Zeitschrift: Bulletin des Schweizerischen Elektrotechnischen Vereins, des Verbandes Schweizerischer Elektrizitätsunternehmen = Bulletin de l'Association suisse des électriciens, de l'Association des entreprises électriques suisses

Herausgeber: Schweizerischer Elektrotechnischer Verein ; Verband Schweizerischer Elektrizitätsunternehmen

Band: 80 (1989)

Heft: 3

Artikel: Integration von wissensbasierten Systemen in Informationssysteme

Autor: Probst, A. R. / Hametner, D.

DOI: https://doi.org/10.5169/seals-903631

Integration von wissensbasierten Systemen in Informationssysteme

A.R. Probst, D. Hametner

Der Informationssystemcharakter von wissensbasierten Systemen (oder Expertensystemen) wurde bis jetzt in der Praxis nicht genügend berücksichtigt. Die in jeder Organisation notwendige Einbettung in die operationellen Systeme stellt gewisse Anforderungen, die in diesem Artikel erörtert werden.

L'aspect informationnel des systèmes à base de connaissances (ou systèmes experts) n'a pas été jusqu'à maintenant suffisamment pris en considération. Cette caractéristique essentielle de ses applications, qui implique leur intégration dans les systèmes d'information opérationnels de toute organisation, entraîne un certain nombre d'exigences qui seront discutées dans cet article.

Adresse der Autoren
Prof. Dr. *André R. Probst* und Dr. *Dieter Hametner*, IBM Schweiz, Postfach, 8022 Zürich

Von der Daten- zur Wissensverarbeitung

Die Informationstechnologie befindet sich in einer Phase des Übergangs von der reinen Datenverarbeitung zur Wissensverarbeitung. Schon Ende der 60er, Anfang der 70er Jahre wurden *intelligente* Anwendungsprogramme unter dem Namen *Expertensysteme* konzipiert und entsprechende Prototypen entwickelt. Das Ziel dieser Programme war und ist, die *heuristischen* Problemlösungsmethoden von Spezialisten nachzuahmen und die Leistungen menschlicher Experten bei klar begrenzten Aufgaben zu erreichen. Expertensysteme enthalten Wissen über einen speziellen Anwendungsbereich und werten dieses mit Hilfe von Schlussfolgerungsmechanismen aus. Sie sind mit einer Schnittstelle zur Aussenwelt versehen, die eine Dialogführung oder eine Prozessüberwachung und -lenkung erlaubt. Eine Erklärungs- und Begründungskomponente macht die Arbeitsweise des Systems nachvollziehbar. Die Fähigkeit, jederzeit die Ableitungsschritte des Programmes dem Benützer mitteilen zu können, geht auf den folgenden Ansatz zur Implementierung des Anwendungsprogrammes zurück: das Wissen ist *explizit* codiert (nicht implizit in einem Algorithmus verborgen) und in einer *Wissensbasis* gespeichert; die Wissensbasis ist getrennt von dem Programm, das den Gesamtablauf steuert (Mechanismus des Schlussfolgerns, Führung des Dialogs mit dem Benützer usw.).

