >200..400<

Zwecks besserer Lesbarkeit koennen runde Klammern verwendet werden.

Grundsaetzlich liefert eine Anfrage das gesamte im System enthaltene zugehoerige "Wissen", wobei vor der Ausgabe der Umfang bewertet wird. Demgemaess wird die Information entweder direkt ausgegeben oder es wird eine Auswahl angeboten.

Wird mit der Anfrage nicht eindeutig eine Individuen- Komponente selektiert, so wird die Informationseinheit mit der jeweils hoechsten Ordnungszahl zuerst ausgewaehlt.

Relationen koennen auch - statt durch ihern eigenen Designator - durch Notation der konstituierenden Objekte in Form eines Kreuzproduktes angegeben werden, z. B. liefert die Anfrage

STATUS * Einkommen

alle Relationen, in denen diese Designatoren als konstituierende Objekte vorkommen.

Durch das Fragezeichen ist zusaetzliche Information zu einzelnen Designatoren zugaenglich. Beispiele:

Angestellter?	liefert den Mitteilungsbereich zu diesem Designator, der z. B. aktuelle Aenderungs-Hinweise enthalten kann
Lohn??	liefert die (feste) verbale Beschreibung zu diesem Designator
einkommen???	liefert die metasprachliche (strukturell-deskriptive) Information zu diesem Designator, also Fundamentalkategorie, semantische Kategorie, Ordnungszahl usw.

Ein vorangestelltes Fragezeichen bewirkt, dass lediglich eine JA/NEIN- Aussage geliefert wird, d. h. es wird angezeigt, ob die angefragte Bedingung erfuellt ist, ob das bezeichnete Objekt existiert usw. Beispiel:

?Meier.Einkommen>1200,00

liefert eine JA/NEIN- Aussage darueber, ob es Personen mit Namen "Meier" gibt, deren Einkommen 1200,00 M uebersteigt.

Mit einem vorangestellten Punkt kann man die vorhergegangene Anfrage weiter detaillieren. Beispiel:

1. Anfrage

Meier.Einkommen

2. Anfrage

.Angestellter

Diese ist das Aequivalent fuer

Meier.Einkommen.Angestellter

(d. h. es wird das Einkommen aller Angestellten mit Namen "Meier" ausgegeben).

Im Sinne einer inversen Operation kann man mit einem vorangestellten Bindestrich die voraufgegangene Anfrage verallgemeinern. Beispiel:

1. Anfrage

Erzeugnis 23.X_Dorf.Januar

2. Anfrage

-X_Dorf

Dies ist aequivalent zu

Erzeugnis.Januar

2. Komplexe Anfragen unter Nutzung von COIN.014

Dazu berechtigte Benutzer koennen die Sprachkonstrukte von COIN.014 umfassend zu komplexen Anfragen nutzen.
In manchen (elementaren) Implementierungen ist diese Moeglichkeit nicht vorgesehen.
Eine der Besonderheiten von COIN.014 ist es, dass elementare Operationen je nach den Objekten, auf die sie angewendet werden, entsprechende Resultate hervorbringen. So wirken arithmetische Operatoren auf Listen von Quantifizierern so, dass durch zeilenweise unabhaengige Verknuepfungen eine neue Liste entsteht.

3. Programmieren mit COIN.014

Programme werden so erstellt, dass zunaechst das gewuenschte Programm mit reservierten Schluesselworten definiert wird. Dazu sind der Designator und die Parameter anzugeben. Parameter koennen auch mit "default"- Werten deklariert werden (unterlaesst dann das aufrufende Programm die Uebergabe eines solchen Parameters, wird der "default"- Wert substituiert).
Innerhalb der Erstellung eines Programms koennen die direkt wirkenden elementaren Funktionen von COIN.014 genutzt werden. Das Umschalten wird durch Funktionstasten gesteuert.

Das Programmieren beginnt durch eine Funktionsauswahl.
Es wird zunaechst die Prozedur definiert:

```
function NETTOLOHN (A:BRUTTOLOHN; B:STEUERSATZ;
                    C:SONDER_ABZUEGE.......) return
                    quantifier: 4,2;
```

begin

-------- hier folgt der eigentliche Prozedurkoerper
----

waehrend des Programmierens wird Information zum Designator BRUTTOLOHN gewuenscht:

bruttolohn??
-- Es erscheinen die gewuenschten Angaben (in einem besonderen Fenster).

