Funktionale Modelle in der Konfigurierung

OEGAI 91 KI-NRW 91-9*

Benno Stein Jürgen Weiner

FB 11 – Praktische Informatik
Universität Duisburg
Postfach 10 15 03
D-4100 Duisburg 1

Zusammenfassung

Wir möchten hier das wissensbasierte System MOKON vorstellen, das die Konfigurierung technischer Anlagen unterstützt und hierfür ein funktionales Modell verwendet.

Kern ist eine Modellierung des Wissens in Form von funktionsorientierten Beschreibungen der zu konfigurierenden Komponenten. Durch diese Beschreibung wird ein Abhängigkeitsnetz zwischen den Komponenten aufgebaut, das den Konfigurierungsprozeß steuert. Dieses Abhängigkeitsnetz drückt kausale Beziehungen zwischen den Komponenten aus, die als Angebote und Forderungen interpretiert werden. Ergänzt wird diese funktionale Beschreibung um assoziatives Wissen.

Dieses Papier stellt in einem einleitenden Abschnitt den Modellbegriff sowohl in allgemeiner Form als auch in Zusammenhang mit der Problemstellung Konfigurierung vor. Abschnitt 2 beschreibt das zugrundeliegende funktionale Modell sowie dessen Operationalisierung anhand des Konfigurierungssystems MOKON und zeigt, wie eine Kopplung von Oberflächenwissen mit einem funktionalen Modell realisiert werden kann. Abschließend skizzieren wir den Stand der Arbeiten und nehmen eine Bewertung des Ansatzes vor.

*This work was supported by the "KI-Verbund NRW", founded by the Ministery for Science and Research of North Rhine-Westphalia. A reprint of this paper can be found in the Report Series KI-NRW (Knowledge-Based Software Technology in North-Rhine Westphalia), number 91-9, October 1991.

1 Einleitung

1.1 Modellbegriff

Ein Modell stellt eine Beschreibung dar, welche die Aspekte der Wirklichkeit enthält, die den Modellgestalter interessieren. Solche Aspekte können für ein technisches System die Funktion, das Fehlerverhalten, die Struktur, assoziative Zusammenhänge usw. betreffen oder eine Kombination daraus darstellen.

Bei assoziativen Beschreibungen handelt es sich um Oberflächenwissen, das überwiegend Heuristiken bzw. Daumenregeln zur Lösungsfindung einsetzt. Abgrenzend hierzu soll mit einem tiefen Modell ein System oder Systemausschnitt von seinem Wesen her beschrieben werden. Dabei liegt der Verwendung des tiefen Modellbegriffs häufig ein relatives Verständnis zugrunde: Vergleicht man zwei Modellierungsansätze für den gleichen Sachverhalt, dann wird einer als tiefer bezeichnet, wenn er sich gegenüber dem anderen z.B. durch bessere Erklärungsfähigkeit auszeichnet oder auf ursächlicheren Zusammenhängen basiert.

Unabhängig von dieser relativen Betrachtungsweise soll hier ein Modell dann als tief bezeichnet werden, wenn Ebenen von Wissen unterschiedlicher Qualität existieren und zueinander in Beziehung gesetzt werden können. Steels argumentiert in [7] mit zwei 'Schichten' eines Modells: „Deep expert systems contain two components: One implements the deep knowledge of the domain, that is, the domain model and an inference calculus operating over it, and the other implements the surface knowledge“.

Mit einer assoziativen Beschreibung verbindet man eine gute Performanz, jedoch eine geringe Unterstützung in den Punkten Erklärung und Akquisition; genau umgekehrt argumentiert man bei einer komplexen, kausalen Modellierung. Besteht eine Modellierung – wie oben skizziert – aus mehreren, qualitativ unterschiedlichen Beschreibungsschichten, kann man versuchen, die jeweiligen Vorteile zu verbinden.