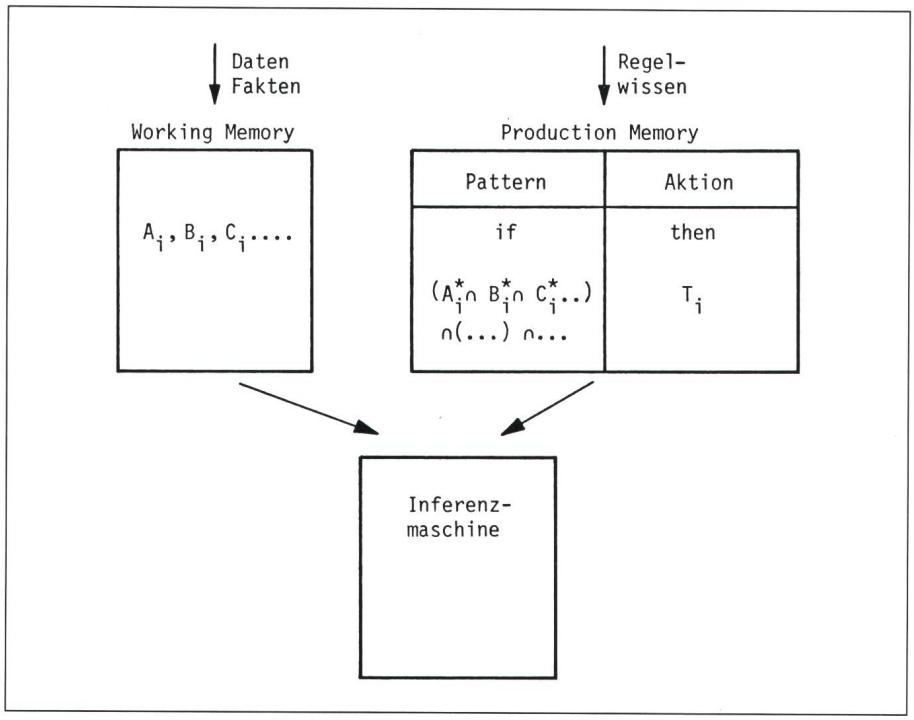
Ein Programm ist nicht ein Expertensystem, nur weil es Expertenwissen enthält; solches können auch konventionelle Programme beinhalten. Der Unterschied ist viel eher technischer Natur: es handelt sich um die *Art und Weise*, wie die Informationen dargestellt, organisiert, strukturiert, miteinander verbunden sind und auch wie sie benutzt werden. Oft wird der Begriff Expertensystem gleich wie der Begriff KI zu sehr mit der menschlichen Intelligenz verknüpft. Um Missverständnisse zu vermeiden, wird deshalb hier der Ausdruck *Wissensbasierte Systeme* (WBS) verwendet.

Wissensbasierte Systeme bilden einen Meilenstein in der Geschichte der Informatik, ähnlich wie damals die Konzepte der höheren Programmiersprachen, der Datenbanken usw. Sie sind aber keineswegs eine völlige Neuentwicklung der Softwaretechnik; es handelt sich vielmehr um eine konsequente Weiterverfolgung eines schon lange eingeschlagenen Weges. Daten hat man zunächst in Dateien, schliesslich ab etwa 1970 in eigenen Datenbanken, getrennt von den Anwendungsprogrammen, gespeichert. Mit den wissensbasierten Systemen geht man dazu über, auch das Wissen getrennt vom Anwendungsprogramm zu speichern. Dieses Wissen besteht jetzt nicht nur in Form von Daten und Fakten, sondern von Verarbeitungsregeln, welche in einer *Wissensbank* (Wissensbasis) abgelegt sind. Die eigentliche Verwendung dieses Wissens benötigt ein zusätzliches (aber genereller anwendbares) Programm, die sogenannte *Inferenzmaschine* (Inferenz bedeutet logisches Schliessen). Diese leitet aus den Fakten und Regeln neues Wissen ab, das die Fragen eines Benützers beantworten oder zumindest zur gesuchten Antwort führen soll.

In einem konventionellen Programm codiert der Programmierer einen Satz von Programminstruktionen, die in einer vorgegebenen Reihenfolge ausgeführt werden. Der Programmierer muss die Reihenfolge im voraus planen und spezifizieren. Er muss für jede denkbare Situation den

Lösungsweg vorgeben, d.h. die Reihenfolge der möglichen Instruktionsausführungen planen. Jede unvorhersehbare Situation kann zu einem Fehlerzustand führen, der unbedingt ausgeschlossen werden muss. Alle denkbaren Lösungswege müssen präzis codiert werden.

In wissensbasierten Systemen werden der *Auswertungsablauf* und die schrittweise Zerlegung eines Lösungsweges *situationsbedingt* von der Inferenzmaschine selber gesteuert. Diese Eigenschaft hat zur Folge, dass ein Programmierer nicht mehr gezwungen wird, für alle Kombinationen aller gegebenen Faktoren (Bedingungen) einen Programmablauf im voraus zu spezifizieren. Dieses Merkmal, das man als *Indeterminismus* bezeichnen kann, stellt eine ausserordentlich wichtige Charakteristik wissensbasierter Systeme dar. Ein Hauptvorteil wissensbasierter Systeme ist ihre grosse Flexibilität und Modularität. Sie können *inkrementell* aufgebaut werden, von Prototypen bis zu operativen Systemen.