Dann wird weiter programmiert.

Jetzt ist eine Hilfsrechnung erforderlich:

435 + 16 + 99 = 550

Dann wird weiter programmiert.

------

end Dies schliesst das Programm ab.

Durch Betaetigen einer Funktionstaste wird der Programmiermodus verlassen. Der Programmtext wird in eine Befehlsfolge mit ART umgesetzt und unter dem angegebenen Designator gespeichert. Wurden Syntaxfehler festgestellt, so wird eine Korrekturmoeglichkeit geboten.
Anschliessend ist das Programm durch seinen Designator (zusammen mit aktuellen Parametern) aufrufbar.

4. Funktionsausloesung

Das logische Terminal umfasst mindestens 10 Funktionstasten, deren aktuelle Belegung sich teilweise in Abhaengigkeit vom aktuellen Funktionszustand aendert. Diese Belegung wird in einem Fenster am rechten Bildschirmrand angezeigt.
Wesentliche Funktionen sind:

- Rueckkehr in den Grundzustand
- Anzeigen von Hilfs- Information
- Uebergang in den Programmierzustand
- Verlassen des Programmierzustandes
- Erstellen einer Hardcopy
- Auswahl der selektierten Information zur Darstellung

In den einzelnen Zustaenden sind manche Funktionstasten anders belegt. So koennen im Programmierzustand die ueblichen reservierten Schluesselworte durch Betaetigen von jeweils einer Funktionstaste eingegeben werden.

5. Hilfe

Durch Betaetigung einer Funktionstaste kann jederzeit zusaetzliche erklaerende Information angefordert werden. Diese Information erscheint in einem Fenster in der rechten oberen Ecke des Bildschirms. Dabei aendert sich die Belegung einiger Funktionstasten, wodurch es moeglich wird, weitere Hilfs- Information abzurufen, die Darstellflaeche dafuer auszudehnen usw.

6. Selektive Darstellung umfangreicher Information

CASE.014 bzw. COIN.014 bewerten die Groesse von Objekten, die ausgegeben bzw. erzeugt werden. Von bestimmten Groessenordnungen an werden Sondermassnahmen eingeleitet. COIN.014- Programme akzeptieren an sich sehr grosse Objekte. Die Kontrolle richtet sich in diesem Fall auf die Moeglichkeit des Unterbringens im Rahmen der gegebenen Speicherausstattung. Wird versucht, mit COIN.014 Objekte zu erzeugen, die "verdaechtig gross" sind, so erscheinen entsprechende Warnungen (u. U. bereits beim Erzeugen des Programms, falls dies da bereits erkennbar ist).
Bei der Ausgabe muss CASE.014 auf die Eigenschaften der angeschlossenen Geraete Ruecksicht nehmen. Deshalb ist die Groesse von Objekten, die ohne weiteres ausgegeben werden, durch die Anzeigekapazitaet von Bildschirmen, die Druckbreite von Druckern, die Speicherkapazitaet von Disketten usw. begrenzt.
Die Einzelheiten werden implementierungsspezifisch sein und sich zudem auf Grund praktischer Erahrungen und Erfordernisse aendern. Einige Vorstellungen seien im folgenden skizziert:

- Objekte, die auf eine Diskette, eine Druckseite oder in das vorgesehene Fenster des logischen Terminals ohne weiteres passen, werden sofort ausgegeben.
- Objekte, die die Groesse des vorgesehenen Fensters mehrmals (z. B. viermal) belegen, werden direkt angezeigt (es wird zunaechst der erste Ausschnitt dargestellt, und es werden Funktionstasten so belegt, dass es moeglich ist, die anderen Ausschnitte auszuwaehlen).
  Wesentlich fuer diese Art der Darstellung ist, dass bereits anhand des ersten Ausschnitts die Struktur des gesamten Objekts erkennbar sein muss.
- Fuer groessere Objekte wird zunaechst deren Struktur dargestellt, und es werden entsprechende Auswahlmoeglichkeiten angeboten.
- Soll eine Vielzahl von Objekten zur Ausgabe kommen, so wird zunaechst eine Uebersicht mit Auswahlmoeglichkeiten und Hinweisen dargestellt.