1.2 Konfigurierungsprinzipien

In den ersten wissensbasierten Ansätzen zur Konfigurierung bildeten assoziative Beschreibungen den wesentlichen Bestandteil des Konfigurierungswissens, wie zum Beispiel bei dem Konfigurierungssystem R1/XCON [3]. Neben assoziativen Beschreibungen orientieren sich spätere Ansätze häufig an einem Strukturmodell des Systems. Ein Strukturmodell läßt sich – wie in Abbildung 1 dargestellt – durch einen hierarchischen, nicht rekursiven Und-Oder-Graphen beschreiben:

Die Lösungsfindung mithilfe eines Und-Oder-Graphen beruht auf der schrittweisen Zerlegung in Teilprobleme (Und-Verknüpfung), wobei ein Teilproblem durch die Auswahl mehrerer Alternativen verfeinert werden kann (Oder-Verknüpfung). Durch diese Repräsentationsform werden grundsätzliche Entscheidungsmöglichkeiten für das Konfigurierungssystem expliziter dargestellt, als dies durch einen assoziativen Ansatz mit einer unstrukturierten Menge von Regeln möglich wäre. Auf einem solchen Strukturmodell bauen zwei wesentliche Konfigurierungsstrategien auf:

1. TOP-DOWN-Konfigurierung: Der Konfigurierungsprozeß geht vom Wurzelknoten

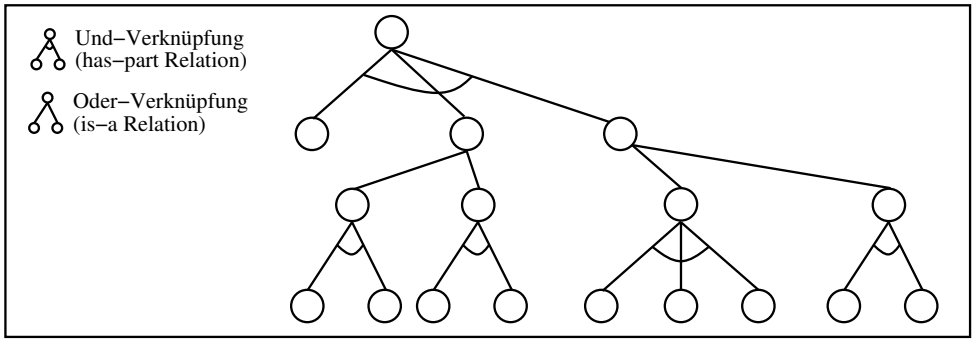

Abbildung 1: *Beispiel für einen Und-Oder-Graph*

aus und verfeinert bzw. zerlegt die Objekte.

2. BOTTOM-UP-Konfigurierung: Falls der Benutzer konkrete Komponenten und Parameter spezifiziert, werden diese zunächst instanziiert und entsprechende Schlußfolgerungen gezogen. Anschließend beginnt eine TOP-DOWN-Konfigurierung mit den bereits instanziierten Objekte als zusätzliche Randbedingung.

Ein typischer Vertreter für diese Vorgehensweise im Konfigurierungsprozeß ist die Entwicklungsumgebung PLAKON [1].

Funktionale Modelle spielen in der Konfigurierung – im Gegensatz zur Diagnose – eine untergeordnete Rolle. Unter dem Begriff des funktionalen Modells wird hier keine Simulation oder verhaltensorientierte Beschreibung verstanden. Gemeint ist, daß die Komponenten über Funktionen beschrieben sind und – abgrenzend zum Struktur- bzw. assoziativen Modell – Beziehungen zwischen den Komponenten ausschließlich über Funktionalitäten hergeleitet werden (müssen).

Durch eine ausschließliche Beschränkung auf Funktionalitäten bei der Beschreibung eines Konfigurierungsproblems können Problemlösungsverfahren entwickelt werden, welche eine eingeschränkte Anwendungsbreite aufweisen, jedoch eine adäquate Unterstützung für spezielle Domänen bieten (vgl. Abschnitt 3.1).

2 Konzept MOKON

Bei MOKON handelt es sich um ein Konfigurierungssystem[1], dessen Kern durch ein funktionales Modell der Anwendungsdomäne gebildet wird. Wir zeigen, wie dieses funktionale Modell um eine assoziative Beschreibungsschicht ergänzt wird, so daß die oben skizzierten Vorteile einer Mehrschichtenarchitektur zum Tragen kommen.

Zunächst stellen wir den Kern von MOKON vor; anschließend beschreiben wir das Zusammenwirken der funktionalen mit der assoziativen Beschreibungsebene.

[1] In diesem Artikel wird von der Problematik der Neukonfigurierung einer technischen Anlage ausgegangen. Darüberhinaus unterstützt MOKON die Aufgabenstellungen Änderungskonfiguration und Konfigurationsprüfung (siehe [8]).