Figur 1 Aufbau von wissensbasierten Systems

$A_i, B_i, C_i \ldots$	Working Memory Elements (WME)
$(A_i^* \cap B_i^* \cap C_i^* \ldots)$	Pattern
T_i	Aktion

Die Inferenzmaschine sucht im Working Memory nach Daten oder Fakten (z.B. A_i, B_i, C_i, ...), die bestimmten Patterns im Production Memory entsprechen. Wenn alle Bedingungen (Patterns) erfüllt sind, d.h. $A_i^* \subset A_i$, $B_i^* \subset B_i$ usw., kann die Aktion T_i ausgelöst werden.

Das Wesen von Wissensverarbeitungssystemen

Unter Wissen wollen wir pragmatisch die Gesamtheit der Informationen verstehen, die benötigt werden, um ein gegebenes Problem zu lösen oder um Fragen über ein Problemgebiet beantworten zu können. Diese Informationen umfassen Fakten, Schlussfolgerungen (logisches Verhalten) und Metaregeln (Regeln über die Verwendung von Regeln), wobei man von der Beobachtung ausgeht, dass der Mensch sein Verhalten in bestimmten Situationen in *Regeln* (Schlussfolgerungen) folgender Art formuliert:

Wenn ... Situation → dann ... Aktion

Solche Regeln werden im Fachjargon *Produktionsregeln* genannt. Das Vorliegen der auf der linken Seite spezifizierten Situation produziert die auf der rechten Seite genannte Aktion[1].

Die Situationen (Bedingungen), die eine Aktion auslösen, werden als *Patterns* oder Muster bezeichnet, wobei verschiedene Typen von Patterns vorkommen, nämlich

- einfache Muster, wie z.B. Zahlenwerte, mit denen die von der Aussenwelt kommenden aktuellen Daten verglichen werden. Beispiel: *Wenn* die Länge der Komponenten kleiner als 100 und ihr Gewicht zwischen 10 und 20 kg ist, *dann* ...
- komplexe Muster, die Variable und Funktionen von Variablen enthalten. Beispiel: *Wenn* es zwei Objekte gibt, die dieselben elektrischen Eigenschaften haben *und* ... *dann*

Die letzteren können komplexe Verarbeitungsoperationen zur Feststellung einer Übereinstimmung der Muster mit den aktuellen Daten verlangen. Die Art und Weise wie die Patterns spezifiert werden, hängt davon ab, wie der betreffende Weltausschnitt *modelliert* wird. In Anlehnung an die Datenmodellierung für relationale Datenbanken kann man die verschiedenen Informationen, die in den Patterns enthalten sind, durch Begriffe wie Entitäten (Objekte), Attribute, Beziehungen zwischen Entitäten und Klassen von Entitäten strukturieren. In wissensbasierten Systemen, die auf dem Konzept von Produktionsregelsystemen (Production Rule Systems) beruhen, werden zusammenhängende (d.h. zum gleichen Objekt gehörende) Daten in sogenannten *Working Memory Elements (WME)* zusammengefasst. Die WME sind also Objekte, welche durch Attribute ihrer Klasse definiert sind.

Ein WME kann mit einer Zeile einer relationalen Datenbank verglichen werden (es besitzt aber eine reichere Informationsstruktur). WME werden dynamisch kreiert und vernichtet und es können ihren Attributen Werte zugeordnet werden. WME werden im sogenannten *Working Memory* verwaltet.

Während im Working Memory also das Faktenwissen gespeichert ist, werden die weiter vorne beschriebenen Regeln separat im sogenannten *Production Memory* verwaltet. Die linke Seite jeder Regel besteht aus einer Konjunktion (Und-Verknüpfung) von Patterns (Bedingungen, Situationen); diese Patterns werden mit den vorhandenen WME verglichen. Alle Bedingungen (Patterns) einer Regel sind er-

[1] Diese Regeln (Wenn das und das geschieht, musst Du auf diese Weise reagieren!) können von der Maschine in gleicher Weise wie logische Ableitungen (Wenn diese und jene Bedingung erfüllt ist, dann gilt ...) verarbeitet werden.

füllt, wenn zu jeder ein WME gefunden werden kann, dessen Inhalt mit ihr übereinstimmt.

