EXPERTENSYSTEME - GRUNDLAGEN UND ENTWICKLUNGSWERKZEUGE

A. Schuh, Ludwigshafen
BASF AG Abteilung ZXT - Informatik-Technologie

Zusammenfassung

Es wird eine Einführung in die grundlegenden Ideen der Expertensystem-Technologie gegeben: was Expertensysteme (ES) sind und was sie leisten können. Die elementaren Bestandteile von ES werden beschrieben, Standardbegriffe der ES-Terminologie werden erläutert. Grundsätzliche Überlegungen zur Auswahl von ES-geeigneten Projekten und zum Vorgehen bei der Durchführung werden angestellt. Werkzeuge für die Entwicklung von ES auf PC's durch den Anwender werden klassifiziert.

Übersicht

1. Grundlagen der Expertensystem-Technik
   1.1 Definition und Zielsetzung von Expertensystemen
   1.2 Funktionsweise von Expertensystemen
2. Entwicklungswerkzeuge für Expertensysteme auf PC
   2.1 Grundsätzliches zu Expertensystem-Projekten und -Werkzeugen
   2.2 Klassifizierung von PC-Shells
3. Literaturangaben

1 Grundlagen der Expertensystem-Technik

1.1 Definition und Zielsetzung von Expertensystemen

Um Computerprogramme namens "Expertensysteme" definieren zu können, wollen wir zunächst überlegen, was ein menschlicher Experte ist und was er leistet:

"Ein Experte ist ein Mensch, der auf Grund seines Wissens und seiner Erfahrung auf einem Fachgebiet in der Lage ist, dort ihm gegebene, meist nichttriviale Probleme zu lösen. Er kann anderen diese Lösungen erklären und während seiner Tätigkeit dazulernen. Er ist in der Lage, seinen Kompetenzbereich einzuschätzen und diesen in Relation zum Allgemeinwissen und Nachbargebieten zu sehen."

Wenn wir diese Definition eines Experten einfach für ES übernehmen würden, würden wir von vorneherein an diesem zu hohen Anspruch scheitern, denn ES können nicht gleichwertig neben menschlichen Experten gesehen werden; dies wird gleich deutlich werden. Wir setzen daher bescheidener und realistischer an und sagen:

"Expertensysteme sind eine neuartige Programmiertechnik, um das Wissen von Experten flexibler und effizienter in DV-fähiger Form zu erfassen als mit konventioneller Programmierung. Entwicklungsziel ist es, Probleme auf einem plausiblen und nachvollziehbaren Weg mit einer einem menschlichen Experten vergleichbaren Leistung lösen zu können."

Um letzteren Anspruch erfüllen zu können, benötigen ES Elemente, über die auch der menschliche Experte verfügt:

- Wissen über das betrachtete Gebiet
- anwenderfreundlicher Dialog
- Einsatz natürlicher Sprache in Dialog und Wissensdarstellung
- Erklärungen
- Wissenserwerbskomponente ("Lernen")

Es sei nochmals auf die Limitierung des Anspruchs, was ES können und leisten sollen, hingewiesen. Sie sind nicht dafür gedacht, den menschlichen Experten zu ersetzen und an seiner Stelle Probleme aus der vordersten Front zu lösen, sondern dafür, dem Experten und dem Anwender zu helfen:

- Entlastung des Experten von Routinearbeiten/anfragen
- Unterstützung des Experten bei der eigenen Arbeit durch Systematisierung, Katalogisierung und Präsenthaltung seines Wissens
- Verfügbarmachung des Wissens für Anwender und Kollegen.

Ist es bei dieser Reduzierung des Anspruchs überhaupt noch sinnvoll, eine derartige neue Technologie einzusetzen ? Sicherlich, denn ES bieten viele positive neue Eigenschaften:

- Vollständige Präsenz des Wissens: was einmal eingegeben wurde, bleibt in einem ES für immer verfügbar. Der Experte vergißt menschlicherweise nach und nach, was er vor langer Zeit gelernt und getan hat. ABER: Es klingt trivial, muß jedoch wegen der oft falschen Vorstellung über ES hier betont werden: ES können nur mit dem Wissen arbeiten, das ihnen vorher eingegeben wurde bzw. was sie während einer Konsultation dazulernen, "von selbst" wissen ES nichts.
- Fehlersicherheit: Ist das eingegebene Wissen richtig, so werden auch die daraus möglichen Schlüsse immer wieder fehlerfrei gezogen werden können.
- Schnelligkeit: im Rahmen des im ES vorhandenen Wissens lösbare Probleme werden aufgrund der Präsenz immer schneller gelöst als "von Hand".
- Ökonomie der Wissenspflege: Der Aspekt der dauerhaften Präsenz des Wissens hat für jedes Unternehmen eine immense wirtschaftliche Bedeutung, denn es wird erstmals möglich, die Erfahrungen bewährter Mitarbeiter in einer effizienten Weise zu konservieren und an neue Kollegen weiterzugeben; das Wissen des Unternehmens bleibt auch nach Weggang des Mitarbeiters erhalten.

"Wissen" zu verarbeiten, geht wesentlich über die Perspektive bisheriger "Daten"-Verarbeitung hinaus, die sich im wesentlichen auf unmittelbar in Zahlen faßbare Informationen beschränkt; "Wissen" zu verarbeiten bedeutet eben auch, Erfahrungen aus der Arbeitspraxis darstellen zu können, die sich nicht in Zahlenwerten ausdrücken lassen. Bei der Besprechung der Funktionsweise von ES werden wir sehen, wie dies ermöglicht wird.