7. Sonderfunktionen

Beispiele fuer solche Funktionen sind:
- beschleunigte Anfragen
- Rechentabellen
- "Taschenrechner"- Funktionen.

Jeder Benutzer kann maximal 10 Anfragen fest vorformulieren und jeder dieser Anfragen eine Ziffer zuordnen. Dann reicht das blosse Eingeben der Ziffer aus, um die gesamte Anfrage auszuloesen.

Beispiel:

Die Anfrage wird formuliert; sie soll unter Ziffer 3 zugaenglich sein:

3-Erzeugnis.Stueckzahl>500.Wert<5,00 --alle Kleinteile mit hohem Bedarf

Dann reicht die einfache Eingabe fuer die Anfrage voll aus:

3

Alle solchen Anfragen werden in einem Menue aufgelistet, das durch Funktionsauswahl zugaenglich ist (dort wird auch der angegebene Kommentar eingetragen).

Prinzipiell stehen fuer jeden Benutzer bis zu 10 Rechenplaene zur Verfuegung, wobei es bis zu 10 fest vorgesehene Algorithmen im Rahmen des Systems gibt, um Rechenoperationen mit diesen auszufuehren (bei manchen Implementierungen muss die Anzahl der Rechen pro Benutzer als Konfigurationsparameter in einer Systemrelation angegeben werden; manche Benutzer koennen somit - z. B. mangels Speicherplatz - von dieser Moeglichkeit ausgeschlossen werden).

Jede Rechentabelle ist durch Eingabe eines Ausrufezeichens und einer nachgestellten Ziffer aufrufbar, z. B.

!3

Ueber Funktionstasten kann jeder Benutzer eine Uebersicht (Menue) seiner Rechentabellen anfordern.
Die Tabellen koennen an sich beliebig belegt werden; Resultate werden sofort nach Eingaben sichtbar.
Ein Benutzer darf keine Tabellen anderer Benutzer veraendern, er kann aber (falls vom jeweiligen anderen Benutzer dazu berechtigt) die fremde Tabelle ansehen und in eine eigene kopieren.
Einfache Beispiele fuer Rechenoperationen sind:

- das spaltenweise Addieren (einschliesslich Bilden von Zwischensummen)
- die zeilenweise Operation "Spalte 3 = Spalte 1 - Spalte 2; Spalte 4 = Prozentwert von Spalte 3 in Bezug auf Spalte 1" (dies ist fuer Soll/Ist- Vergleiche besonders geeignet).

Des weiteren sind vier Variablen A,B,C,D vorgesehen, die Rechenoperationen mit Zahlen sowie Werten von Quantifizierern erlauben. Das Eingeben einer Variablenbezeichnung reicht zur Anzeige des aktuellen Wertes. Die Variablen koennen auch in Anfragen verwendet werden. Beispiele:

A = 12**3

B = Einkommen.Name = Maier * 0,3

A = B * 0,1 + C*(199/0,51)

Liefermenge.Schrauben = "M 6 * 20" < C * 2

4.7. Entwicklungsumgebung DEVSYS.014

Diese wird im allgemeinen durch ein uebliches Betriebssystem mit den zugehoerigen Compilern, Dienstprogrammen usw. gebildet, das implementierungsspezifisch durch spezielle zusaetzliche Dienstprogramme erweitert ist.
Das Entwickeln von Programmen (in anderen Sprachen als in COIN.014) parallel zum normalen Betrieb ist dabei nicht vorgesehen.
Ueblicherweise werden solche Programme auf einem Entwicklungssystem erstellt, ausgetestet und in System.014-Installationen eingebracht.

4.8. Wartungssystem MAINTSYS.014

Dazu gehoeren Testroutinen der Rechnerhersteller sowie spezifische Programme zum Testen der Konfiguration. Weiterhin sind Routinen zum Aufzeichen und Auswerten von Fehlern vorgesehen, die waehrend des normalen Betriebs auftreten. Diese Aufzeichnungen sind ueber CASE.014 zugaenglich. Auch koennen waehrend des normalen Betriebs ueber RTE.014 selektiv Tests fuer bestimmte Terminals, Drucker usw. ausgeloest werden.