Der *Inferenzprozess* wird durch die Komponente *Inference Engine* (auch Production System Interpreter genannt) gesteuert. Dieses Kontrollsystem kontrolliert laufend die Erfüllung der Regeln und wählt - falls mehrere Regeln gleichzeitig erfüllt sind - eine der fälligen Aktionen aus. Das Kontrollsystem wiederholt dazu laufend den folgenden Zyklus:

- *Recognize oder Match Patterns:* um die Regeln zu finden, deren linke Seiten aus Patterns bestehen, die mit vorhandenen WME, d.h. mit gegebenen Situationen oder Bedingungen übereinstimmen,
- *Select (Conflict Resolution):* um eine als erfüllt befundene Regel zur Ausführung auszuwählen,
- *Act (Execute):* um die Aktionen (rechte Seite) der selektierten Regel auszuführen. Durch diese Aktionen können unter anderem neue WME kreiert, vorhandene vernichtet und neue Werte einem WME zugeordnet werden.

Die hier beschriebene Inferenzsequenz (vorwärtsgerichtete Form, Forward Chaining), bezeichnet man auch als datengetriebenen (Data-Driven) Schlussfolgerungsmechanismus.

Häufig, insbesondere bei Diagnoseproblemen, wird die rückwärts gerichtete Form des Inferenzmechanismus, Backward Chaining, angewendet. Ausgehend von der Aktionskomponente (rechte Seite der Regel) werden die Bedingungen (Patterns) gefunden, die für die Lösung dieses Problems erfüllt werden müssen. Jede Bedingung (Pattern) bildet dann ein neues Teilproblem des ursprünglichen Problems. Ein gegebenes Problem wird so rekursiv in Teilprobleme zerlegt.

Die vorhergehende Beschreibung von wissensbasierten Systemen sollte den *Informationssystemcharakter* von Expertensystemanwendungen genügend betonen, wobei allerdings nachzuführen ist, dass es noch andere Arten von Wissensdarstellung wie Frames, objektorientierte Darstellungen usw. gibt. Diese können als Erweiterung des WME-Konzeptes betrachtet werden, wobei bei allen das Wissen von der Programmsteuerungsstruktur getrennt ist. Beim Bau eines wissensbasierten Systems sollte man nicht vergessen, dass diese in ein gesamtes Informationssystem eingebettet ist und dass von dorther normalerweise auch die primären Daten stammen. Diese Einbettung stellt gewisse Anforderungen, die im folgenden Kapitel näher untersucht werden.

Integration in Informationssysteme

Von ihrem Konzept her erlauben wissensbasierte Systeme eine Vielzahl von Problemen zu lösen, welchen man in der Welt der konventionellen Informationssysteme begegnet. Erstaunlicherweise sind bis zum heutigen Datum aber nur eine vergleichsweise geringe Zahl von wissensbasierten Systemen bekannt, die aus dem Prototypstadium herausgewachsen sind und produktiv eingesetzt werden. Dies könnte mit der Neuheit der Technologie und dem Mangel an speziell ausgebildetem Personal erklärt werden. Eine bessere Erklärung aber ist, dass die meisten Entwickler von wissensbasierten Systemen mit speziellen Sprachen (Lisp, Prolog) und spezialisierter Hardware (z.B. Single-User-Lisp-Workstations) arbeiten. Die Folge davon ist, dass Anwender von den existierenden konventionellen Informationssystemen und damit auch von deren ausgebildetem Personal isoliert werden.

Dies kann vermieden werden, wenn das wissensbasierte System *innerhalb* der Architektur eines operationellen Informationssystems entwickelt wird.

Trotz der Tatsache, dass die wissensbasierten Systeme Charakteristiken aufweisen, welche sie einzigartig machen, benötigen sie doch immer häufiger eine Verbindung zu existierenden Informationssystemen, sei es, um Daten aus diesen Systemen zu verwenden, oder für andere Verarbeitungen aufzubereiten. Es liegt auf der Hand, dass mit der wachsenden Anzahl von Benützern eines wissensbasierten Systems die Integration mit vorhandenen Informationssystemen unerlässlich wird.