Es wird oft der Einwand gegen die Anwendung von ES erhoben, daß ein Problem auch mit konventioneller Programmierung gelöst werden könne. Prinzipiell trifft das sogar immer zu, aber:

- Für die von uns ins Auge gefaßten Problemklassen ist die herkömmliche Technik um ein vielfaches aufwendiger, oft praktisch unrealisierbar;
- Erweiterungen ("Dazulernen") sind nur durch Weiterprogrammieren möglich, erfordern also entsprechende Programmkenntnisse;
- Konventionelle Programme erklären nicht, was sie gerade warum tun und wohin es führt (Oder können Sie ein Fortran-Programm an einer beliebigen Stelle fragen, warum es gerade eine bestimmte Zahl braucht ?)

Wir behaupten nicht, daß Teile der Problemlösefähigkeit eines menschlichen Experten nicht auch mit konventionellen Programmen nachgebildet werden könnten. Der Aufwand dafür ist aber meist prohibitiv hoch, wohingegen diese Eigenschaften in der ES-Technik, da von vorneherein vorgesehen, leichter zu realisieren sind.

1.2 Funktionsweise von Expertensystemen

Wie sieht nun der Aufbau eines Programms aus, das die oben angeführten Eigenschaften besitzen soll, und wie läuft die Arbeit mit ihm ab ? Es ist klar, daß an dieser Stelle keine tiefgehende Einführung in die Techniken der ES-Programmierung gegeben werden kann, sondern nur eine Übersicht, die die wesentlichen Elemente verständlich machen soll.

Um die Anforderungen der in 1.1 gegebenen Definition zu erfüllen, enthält ein ES folgende wesentliche Teile (Abb. 1):

1. Wissensbasis
2. Inferenzmaschine
3. Dialogkomponente
4. Erklärungskomponente
5. Wissenserwerbskomponente

1. Beginnen wir mit der Betrachtung der Wissensbasis. Die Wissensbasis enthält die Informationen über das zu lösende Problem, nach Möglichkeit in einer Form dargestellt, die nahe an der natürlichen Sprache des Menschen gehalten ist. Es sollen hier drei Typen von Wissensdarstellung ("Wissensrepräsentation") erwähnt werden:
- Regeln
- Rahmen
- Semantische Netze

Der Begriff der "Regeln" ist einfach zu erklären, da er praktisch identisch ist mit der allgemeingültigen Definition dieses Begriffes. Regeln drücken aus, unter welchen Voraussetzungen bestimmte Schlußfolgerungen möglich sind. Beispiel: In einem ES zur Pflanzenbestimmung, mit dessen Hilfe auch ein Laie in der Botanik die Einordnung (Familie) einer ihm vorliegenden Pflanze feststellen kann, würde eine (sachlich unvollständige) Regel etwa lauten:

WENN Der Stamm der Pflanze IST holzig
UND Die Position der Pflanze IST kriechend
DANN Die Familie der Pflanze IST Araliaceae

Durch Großschreibung sind Standardelemente der Regelsprache gekennzeichnet, die die logische Struktur der Regel und die erlaubten Verknüpfungen aufbauen.