2.1 Funktionales Modell

In diesem Unterkapitel wird zunächst die zugrundeliegende Syntax und Semantik des funktionalen Modells vorgestellt. Der darauffolgende Abschnitt beschreibt eine Operationalisierung dieses Modells (Wissensrepräsentation und Verarbeitung) anhand des Konfigurierungssystems MOKON.

Syntax

Basis dieses Modells ist die Beschreibung der Domäne durch die Mengen K (Komponenten), F (Funktionalitäten) und P (Prädikate). Jeder Funktionalität $f \in F$ ist ein Domänenbereich $dom(f)$, zwei Abbildungen g_F und g_A und ein Prädikat Q zugeordnet:

$$g_F, g_A : dom(f)^* \rightarrow dom(f)$$

$$Q : dom(f) \times dom(f) \rightarrow \{\text{wahr, falsch}\}$$

Zu jedem $k \in K$ gehören zwei Mengen D_k und S_k mit Tupeln der Form (f, x), $f \in F$, $x \in dom(f)$ sowie die Abbildungen:

$$g_{D_k}, g_{S_k} : \mathbf{N} \rightarrow dom(f)$$

Eine Anforderungsdefinition A ist eine Menge von Tupeln (f, x) mit $f \in F$, $x \in dom(f)$. Eine Konfiguration C enthält Elemente (n, k) mit $n \in \mathbf{N}, k \in K$. Eine Konfiguration C ist zulässig, falls sie die nachfolgende Bedingung erfüllt:

$$\forall f : f \text{ kommt in } A \text{ vor oder } f \text{ kommt in } C \text{ vor: } Q(g_F(D_{ges}), g_A(S_{ges})) = \text{wahr}$$

mit

$$D_{ges} := \{x \in dom(f) \mid x = g_{D_k}(n) \text{ mit } (n, k) \in C \wedge (f, \_) \in D_k\} \cup \{y \in dom(f) \mid (f, y) \in A\}$$
$$S_{ges} := \{z \in dom(f) \mid z = g_{S_l}(m) \text{ mit } (m, l) \in C \wedge (f, \_) \in S_l\}$$

Semantik

Hauptbestandteil des Konfigurierungswissens ist eine funktionsorientierte Beschreibung von Komponenten. Dabei wird zwischen Forderungs- und Angebotsfunktionalitäten unterschieden (D_k, S_k). Jede Komponente stellt gewisse Funktionalitäten zur Verfügung und fordert auch welche. Über diese Angebote und Forderungen werden die einzelnen Komponenten zueinander in Beziehung gesetzt:

Die Steckkarten in Abbildung 2 fordern jeweils einen Steckplatz und die Grundplatine stellt Steckplätze bereit. Das Netzteil bietet Strom an, die übrigen Komponenten fordern Strom.

Eine Konfiguration C ist eine Liste von Komponenten mit zugehöriger Mengenangabe. Eine Anforderungsdefinition A ist eine Liste von Funktionalitäten mit Ausprägungen. Die Aufgabe des Konfigurierungsprozesses ist, ausgehend von einer Anforderungsdefinition eine zulässige Konfiguration zu bestimmen. Hierbei ist eine Konfiguration zulässig, wenn für jede Funktionalität f die Gesamtforderung D_{ges} durch das Gesamtangebot S_{ges} überdeckt ist. Das heißt, das zu f gehörige Prädikat $Q(D_{ges}, S_{ges})$ ist erfüllt.