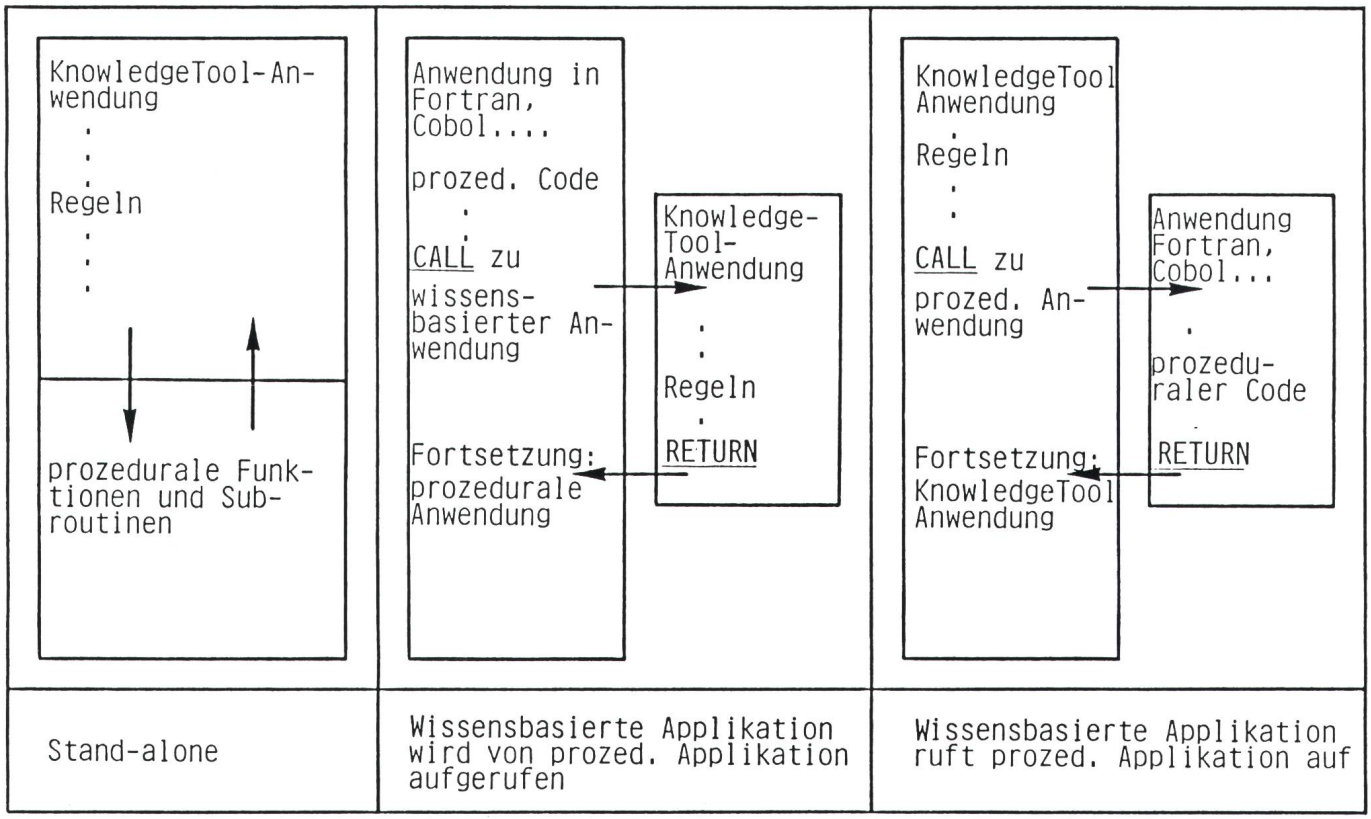
Integration bei Systemen mit Datenbanken