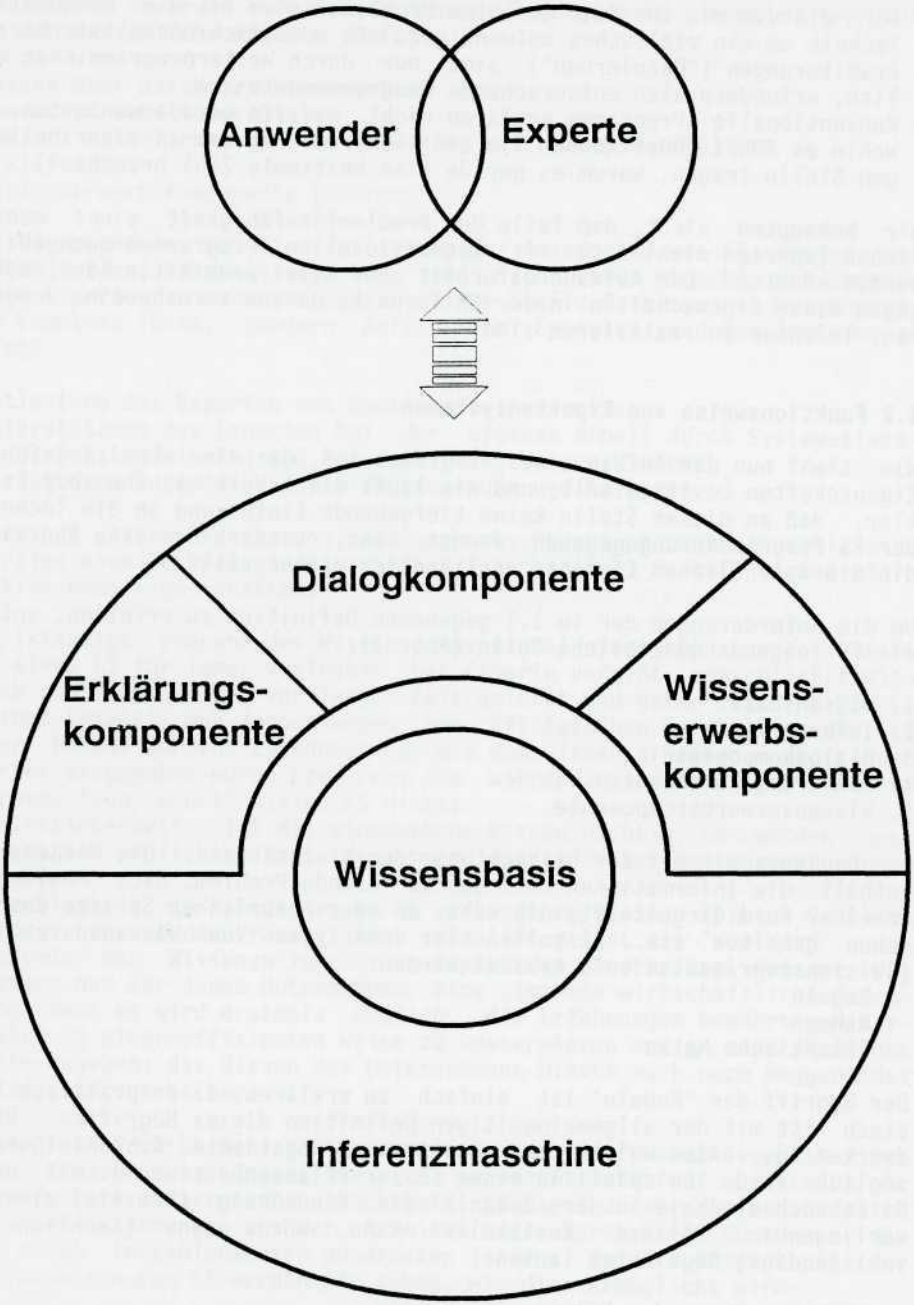

Abb.1: Grundelemente eines Expertensystems

"Rahmen" ("Frames", "Klassen", "Schemata") sind ein Mittel zur Zusammenfassung des Wissens über ein abstraktes Objekt. Dem Objekt wird eine Liste von Eigenschaften ("Attribute") in der Form von "Slots" zum Einfüllen von Werten beigefügt. Reale Objekte sind dann "Instanzen" dieser Rahmen, den Eigenschaften werden konkrete Werte zugeordnet. Man könnte z.B. einen Rahmen "Pflanzen" definieren mit den Slots

Pflanze:
- Familie:
- Gattung:
- Art:
- Sproß:
- Blätter:

Eine Instanz "Himbeere" von "Pflanze" kann dann so aussehen:

Himbeere:
- ist-Instanz-von: Pflanze
- Familie: Rosaceae
- Gattung: Rubus
- Art: Rubus idaeus
- Sproß: grün
- Blätter: gefiedert

Wiederum sieht man, daß hier Wissen in einer der natürlichen Sprache nahestehenden Form erfaßt werden kann.

Schließlich seien noch "Netze" (genauer: "Semantische Netze") erwähnt. Hier können Objekte und Werte als Knoten eines Beziehungsnetzes dargestellt werden. Die Knoten werden graphisch durch Bögen miteinander verbunden, die die zwischen jenen bestehenden Relationen anzeigen. In einem Netz kann eine Hierarchie von Rahmen aufgebaut werden, in der Eigenschaften vererbt werden können (von der maximalen Abstraktion über intermediäre Klassen bis hin zu den Instanzen = realen Objekten). Ein Beispiel wird in Abb. 2 gezeigt. Ein Netz wird in einer Wissensbasis durch spezielle Slots realisiert, die für ihre Objekte die Relationen und die Vererbung angeben, z.B. aus Abb. 2:

Baum:
- ist-Subklasse-von: Pflanze
- hat-Subklasse: Nadelbaum, Laubbaum
- ...(Weitere Slots)

"ist-Subklasse-von" ist hierbei die inverse Relation zu "hat-Subklasse".

Wir haben damit einige der grundlegenden Möglichkeiten gezeigt, wie Wissen explizit in einer Wissensbasis dargestellt werden kann (Werden mehrere solche Mechanismen in einer miteinander integrierbaren Form eingesetzt, spricht man übrigens von "hybrider" Wissensrepräsentation). Wie ein ES dieses Wissen verarbeitet, wollen wir bei der Betrachtung der Inferenzmaschine kennenlernen.

2. Die Inferenzmaschine ("Inferenz" = Schlußfolgerung) ist das zentrale Element außerhalb der Wissensbasis des ES, sie steuert den Ablauf der Verarbeitung und die Auswertung des Wissens. Beschränken wir uns hier