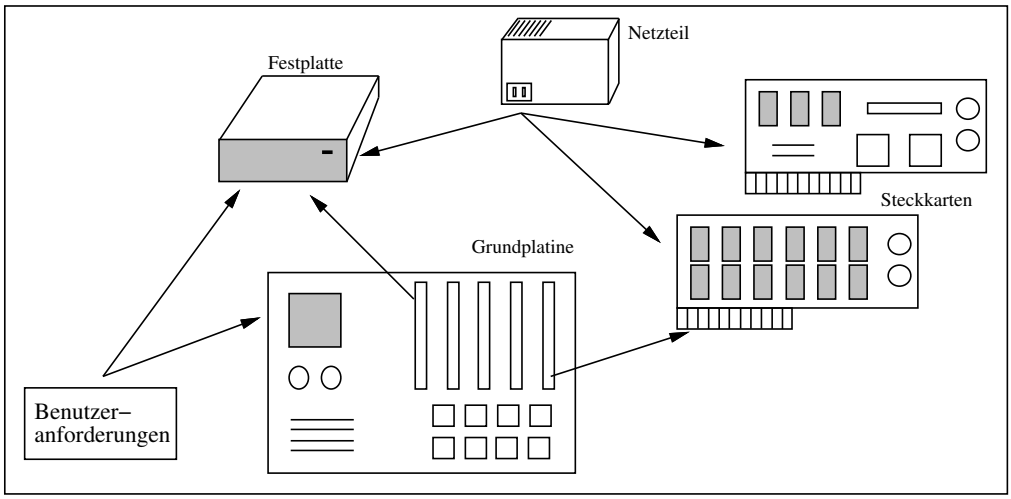

Abbildung 2: *Komponentenbeziehungen*

Zur Berechnung des Gesamtangebots bzw. der Gesamtforderung werden die Funktionen $g_F, g_A, g_{D_k}, g_{S_k}$ benötigt. Hierbei berechnen g_{D_k}, g_{S_k} eine Funktionalitätsausprägung in Abhängigkeit der ausgewählten Komponentenanzahl für *eine* Komponente; die Funktionen g_F, g_A dienen zur Verarbeitung der sich ergebenden Wertemengen gleicher Funktionalitäten *verschiedener* Komponenten. Die Gesamtforderung beinhaltet neben den Komponentenforderungen auch die Werte der Anforderungsdefinition.

2.2 Operationalisierung

In obiger Modellbeschreibung wurde noch nicht geklärt, wie man von einer Anforderungsdefinition A zu einer zulässigen Konfiguration C gelangt. In MOKON besteht der Prozeß des Konfigurierens aus der Propagierung der Angebots- und Forderungsbeziehungen von Komponenten. Diese Vorgehensweise ist mit dem ressourcenorientierten Ansatz von Heinrich [2] vergleichbar. Im folgenden gehen wir detaillierter auf die Wissensrepräsentation und -verarbeitung in MOKON ein.

Wissensrepräsentation

Die Wissensrepräsentation besteht u.a. aus dem attributiven Wissen der Komponenten und der Funktionsbeschreibungen:

- Attributives Wissen einer Komponente
  Dieses Wissen besteht zum einen aus funktionalem Wissen, wie den Forderungen D_k und Angeboten S_k, die sich aus der Funktionalität der Komponente ableiten lassen, zum anderen aus sonstigen Attributen wie Preis, Lagerbestand usw.(vgl. Abb. 3).

- Attributives Wissen einer Funktion
  Hierbei handelt es sich um Wissen, das eine dem kausalen Modell angemessene Behandlung der Funktionalität ermöglicht: Die Abbildungen g_F und g_A einer Funktionalität f legen fest, wie aus einer Liste von Forderungen bzw. Angeboten eine

Bezeichnung	HD−1	Gehäuse−7
Angebot	Plattenkapazität: (Anzahl * 60) Zugriffsgeschwindigkeit: 20	Sicherheitsnorm: USA Analoge Steckplätze: (Anzahl * 10) Digitale Steckplätze: (Anzahl * 14)
Forderung	Stromwert: (Anzahl * 30) Steckplatz: (Anzahl * 1)	...
Preis	500	1100
Lagerbestand	35	20

Abbildung 3: *Komponentenbeschreibung*

Gesamtforderung bzw. ein Gesamtangebot für diese Eigenschaft generiert wird. Mithilfe des zugehörigen Prädikats Q werden die ermittelten Werte verglichen.

Wissensverarbeitung

Abbildung 4 zeigt eine grafische Darstellung der Verarbeitung.