In jeder Organisation sind heute enorme Mengen an Informationen elektronisch gespeichert. Diese Informationen stellen eine der wichtigsten Ressourcen des Unternehmens dar. Es ist also zwingend notwendig, aus einem wissensbasierten System auf diese Informationen zugreifen zu können. Dazu müssen die wissensbasierten Systeme mit Standardschnittstellen und Zugriffsmethoden ausgestattet sein. Wissen lässt sich - wie früher beschrieben - z.B. mit Hilfe von Regeln darstellen; andererseits sind aber Fakten (WME) notwendig, um mit Hilfe dieser Regeln neue Fakten ableiten zu können. Informationen aus Datenbanken werden von wissensbasierten Systemen als Faktenwissen betrachtet und können von diesen genauso verarbeitet werden wie Benutzereingaben oder Konstanten. Da diese Daten sehr oft dynamisch durch andere Systeme verändert werden, ist dies nur sinnvoll, wenn die Daten direkt aus der Datenbank in WME überführt und ins Working Memory geladen werden können (Aktualität der Daten). Dies bedeutet, dass die Schnittstellen Prozeduren beinhalten müssen, welche die Manipulation der Datenbank bewerkstelligen. Sind Datenbanken und das wissensbasierte System nicht auf der gleichen Hard- und Softwareumgebung implementiert, ist dies bis heute nur unter grossen Schwierigkeiten möglich (unterschiedliche Architektur bedeutet erschwerte Kommunikation). Ausserdem ist in diesem Fall auch keine Gewähr gegeben, dass mit den aktuellsten Daten gearbeitet wird, da sich während des Datentransfers der Datenbankinhalt bereits wieder ändern kann. Wissensbasierte Anwendungen in Kooperation mit Datenbanken kommen vor allem dort in Frage, wo es gilt, im Unternehmen gespeicherte Informationen auszuwerten, zu interpretieren oder umzusetzen, also beispielsweise

- bei der Verwendung von Daten anstelle Benutzereingaben
- bei der Auswertung statistischer Daten
- bei der Interpretation von Messwerten
- bei der Analyse von Unternehmensdaten zur Unterstützung der Planung
- bei der Konfiguration komplexer Anlagen.

Einbettung in Programme mit Standard-Programmiersprachen

Es ist unbestritten, dass eine Informationsverarbeitung zur Lösung von komplexen Aufgaben immer prozedurale und deklarative Aspekte beinhaltet. Für eine optimale Anwendung müssen daher beide Arten der Programmierung zu beliebigen Zeitpunkten einer Verarbeitung möglich sein.



Figur 2 Integration von prozeduralen und wissensbasierten Anwendungen

Dies bedeutet aber, dass eine Einbettung wissensbasierter Systeme in konventionelle Umgebungen möglich sein muss, eine Bedingung, welche die meisten heute erhältlichen Werkzeuge für die Entwicklung von WBS nicht erfüllen. Unter Einbettung wird hier die Fähigkeit einer Softwarekomponente verstanden, in eine grössere Systemumgebung transparent integriert zu werden. Diese Fähigkeit verlangt, dass die wissensbasierte Applikation eine aufrufbare Subroutine für das umgebende System darstellt und ihrerseits konventionelle Programme zur Erledigung spezifischer Aufgaben aufrufen kann.

Am Beispiel von IBM KnowledgeTool, einem auf der Programmiersprache PL/1 basierenden Werkzeug zur Entwicklung von wissensbasierten Systemen, sind in Figur 2 die drei Möglichkeiten einer Integration von prozeduraler und deklarativer Programmierung aufgezeigt. IBM KnowledgeTool erlaubt:

- innerhalb einer wissensbasierten Applikation prozedurale Subroutinen und deklarative Programmierung beliebig zu mischen
- aus einem Programm innerhalb einer Applikation (geschrieben in einer prozeduralen Sprache wie Cobol, Fortran, Pascal usw.) eine (deklarative) KnowledgeTool-Applikation aufzurufen und wieder zum ursprünglichen Programm zurückzukehren
- aus einer KnowledgeTool-Applikation ein externes Programm mit prozeduralem Code aufzurufen und wieder zur wissensbasierten Applikation zurückzukehren.

Einbettung in Transaktionsverarbeitung

Von gleicher Bedeutung wie die Integration von wissensbasierten Systemen und prozeduralen Applikationsprogrammen ist die Einbettung von WBS in die Transaktionsverarbeitung. Noch Anfang 1980 waren ein grosser Teil der Programme in einer operationellen Datenverarbeitungsumgebung Batchprogramme. Mit dem Übergang zur Informationsverarbeitung gewannen die Online-Informationsverarbeitungssysteme mit direktem Zugriff auf Datenbanken mehr und mehr an Boden. In den meisten Unternehmen müssen mehrere Benützer gleichzeitig auf dieselben Datenbanken zugreifen und dieselben Verarbeitungsprogramme verwenden können. Dazu werden Datenkommunikations- oder Online-Transaktionsverarbeitungssysteme benötigt, welche Aufgaben der Terminalkontrolle, Datenverwaltung und Applikationsprogrammunterstützung übernehmen.