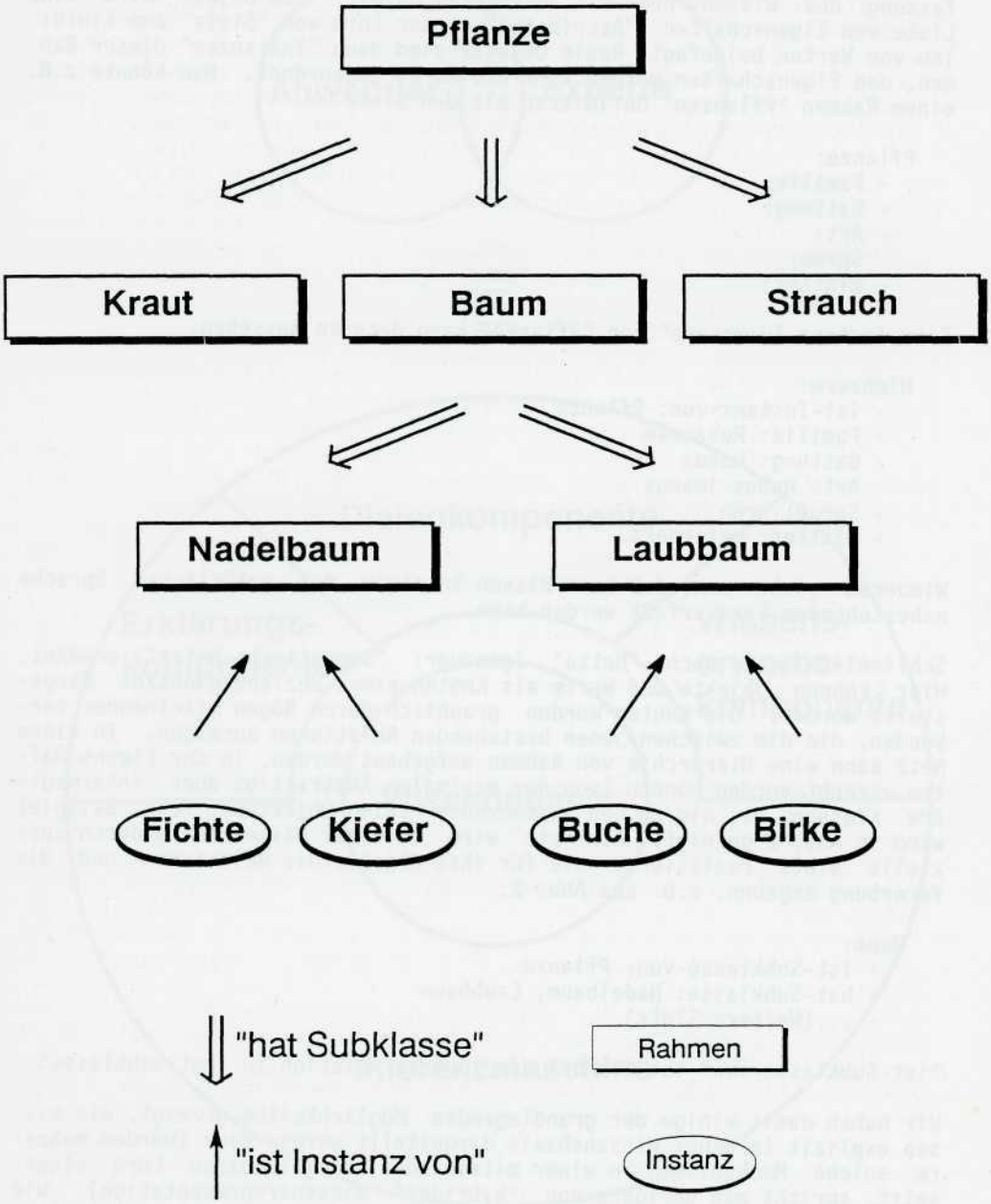

Abb. 2: Ein einfaches semantisches Netz

einmal auf die Auswertung von Regeln. Hier gibt es zwei grundlegende "Inferenzstrategien": "vorwärtsverkettend" (datengetrieben) und "rückwärtsverkettend" (zielgetrieben).

Bei vorwärtsverkettender Regelauswertung wird geprüft, welche Regel bei den derzeit vorhandenen Daten gültig ist; deren Schlußfolgerungen können also gezogen und als neue gültige Fakten gespeichert werden. Für die meisten ES-Anwendungen ist die rückwärtsverkettende Vorgehensweise angebracht: Für ein gegebenes Ziel (zu findender Wert) wird eine Regel gesucht, die etwas über dieses Ziel aussagt. Deren Prämissen werden geprüft. Falls notwendig, werden Subziele aufgestellt, zu deren Prüfung weitere Regeln analysiert werden müssen, solange bis eine Aussage über das Ziel gemacht werden kann (oder die Wissensbasis ausgeschöpft ist). Besser versteht man diesen Mechanismus anhand eines Beispiels: Es seien drei Regeln in einer einfachen Wissensbasis vorhanden:

Regel 1: WENN Blattform IST dreizählig-gefiedert (P1)
UND Frucht IST steinlos (P2)
DANN Gattung IST Fragaria

Regel 2: WENN Blattform IST dreizählig-gefiedert (P1)
UND Frucht IST Steinfrucht (P2)
DANN Gattung IST Rubus

Regel 3: WENN Gattung IST Rubus (P1)
UND Fruchtstand IST einzeln (P2)
DANN Art IST Himbeere (Rubus idaeus)

Ziel der Befragung des ES mit dieser Wissensbasis sei es, festzustellen, welche Art von Pflanze dem Betrachter vorliegt. Die Inferenzmaschine sucht die Wissensbasis nach einer Regel durch, die etwas über "Art" aussagt; sie findet Regel 3. Jetzt müssen die Prämissen "Gattung" und "Fruchtstand" geprüft werden. Regel 1 sagt etwas über "Gattung". Die Prämissen von Regel 1 werden aber nicht durch weitere Regeln behandelt; in dieser Situation kann das ES den Benutzer fragen:

"Wie ist die Blattform der Pflanze ?"

Falls der Benutzer mit "dreizählig-gefiedert" antwortet, ist P1 von Regel 1 wahr und P2 muß geprüft werden. Das Es muß wiederum fragen, und zwar:

"Wie ist die Frucht der Pflanze ?"

Da wir eine Himbeere vor Augen haben, sagen wir "Steinfrucht". Damit ist P2 falsch und wegen der UND-Verknüpfung auch die ganze Regel 1. Das ES muß jetzt nach einer anderen Regel suchen, die etwas über "Gattung" aussagt; dies tut Regel 2. P1 von Regel 2 wurde bereits als wahr erkannt und gespeichert; P2 von Regel 2 wurde ebenfalls gerade vom Benutzer gegeben, so daß Regel 2 wahr ist. Damit kann die "Gattung" als "Rubus" festgestellt werden. Damit ist aber P1 von Regel 3, zu der die Inferenzmaschine jetzt zurückkehrt, wahr. Jetzt muß noch P2 von Regel 3 geprüft werden. Wird die erwartete Antwort "einzeln" gegeben, so kann die Art als "Himbeere" (Rubus idaeus) festgestellt werden, und das gegebene Ziel, die "Art" zu bestimmen, wurde erreicht. Wird eine andere Antwort gegeben, und es gibt keine weiteren Regeln über "Art", so muß das ES antworten "Art nicht bestimmbar".

Wir haben einen Eindruck erhalten, wie ein ES bei der Verarbeitung von Wissen vorgehen kann, und wollen nun noch kurz die restlichen wichtigen Komponenten eines ES besprechen.

3. Die Dialogkomponente hält die Verbindung zwischen ES und Außenwelt, also dem Entwickler und dem Anwender, aufrecht, und wird ebenfalls von der Inferenzmaschine gesteuert. Als wichtiges separates und qualifizierendes Modul eines ES wird die DK erwähnt, weil die Experten-ähnlichen Eigenschaften eines ES nur durch hochstehende Kommunikationsformen mit dem Menschen erreicht werden können. Dazu gehören z. B.