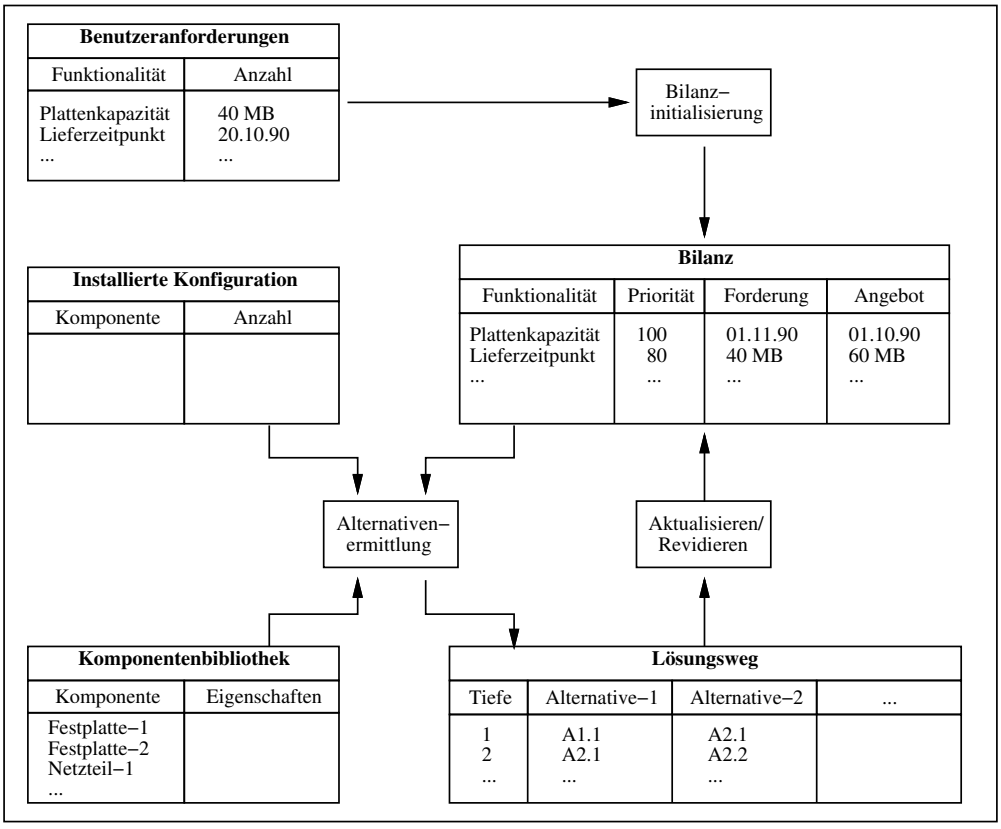

Abbildung 4: *Skizzierung der Verarbeitung*

Ausgangspunkt des Konfigurierungsprozesses ist eine Anforderungsdefinition A, die aus einer Auflistung von Funktionalitäten besteht, welche das zu konfigurierende System erfüllen soll. Mit diesen Funktionalitäten wird eine Bilanz initialisiert, in der Angebote und Forderungen eingetragen werden. Für jede Funktionalitätsforderung innerhalb der Bilanz, der kein geeignetes Angebot gegenübersteht ($P(D_{ges}, S_{ges}) =$ falsch), wird eine Alternativenermittlung angestoßen. Sie versucht eine Alternative aus der Komponenten-

bibliothek zusammenzustellen, welche die offene Forderung erfüllt. Meistens existieren mehrere Alternativen, die eine offene Forderung befriedigen können.
Abhängig von der aktuellen Strategie wird eine Alternative ausgewählt und die Bilanz aktualisiert mit der Konsequenz, daß Angebote und Forderungen, die sich aus den neuen Komponenten ergeben auf der Bilanz nachvollzogen werden müssen. Kann eine Forderung nicht erfüllt werden, so muß im Suchraum zurückgesprungen, d.h. andere Alternativen verwendet werden. Neben der Hill-Climbing-Strategie – lokale Auswahl z.B. der preisgünstigsten oder steckplatzminimalen Alternative – kann der Benutzer den Suchraum interaktiv verwalten.
Der Prozeß des Überprüfens und Aktualisierens der Bilanz wird solange fortgesetzt, bis alle offenen Forderungen erfüllt sind oder festgestellt wird, daß keine Lösung existiert.

In der bisherigen Beschreibung der Wissensverarbeitung wurde offengelassen, in welcher Reihenfolge offene Forderungen abgearbeitet werden. Hierzu zeigt Abbildung 5 ein Abhängigkeitsnetz mit Komponenten (Rechtecke) und Funktionen (Rauten), aus dem sich eine Abarbeitungsreihenfolge ableiten läßt.