Greifen mehrere Anwender auf dieselben Daten zu, muss gewährleistet sein, dass nicht mehrere Benützer denselben Datenbestand gleichzeitig verändern. Die Aufgabe von solchen On-line-Transaktionsverarbeitungssyste-

men ist nun, sicherzustellen, dass eine Änderung abgeschlossen ist, bevor ein anderer Benutzer die gleichen Daten manipulieren kann. Ebenso müssen Recovery-Möglichkeiten im Falle eines Transaktionsabbruchs oder bei auftretenden Systemfehlern vorhanden sein, um die Integrität der Daten zu gewährleisten.

Damit mehrere Anwender gleichzeitig mit demselben Programm arbeiten können, ohne dass dafür jedesmal eine neue Kopie des Programmcodes geladen werden muss, hat das System sicherzustellen, dass für jeden Benutzer ein kontrollierter Pfad durch den verwendeten Programmcode unterhalten wird.

Damit ein wissensbasiertes System von mehreren Anwendern gleichzeitig benutzt werden kann, muss das Werkzeug zu seiner Entwicklung die oben beschriebenen Anforderungen in gleicher Weise unterstützen (wie z.B.: IBM KnowledgeTool und IBM Expert System Environment).

Systemanforderungen

Die in den vorhergehenden Kapiteln gegebenen Hinweise über die Integration von wissensbasierten Systemen in Informationssysteme setzen unmittelbar voraus, dass jene auf der konventionellen, für das bestehende Informationssystem verwendeten Hardware und unter demselben Betriebssystem implementiert sind. Unter anderem kann so das bestehende Kommunikationsnetzwerk verwendet werden. Damit werden auch die komplexen Probleme, welche durch die neue Architektur einer Stand-alone-special-prepose-Umgebung bezüglich Netzwerk-Kompatibilität und Konnektivität entstehen können, umgangen.

Wissensbasierte Systeme von IBM

Zurzeit sind bei IBM mehr als 100 wissensbasierte Systeme im internen Einsatz. Im folgenden Abschnitt sind einige Beispiele aufgeführt.

Diagnostic Expert for Final Test (DEFT): Es unterstützt den Abschlusstest für die IBM 3380 Plattenspeicher, wobei es anhand der auftretenden Fehlercodes den Techniker bei der Fehlersuche und dem Auffinden fehlerhafter Komponenten berät.

IBM 9370 Konfigurator: Der IBM-9370-Computer ist ein sehr flexibles System. Entsprechend den wechselnden Kundenbedürfnissen müssen verschiedene Kombinationen von Karten und Adaptern in das Gehäuse montiert werden, eine Tätigkeit die mit Hilfe des Konfigurators durchgeführt wird. Vom Werk werden beim Bau des Computers die nötigen Informationen direkt vom Konfigurator abgefragt.

YES/MVS: Dieses Real-time-WBS unterstützt den Operator eines Computersystems, das aus mehreren Teilsystemen besteht, wobei das Expertensystem den Operator bei den nötigen Aktionen berät.

Glendale Help Desk: Es unterstützt den Benützer bei Hard- und Softwareproblemen; falls es das Problem nicht zufriedenstellend lösen kann, fordert es einen menschlichen Experten an. Da die Probleme automatisch in einer Datenbank festgehalten werden, können mit statistischen Auswertungen häufige Fehlerursachen entdeckt und behoben und zudem die Wissensdatenbank erweitert werden.

Schlussfolgerungen

In Zukunft werden wissensbasierte Systeme mit der konventionellen Datenverarbeitung zusammenwachsen. Wissensverarbeitung und Datenverarbeitung müssen für den Anwender unbemerkbar ineinander übergreifen. Die bestehende Informationssoftware wird wissensbasierte Systeme als Routinen aufrufen (oder umgekehrt), wenn Situationen auftreten, die ein algorithmisches Verarbeiten inadäquat erscheinen lassen.

Innerhalb von IBM werden eine ganze Reihe integrierter wissensbasierter Systeme eingesetzt. Die damit gemachten positiven Erfahrungen zeigen, dass diese Technologie einen solchen Reifegrad erreicht hat, dass entsprechende Investitionen sehr lohnend sein können.