- Fenstertechnik
- Menüs mit vordefinierten Antwortmöglichkeiten
- Vermeiden von überflüssigem Eintippen
- Einsatz einer Maus
- Hilfen zur aktuellen Situation

Fortschrittliche Systeme können z.B. mit Hilfe eines "Parsers" (Zerlegers) die eingegebenen Werte semantisch (also auf ihren Sinn hin) prüfen, und dem Bediener anbieten, ob er etwas korrigieren will oder nicht.

Da die Akzeptanz eines ES wesentlich von der Benutzeroberfläche abhängt, mit der das System gegenüber dem Entwickler und Anwender erscheint, ist eine gute Dialogkomponente ein wichtiges Element eines brauchbaren ES und auf deren Realisierung ist entsprechende Sorgfalt zu verwenden. Die Möglichkeiten, die ein ES-Entwicklungswerkzeug in dieser Hinsicht bietet, sind daher auch ein Kriterium für die Auswahl des Werkzeuges; dazu mehr im zweiten Abschnitt.

4. Die Erklärungskomponente ist ein Modul, welches zur Charakterisierung von ES in besonderem Maße beiträgt. So wie auch ein menschlicher Experte erklären kann (können sollte!), wie er zu der Lösung eines Problems gelangt, und mit logischen Argumenten die Korrektheit der Lösung rechtfertigen kann, soll auch ein ES dies können. Der Benutzer soll z.B. fragen können:

- Warum ? und das ES gibt aus, warum eine Frage gestellt, eine Regel verwendet, eine Schlußfolgerung gezogen wird. Oder:

- Was ? und das ES sagt, was bisher bereits abgeleitet wurde; oder:

- Wie ? und das ES zeigt, wie ein Faktum abgeleitet wurde oder werden könnte. Sehr interessant ist die Möglichkeit,

- Was wäre wenn ? zu fragen, und das ES gibt aus, welche Schlußfolgerungen sich aus einer Benutzereingabe ergeben können und welche Richtung die weitere Konsultation damit nimmt.

Mit diesen Kommunikationsmöglichkeiten wird die Konsultation eines ES zu einem lebendigen, durchsichtigen Vorgang; der Anwender erhält die Chance, das Ergebnis zu verstehen und zu lernen, wie es abgeleitet wurde. Dies ist ein enormer Fortschritt gegenüber dem Ablaufenlassen konventioneller Programme, und damit ein bedeutender Grund, die neue Technik der ES einzusetzen.

5. Die Wissenserwerbskomponente ist ein weiteres besonderes Modul, welches im ES-Benutzer-Dialog sichtbar wird. Sie unterstützt den Benutzer bei der Eingabe neuen Wissens, sogar interaktiv während einer Konsultation. Ziel ist es, ohne Systemkenntnisse das Wissen in möglichst natürlicher Form eingeben zu können, die Umsetzung in die systemkonforme verarbeitbare Darstellung erfolgt automatisch. Die Wissenserwerbskomponente muß deshalb auch prüfen können, ob sich das neue Wissen mit der bisherigen Wissensbasis verträgt, denn ohne diese Konsistenzprüfung müßte man vom eingebenden Anwender Detailkenntnisse der Wissenbasis erwarten. Ringschlüsse, Widersprüche, Redundanzen oder Zweige des logischen Baumes mit offenen Enden müssen hier aufgespürt werden können. In der WEK muß der Verarbeitungsmechanismus innerhalb der Inferenzmaschine berücksichtigt werden, aber auch die formale und inhaltliche Struktur der Wissensbasis. Wir sehen, daß eine WEK eine sehr anspruchsvolle Programmieraufgabe darstellt, und es soll nicht verschwiegen werden, daß heutige ES Wissenserwerbskomponenten nur in Ansätzen - wenn überhaupt - enthalten. Die Automatisierung des Lernens und die Überprüfung der Inhalte stellen ein wirklich komplexes Problem dar. In diesem Gebiet muß in den nächsten Jahren noch wesentliche Entwicklungsarbeit geleistet werden, und es ist mit entsprechenden Fortschritten zu rechnen.

Anhand eines kleinen ES zur Pflanzenbestimmung zeigen wir das Zusammenwirken der ES-Komponenten und den Ablauf einer Konsultation in anschaulicher Weise. Damit kommen wir zum zweiten Thema dieses Vortrages: Wie hat man bei der Entwicklung eines ES vorzugehen, und welche Werkzeuge setzt man dafür ein ?

2 Entwicklungswerkzeuge für Expertensysteme auf PC

2.1 Grundsätzliches zu Expertensystem-Projekten und -Werkzeugen

Die Frage, welche Entwicklungswerkzeuge beim Aufbau eines ES einzusetzen sind, ist nur ein Teil der notwendigen Diskussion; diese umfaßt:

- Auswahl des geeigneten Projektes
- Vorgehensweise für die Abwicklung

Sofort wird sichtbar, daß die Werkzeugauswahl ein Teil des zweiten Schrittes ist. Doch wir wollen zunächst auch die Aspekte des ersten Schrittes anreißen: Wann ist ein ES überhaupt sinnvoll ?

Für ein ES eignen sich Aufgaben, die folgende Kriterien erfüllen:

- Sie sind weder trivial noch unlösbar schwierig; typischerweise würde ein menschlicher Experte dafür eine bis mehrere Stunden benötigen;
- Die Struktur des Problems ist eher von Experten- und Erfahrungswissen geprägt als von Massendaten in Zahlen;
- Es ist wichtiger, Symbole manipulieren zu können als mit Zahlen zu rechnen;
- Es kann mehrere Lösungen unterschiedlicher Qualität geben;
- Das Programm soll mit unvollständigen oder fehlerbehafteten Eingaben oder unsicherem Wissen umgehen können;
- Konventionelle DV-Methoden haben bei dem betrachteten Problem bereits versagt;

- Ein ökonomischer Nutzen des fertigen Systems ist absehbar;

und schließlich nicht zu vergessen:

- Experten für die Erstellung der Wissensbasis sind vorhanden und zur Mitarbeit bereit!