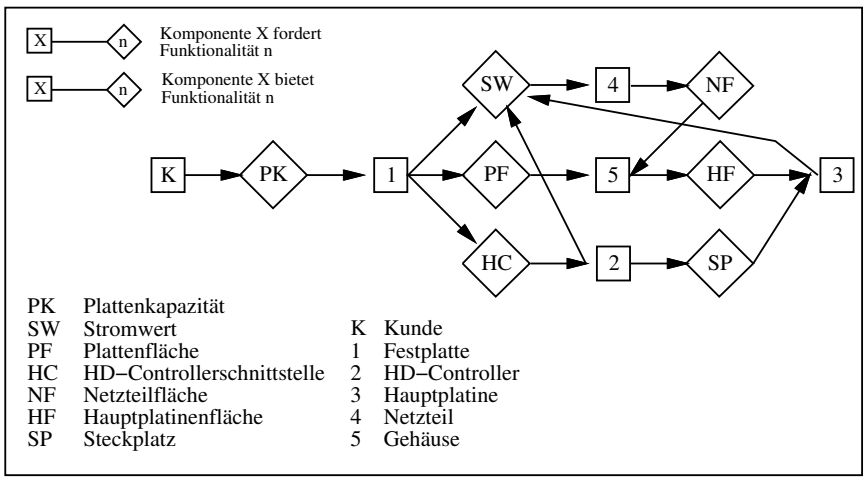

Abbildung 5: *Abhängigkeitsnetz*

Zum Beispiel bieten Netzteile u.a. die Funktionalität Stromwert (SW) an. Bevor aufgrund des geforderten Stromwertes ein oder mehrere Netzteile ausgewählt werden, sollten alle stromfordernden Komponenten bestimmt sein.

In MOKON wird aus den Komponentenbeschreibungen automatisch dieses Abhängigkeitsnetz aufgebaut. Innerhalb dieses Netzes werden die Funktionalitäten topologisch sortiert und ihnen eine dieser Sortierung entsprechende Priorität zugeordnet. Daher werden in unserem Beispiel alle Funktionen, aufgrund derer stromfordernde Komponenten ausgewählt werden, eine höhere Priorität als die Funktion Stromwert besitzen. Werden zyklische Abhängigkeiten festgestellt (starke Zusammenhangskomponenten im Graphen), muß der Benutzer festlegen, mit welcher Funktionalität innerhalb des Zyklus fortgefahren werden soll. Mit dem Generieren von Prioritäten wird ein Teil des bereichsspezifischen Kontrollwissens automatisch durch die Problemlösungsmethode festgelegt. Dadurch wird die Wissensakquisition unterstützt und MOKON im Sinne von McDermott [4] zu einer starken Problemlösungsmethode erweitert. Wesentliches Merk-

mal einer starken im Vergleich zu einer schwachen Problemlösungsmethode ist, daß eine starke Methode nur hochstrukturiertes oder gar kein bereichsspezifisches Kontrollwissen benötigt.

2.3 Mehrschichten-Modell

Neben der im vorherigen Abschnitt dargestellten funktionalen Beschreibungsebene existiert eine assoziative Ebene, die Abarbeitungsvorschläge enthält, welche nicht kausal (durch das funktionale Modell) gestützt sind. Es handelt sich hierbei um assoziative Vorschlags- und Korrekturregeln.

Die Lösungsfindung mithilfe des funktionalen Abhängigkeitsnetzes basiert auf einem Generate-and-Test–Algorithmus mit einem exponentiellen Laufzeitverhalten. Dieses Laufzeitverhalten wird dadurch verbessert, daß nicht alle Komponenten aufgrund dieser kausalen Zusammenhänge bestimmt werden: Durch Vorschlagsregeln wird aufgrund einiger charakteristischer Initialforderungen ein bestimmter Grundausbau der Konfiguration ausgewählt. Falls jetzt noch Forderungen offen sind, greift das System auf die funktionale Beschreibungsebene zurück. Das heißt, der oben skizzierte Weg der Wissensverarbeitung, welcher auf einer Propagierung von Angebots- und Forderungsbeziehungen beruht, wird eingeschlagen.