Die Kategorien der Aufgaben, in welchen ES erfolgreich eingesetzt werden können, werden oft so zusammengefaßt:

- DIAGNOSE (Klassifikation): Aus gegebenen Befunden oder Meßwerten soll ein bekanntes Muster wiedererkannt werden, z.B. MYCIN zur Diagnose und Therapie bakterieller Erkrankungen;
- DESIGN (Konfiguration): Anhand eines Kataloges von Anforderungen werden Gegenstände zusammengestellt, die den Anforderungen genügen, z.B. XCON von DEC zur Konfigurierung von VAX-Rechnern;
- PLANUNG: Für vorgegebene Anfangs- und Zielzustände soll eine Aktionssequenz ausgegeben werden, die den Anfangs- in den Zielzustand überführt, z.B. MOLGEN für die Durchführung molekularbiologischer Experimente;
- SIMULATION: Aus einem Ausgangszustand und erlaubten Aktionen oder Ereignissen sollen mögliche Folgezustände berechnet werden; z.B. SWIRL zur Simulation von Luftgefechten;
- TUTORIAL (Hilfesystem): Unterstützung des Benutzers bei EDV-Aufgaben, z.B. YES/MVS für das Operating von IBM-Großrechnern.

Die meisten bisher bekanntgewordenen ES wurden in den Bereichen Diagnose und Design geschaffen; das Gebiet Tutorial gewinnt in letzter Zeit an Bedeutung. Die Felder Planung und Simulation sind bisher wenig bearbeitet worden; teilweise liegt dies an der Konkurrenz mächtiger anderer Werkzeuge.

Für die Abwicklung von ES-Projekten hat sich folgende dreistufige Vorgehensweise als brauchbar herausgestellt (Ein strukturiertes Vorgehen ist natürlich für jede Art von Projekt empfehlenswert!):

1. Machbarkeitsstudie:

- Es wird untersucht, ob ein ES für das Problem sinnvoll ist;
- Die geeignete Methode der Wissensrepräsentation wird gesucht;
- Das Entwicklungswerkzeug wird gewählt;
- Eine "Demoversion" wird erstellt, die ein Beispiel aus dem Wissensgebiet vollständig abbildet.

Falls die Machbarkeit des ES gezeigt werden kann, folgt die

2. Prototypphase:

- Erstellung einer Prototypversion, die ein Teilgebiet des Problems vollständig behandelt und dort alle Wissensstrukturen des späteren Arbeitssystems enthält;
- Endgültige Festlegung des Entwicklungswerkzeuges;
- Bestimmung der Ablaufumgebung, in der das ES den Anwendern zur Verfügung gestellt werden soll.

Wird der Prototyp als erfolgreich beurteilt, kommt schließlich der 3. Endausbau zum Arbeitssystem:

- Erstellung der arbeitsfähigen Version;
- Portierung in die Ablaufumgebung;
- Pflege und Weiterentwicklung der Wissensbasis.

Ein ideales ES wird eigentlich nie "fertig", denn genauso wie der Experte während seiner Arbeit dazulernt, soll ein ES immer weiter verbessert und verfeinert und dem Fortgang des Wissens angepaßt werden.

Mit welchen Werkzeugen soll nun ein Programmierprojekt "Expertensystem" verwirklicht werden ? Das einzusetzende Werkzeug muß mit mehreren Seiten des Projektes in Übereinstimmung sein:

- Natur und Eigenschaften des Problems;
- Fähigkeiten des Entwicklers;
- Verfügbare Entwicklungshardware;
- Eigenschaften der vorzusehenden Ablaufumgebung.

Über die Natur von ES-Aufgaben haben wir bereits gesprochen; beispielsweise gibt es Tools, die in besonderem Maße für Diagnoseprobleme geeignet sind, etc. Sowohl bei der Entwicklungshardware als auch der Ablaufumgebung wollen wir uns heute auf den PC beschränken, als weit verbreitetes, leicht verfügbares und weitgehend standardisiertes Medium sowohl für Entwicklung als auch Konsultation des ES durch die vorgesehenen Benutzer. Letzteres ist ein nicht zu vernachlässigender Aspekt, da bei der Entwicklung von ES auf Spezialhardware oft die Problematik entsteht, in welcher geeigneten Umgebung man das fertige System dem Anwender überhaupt zur Verfügung stellen kann. Der Markt der verfügbaren Werkzeuge deckt zudem heute in erster Linie die Bedürfnisse des PC-Benutzers ab, wegen der Enge ihrer Spezialmärkte sind Betrachtungen über Werkzeuge für Spezialsysteme und Workstations nicht so ergiebig, wobei letztere allerdings an Bedeutung gewinnen und der Übergang vom PC zur Workstation sowieso immer fließender wird.

Nun zu den Voraussetzungen, die durch die Kenntnisse des Entwicklers gegeben werden. Wir sind heute an einem Punkt, wo ES und ihre Entwicklungswerkzeuge die Labors der DV-Spezialisten verlassen und in die Bereiche breitgestreuter Anwendungsfelder vordringen. Damit ist der Trend bei den Werkzeugen ganz klar: Weg von Spezialisten-Tools, die den Fachgebietsexperten als Entwickler hilflos alleine lassen, und hin zu gut durchdachten, kommerziell reifen Produkten, die auch in der Hand des DV-Endanwenders ohne Vorkenntnisse in KI oder speziellen Programmiersprachen wie LISP oder PROLOG zu brauchbaren Systemen führen. Zusammenfassend: Wir brauchen Werkzeuge, die für den Experten sinnvoll sind und nicht nur für den DV-Spezialisten.

Für den direkten Einsatz einer Sprache wie LISP spricht vieles, wie Leistungsumfang und Flexibilität, was besonders bei komplexen Aufgaben benötigt wird. Doch mit dem von uns ins Auge gefaßten Ziel, den Fachmann selbst entwickeln zu lassen, treten andere Kriterien in den Vordergrund:

- Leichte Erlernbarkeit und Bedienbarkeit des Werkzeuges
- Benutzerfreundliche Oberfläche und Bedienerführung
- Verfügbarkeit von Standardmodulen
- Softwaresupport des Anbieters
- Wissensdarstellung nahe an der natürlichen Sprache

- Einbindung von existierenden Programmen
- Verträglichkeit mit der existierenden DV-Umgebung
- Niedrige Entwicklungskosten

An diesen Anforderungen werden wir uns orientieren, wenn wir im nächsten Abschnitt einige Werkzeuge klassifizieren.

2.2 Klassifizierung von PC-Shells

Zunächst wollen wir den Begriff "Shell" klären. Wenn wir Abb. 1 nochmals betrachten, finden wir bereits einen Ansatzpunkt für den Umfang eines ES-Werkzeuges. Denn wenn wir die problemspezifische Wissensbasis herausnehmen, so verbleibt uns eine leere "Schale" aus weitgehend standardisierbaren Elementen; dies versteht man unter einer ES-"Shell". Gute Shells sollten außerdem noch versehen werden mit

- Editoren
- Graphikwerkzeugen
- Schnittstellen (zu externen Programmen, Datenbanken, Worksheets etc.)

Im vorherigen Abschnitt haben wir bereits einige wesentliche Kriterien für Shells genannt. Anhand der Anforderungen für Vorhaben unterschiedlicher Komplexität lassen sich die Shells gemäß den von ihnen gebotenen Eigenschaften z.B. auf drei Ebenen ansiedeln:

1. Ebene: Kleine Shells für einführende, eher pädagogische Zwecke, um die Wissensdarstellung und Programmiertechnik der ES-Technologie darstellen zu können. Günstigerweise wird hier mit Regeln in Rückwärtsverkettung gearbeitet; Menüs, Hilfen und ein gutes Manual sollten vorhanden sein.

2. Ebene: Mittlere Shells für Prototyp-Entwicklung und kleinere Projekte. Hier kann verlangt werden:

- Regeln rückwärts und vorwärts
- Erweiterte Regelsyntax
- Schnittstellen (Datenbanken, Worksheets, Betriebssystem, Programmiersprachen, externe Programme)
- Kontrollstrukturen
- Wahrscheinlichkeitsfaktoren

3. Ebene: Große Shells mit hybrider Wissensrepräsentation für große Projekte. Alle Elemente der zweiten Ebene sind vorhanden, dazu:

- Hybride Wissensrepräsentation mit Regeln und Rahmen
- Semantische Netze
- Graphisches Interface mit Maus
- Übergangsmöglichkeiten auf leistungsfähigere Hardware (z.B. VAX)

Der Leistungsumfang der großen Shells orientiert sich bereits an den Produkten für Workstations und LISP-Maschinen wie KEE, ART, Knowledge Craft. Es ist verständlich, daß mit zunehmender Komplexität des Werkzeuges der Aufwand für Erlernen und Handhaben zunimmt.

In Abb. 3 zeigen wir für 11 Shells die Einstufung in die drei genannten Leistungsklassen, außerdem geordnet nach dem ungefähren derzeitigen Preis in Tausend DM ohne MWSt. Die Übergänge zwischen diesen Klassen sind natürlich nicht scharf, sondern fließend. Ideal wäre ein einziges Werkzeug,