Korrekturregeln dienen in erster Linie dazu, die heuristische Abarbeitung des Suchraumes zu unterstützen. Gerät die Suche in eine Sackgasse (für eine offene Forderung können keine Alternativen ermittelt werden), so muß im Suchraum zurückgesprungen werden. Dieser Rücksprung ist wissensbasiert organisiert (vgl. [6]): Bei einer Funktionalität kann ein Hinweis auf eine Funktion abgelegt sein, welche die gescheiterte Forderung stark beeinflußt und zu der im Suchraum zurückgesprungen wird. Zusätzlich wird nach einem solchen Rücksprung die Heuristik zur Auswahl einer Alternative in Hinsicht auf die gescheiterte Funktionalität geändert. Abbildung 6 zeigt typische Funktionsspezifikationen:

Name	Plattenkapazität	Lieferzeitpunkt	Stromwert
Typ	numerisch	datum	numerisch
Constraint	erfüllbar	unerfüllbar	erfüllbar
Priorität	70	100	10
Saldierungsoperator	>=	<=	>=
Rücksprung			
Funktion	–	–	Plattenkapazität
Heuristik	–	–	Min. Stromwert
Angebot			
Kommunikationstyp	intern	intern	intern
Defaultwert	–	–	–
Wertebereich	0–300	–	–
Verarbeitung	+	Maximum	+
Forderung			
Kommunikationstyp	ask	ask	intern
Defaultwert	20	–	–
Wertebereich	–	–	–
Verarbeitung	+	Minimum	+

Abbildung 6: *Funktionsspezifikation*

Kann zum Beispiel für eine bestimmte Stromwertforderung keine Alternative gefunden werden, so wird zur Funktion Plattenkapazität zurückgesprungen, weil hier der Stromwert am stärksten beeinflußt wird. Die Auswahl einer neuen Alternative für die geforderte Plattenkapazität orientiert sich dann nicht mehr am Preis, sondern am minimalen Stromverbrauch.

Neben dem wissensbasierten Rücksprung ist auch ein abhängigkeitsunterstütztes Rücksetzen (Dependency Directed Backtracking) denkbar: Aus dem Abhängigkeitsnetz kann abgeleitet werden, welche Komponenten eine Funktionalität beeinflussen. Gerät der Suchprozeß in eine Sackgasse, d.h. die Forderung einer Funktionalität kann nicht erfüllt werden, so kann die Abhängigkeitsinformation für ein gezieltes Rücksetzen verwendet werden. Dies ist vergleichbar mit der Dienstleistung, die ein TMS einem Problemlöser zur Verfügung stellt. Eine solche Verwertung des Wissens aus dem Abhängigkeitsnetz ist in MOKON nicht realisiert.

3 Stand und Abgrenzung

3.1 Anwendung

Eine funktionsorientierte Betrachtungsweise zielt auf Domänen, in denen die zu konfigurierenden Komponenten nicht nur über ein Strukturmodell, sondern über ihre Funktionalität in Beziehung gesetzt werden können. Beispiele für solche Domänen sind die Konfigurierung von Computern und Telekommunikationsanlagen.

Als Anwendungsbeispiel dient hier die Konfigurierung der Telefonanlage Intergral 33x der Firma Telenorma, Frankfurt. Hierbei handelt es sich um eine ISDN-Anlage, die bis zu mehreren Tausend Teilnehmern ausgebaut werden kann. Es hat sich gezeigt, daß der überwiegende Teil des Wissens über Funktionalitäten beschreibbar ist[2]: Leiterplatten stellen verschiedene Anschlüsse zur Verfügung; Leiterplattenplatten benötigen u.a. Steckplätze und Strom usw.

Weil es sich hier nicht um tief strukturierte Produkte handelt, spielen räumliche Informationen keine dominierende Rolle – sie müssen jedoch auch modelliert werden:
Die Berücksichtigung räumlicher Information bei der *Auswahl* von Komponenten geschieht durch die Einführung von Pseudofunktionalitäten (Netzteilplatz, Netzteilbeschränkung usw.) mit entsprechenden Testprädikaten. Die *Anordnung* der Komponenten läßt sich jedoch nur sehr umständlich über Funktionalitäten modellieren und wird mit einem externen Modul durchgeführt.

Zur Zeit ist MOKON in Lucid Common LISP und KEE auf einer SUN Sparc1+ implementiert. Dabei wurde das KEE Object System zur Wissenrepräsentation sowie KEE Pictures zur Realisierung der Benutzerschnittstelle eingesetzt.