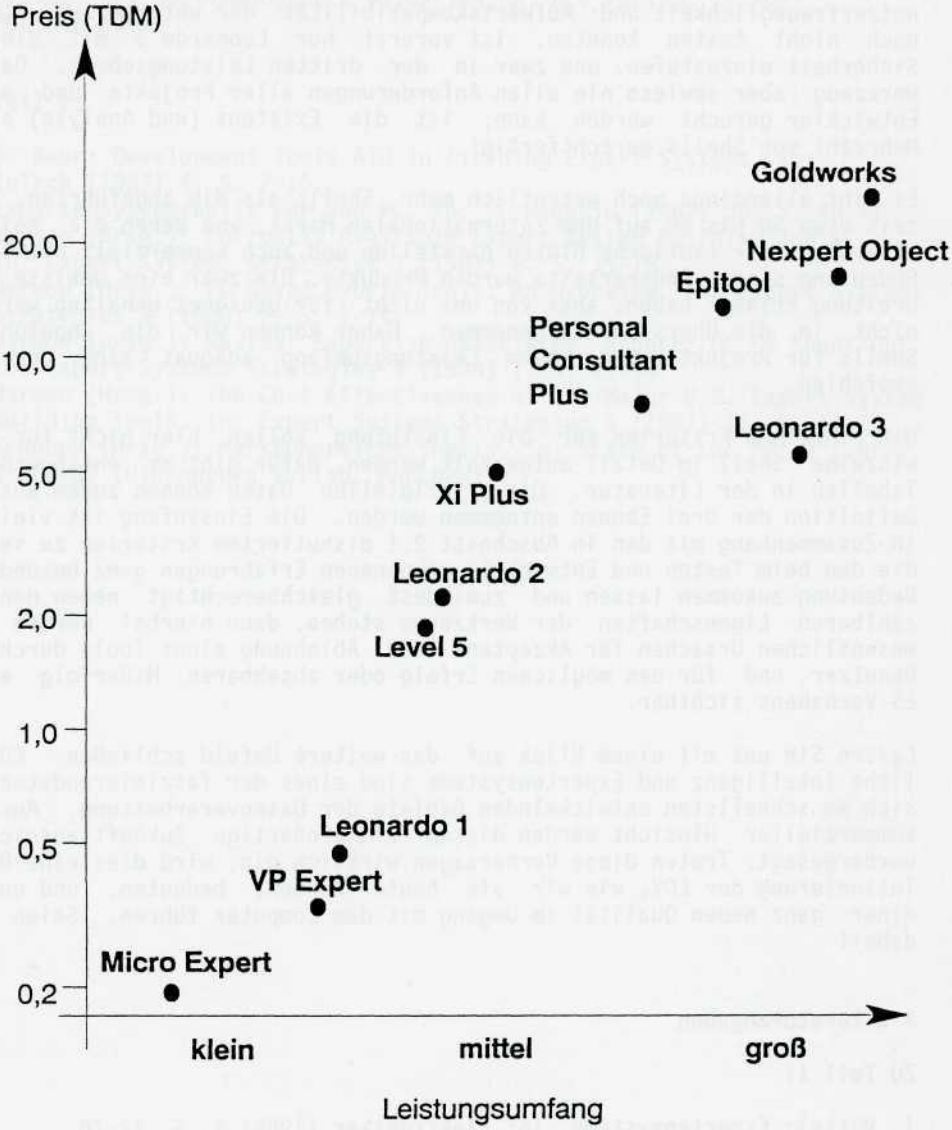

Abb. 3: Einordnung einiger ES-Entwicklungswerkzeuge (Shells) für PC

welches so modular aufgebaut ist, daß es auf allen drei Stufen eingesetzt werden kann. "Leonardo" ist in dieser Hinsicht konzipiert; da wir die Benutzerfreundlichkeit und Aufwärtskompatibilität der unteren zwei Stufen noch nicht testen konnten, ist vorerst nur Leonardo 3 mit einiger Sicherheit einzustufen, und zwar in der dritten Leistungsebene. Da ein Werkzeug aber sowieso nie allen Anforderungen aller Projekte und aller Entwickler gerecht werden kann, ist die Existenz (und Analyse) einer Mehrzahl von Shells gerechtfertigt.

Es gibt allerdings noch wesentlich mehr Shells als die angeführten, derzeit etwa 50 bis 60 auf dem internationalen Markt, von denen die meisten allerdings nur exotische Blüten darstellen und auch kommerziell nicht von Bedeutung sind. Andererseits wurden Produkte, die zwar eine gewisse Verbreitung erlangt haben, aber von uns nicht für geeignet gehalten werden, nicht in die Übersicht aufgenommen. Daher können wir die angeführten Shells für Projekte, die ihrem Leistungsumfang adäquat sind, durchaus empfehlen.

Die einzelnen Kriterien für die Einstufung sollen hier nicht für jede einzelne Shell im Detail aufgezählt werden, dafür gibt es entsprechende Tabellen in der Literatur. Die prinzipiellen Daten können zudem aus der Definition der drei Ebenen entnommen werden. Die Einstufung ist vielmehr in Zusammenhang mit den in Abschnitt 2.1 diskutierten Kriterien zu sehen, die den beim Testen und Entwickeln gewonnenen Erfahrungen ganz besondere Bedeutung zukommen lassen und zumindest gleichberechtigt neben den abzählbaren Eigenschaften der Werkzeuge stehen, denn hierbei werden die wesentlichen Ursachen für Akzeptanz oder Ablehnung eines Tools durch den Benutzer und für den möglichen Erfolg oder absehbaren Mißerfolg eines ES-Vorhabens sichtbar.

Lassen Sie uns mit einem Blick auf das weitere Umfeld schließen. Künstliche Intelligenz und Expertensysteme sind eines der faszinierendsten und sich am schnellsten entwickelnden Gebiete der Datenverarbeitung. Auch in kommerzieller Hinsicht werden diesem Feld großartige Zukunftsaussichten vorhergesagt. Treten diese Vorhersagen wirklich ein, wird dies eine Revolutionierung der EDV, wie wir sie heute kennen, bedeuten, und uns zu einer ganz neuen Qualität im Umgang mit dem Computer führen. Seien Sie dabei!

3 Literaturangaben

Zu Teil 1:

I. Büttel: Expertensysteme. In: Elektroniker (1986) 3, S. 72-78.

F. Haugg, S. Omlor: Expertensysteme auf PCs. Hanser Verlag, München 1987.

F. Puppe: Expertensysteme. In: Informatik-Spektrum (1986) 9, S. 1-13.

P. Schnupp, U. Leibrandt: Expertensysteme - Nicht nur für Informatiker. Springer Verlag, Berlin 1986.

P. Struß: Gibt es Expertensysteme ? In: Computer Magazin (1986) 5, S. 49-53.

B. A. Thomson, W. A. Thomson: Inside an Expert System. In: BYTE (1985) 4, S. 315-330.

P. Trum, N. R. Iudica: Expertensysteme - Aufbau und Anwendungsgebiete.

In: Chem. Ing. Tech. 58 (1986) 1, S. 6-9.
B. Ursing: BLV Bestimmungsbuch Wildpflanzen. BLV Verlag, 2. Aufl. München 1966.

Zu Teil 2:

P. S. Baur: Development Tools Aid in Fielding Expert Systems. In: InTech (1987) 4, S. 7-15.
Expertensystemshells für den PC. In: PC Magazin (1987) 30, S. 40-41.
P. Harmon (Hrsg.): Currently Available Expert System Building Tools. In: Expert Systems Strategies 3 (1987) 6, S. 11-18.
P. Harmon (Hrsg.): PC Based Hybrid Tools: An Overview. In: Expert Systems Strategies 4 (1988) 2, S. 1-14.
P. Harmon (Hrsg.): Review: Expertise Associates' Report: PC ES Shoot-Out. In: Expert Systems Strategies 4 (1988) 1, S. 13-16.
P. Harmon (Hrsg.): The Cost Effectiveness of the Major U.S. Expert System Building Tools. In: Expert Systems Strategies 3 (1987) 11, S. 14-16.
P. Schnupp (Hrsg.): Marktübersicht Shells. In: State of the Art 3 (1987), S. 80-87 (Oldenbourg Verlag, München).