[2]Eine ausführliche Beschreibung der Wissenrepräsentation erfolgt in [9].

3.2 Einordnung

Mittlerweile existiert eine große Anzahl von Systemen, die für unterschiedliche Aufgabenstellungen aus dem Bereich der Konfigurierung bzw. zu deren Unterstützung entwickelt wurden. Aus Sicht der Wissensverarbeitung stellen die ressourcenorientierten Ansätze COSMOS [2] und AKON [5], [10] artverwandte Systeme dar.

COSMOS verwendet taxonomische Hierarchien zur ressourcenorientierten Beschreibung von Komponenten. Die Abarbeitung benutzt kausale Ursache-Wirkungsbeziehungen und basiert somit auf einem tieferen Verständnis der Anwendungsdomäne.
Kern der Wissensrepräsentation von AKON ist eine kompositionelle Hierarchie, in der die kleinste Einheit das Basiskonzept ist. Innerhalb eines Basiskonzeptes dienen explizite Relationen und Regeln zur Beschreibung von Konfigurationsvorgaben; basiskonzeptübergreifende Beziehungen werden über kausale Abhängigkeiten auf einer Bilanz verwaltet.

Vergleichbar den Systemen COSMOS und AKON basiert die Wissensverarbeitung von MOKON auf dem ressourcenorientierten Ansatz. ZusŁtzlich wird die funktionale Beschreibungsebene um assoziatives Wissen ergänzt sowie Suchrauminformationen aus dem funktionalen Abhängigkeitsnetz abgeleitet.

PLAKON ist wohl das am weitesten entwickelte Konfigurierungssystem und ist für verschiedene Aufgabenstellungen in unterschiedlichen Anwendungsdomänen einsetzbar. PLAKON integriert u.a. verschiedene Konfigurierungsstrategien, eine taxonomisch-kompositionelle Hierarchie, Parameterberechnung mit Defaultwerten, Constraintpropagierung und verschiedene Modi zur Lösungsraumverwaltung.
Durch die hybride Wissensrepräsentation und die komplexen Verarbeitungsmechanismen bedingt, ist diese Entwicklungsumgebung nur durch einen KI-Spezialisten bedienbar. Abgrenzend hierzu versuchen die ressourcenorientierten Ansätze nicht die Konfigurierungsproblematik in ihrer Allgemeinheit zu bearbeiten; vielmehr entsprechen sie der Philosophie der starken Problemlösungsmethoden [4].

Literatur

[1] R. Cunis, A. Günter, I. Syska: *PLAKON - Ein übergreifendes Konzept zur Wissensrepäsentation und Problemlösung bei Planungs- und Konfigurierungsaufgaben,* Expertensysteme ‘87, Konzepte und Werkzeuge, H. Balzert (Hrsg.), Teubner Verlag, Stuttgart 1987

[2] M. Heinrich: *Ressourcenorientierte Modellierung als Basis modularer technischer Systeme,* in: BeitrŁge zum 5.Workshop <<Planen und Konfigurieren<<, LKI-M-1/91

[3] J. McDermott: *R1: A Rule-Based Configurer of Computer Systems,* Artificial Intelligence 19 (1982) p.39-88

[4] J. McDermott: *Preliminary Steps Toward a Taxonomy of Problem-Solving Methods*. In S. Marcus: Automating Knowledge Acquisition for Expert Systems, Kluwer Academic Publishers,1988.

[5] B. Neumann: *Ein Ansatz zur wissensbasierten Auftragsprüfung für technische Anlagen des Breitengeschäfts,* Dissertation Universität-GH-Duisburg, 1990

[6] F. Puppe: *Problemlösungsmethoden in Expertensystemen,* Springer Verlag, 1990

[7] L. Steels: *Components of Expertise,* AI Magazine, 1990

[8] B. Stein, J. Weiner: *MOKON,* Interner Report, Universität-Gesamthochschule-Duisburg, SM-DU-178, 1990

[9] S. Schmitgen, B. Stein, J. Weiner: *Anlagenkonfigurierung – Analyse des Erweiterungsgeschäfts,* Interne Studie, TELENORMA, Frankfurt, 1991

[10] J. Weiner: *Aspekte der Konfigurierung technischer Anlagen,* Dissertation Universität-GH-Duisburg, 1991