Kapitel MK:II

II. Wissensrepräsentation

- Wissensrepräsentation in der Klassifikation
- Symbolisch versus subsymbolisch
- Problemlösungswissen
- Kennzeichen von Problemlösungswissen
- Prinzipien wissensbasierter Systeme
- Expertensysteme
- Problemklassen für Expertensysteme
- Erstellung wissensbasierter Systeme

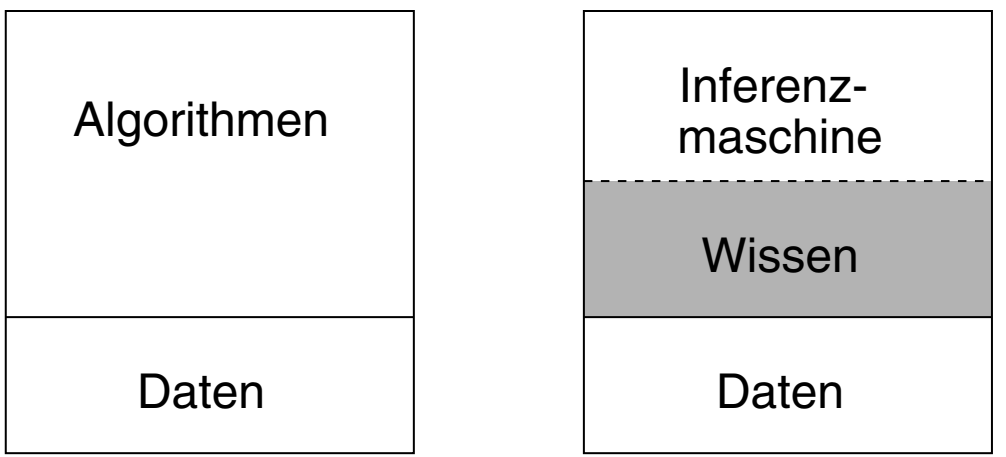
Prinzipien wissensbasierter Systeme

Wunsch: Operationalisierung von Problemlösungswissen

Prinzipien:

1. Trennung von anwendungsbezogenem und anwendungsunabhängigem Wissen
2. Anwendung von Problemlösungsprinzipien der KI

wissensbasierte Sicht auf Softwaresysteme:



(konventionell) *(wissensbasiert)*

Prinzipien wissensbasierter Systeme

Definition 2 (wissensbasiertes System, WBS)

Ein wissensbasiertes System (WBS) ist ein Softwaresystem, bei dem das Fachwissen über das Anwendungsgebiet (Domain Knowledge) *explizit* und *unabhängig* vom allgemeinen Problemlösungswissen dargestellt wird.

Prinzipien wissensbasierter Systeme

Fragen zu konventionellen Softwaresystemen:

(a) Enthalten sie kein Wissen?

(b) Wenn nein, warum nicht?

(c) Wenn ja, wo ist dieses Wissen zu finden,

(d) . . . und woher kommt es?

Prinzipien wissensbasierter Systeme

Fragen zu konventionellen Softwaresystemen:

(a) Enthalten sie kein Wissen?

(b) Wenn nein, warum nicht?

(c) Wenn ja, wo ist dieses Wissen zu finden,

(d) . . . und woher kommt es?

zu (c) Wo ist das Wissen?

Sowohl das Wissen über das Anwendungsgebiet als auch das allgemeine Problemlösungswissen ist in Algorithmen, Daten und Datenstrukturen *verteilt* und *implizit* codiert.

Prinzipien wissensbasierter Systeme

Fragen zu konventionellen Softwaresystemen:

(a) Enthalten sie kein Wissen?

(b) Wenn nein, warum nicht?

(c) Wenn ja, wo ist dieses Wissen zu finden,

(d) . . . und woher kommt es?

zu (c) Wo ist das Wissen?

Sowohl das Wissen über das Anwendungsgebiet als auch das allgemeine Problemlösungswissen ist in Algorithmen, Daten und Datenstrukturen *verteilt* und *implizit* codiert.

zu (d) Woher kommt es?

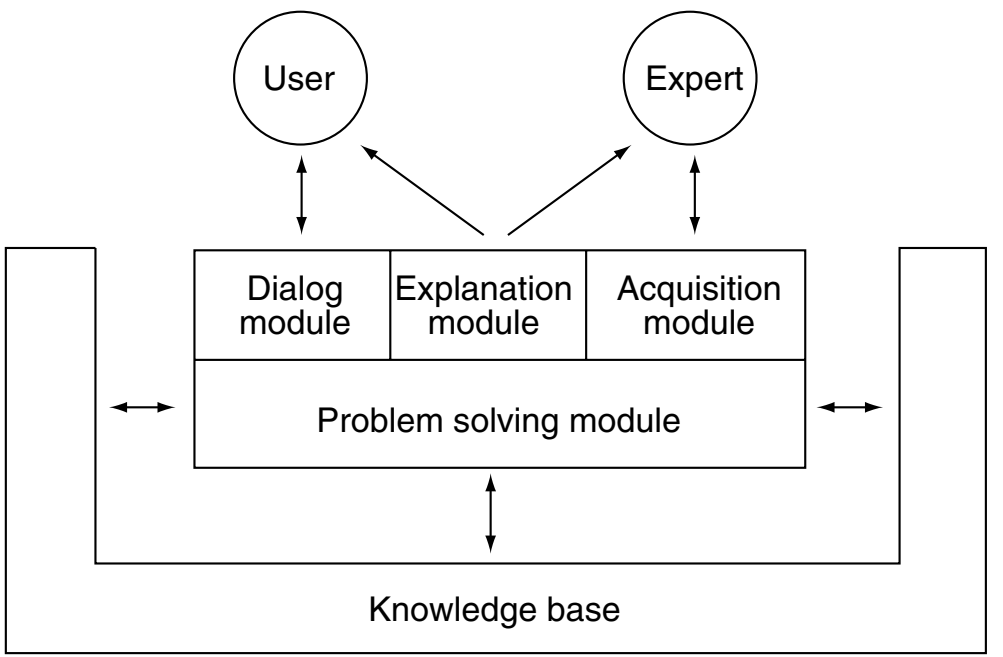
- Vorgaben/Rahmenbedingungen des Auftraggebers
- Fachabteilungen/Labore liefern z. B. betriebliche Zusammenhänge, „Rezepturen“ und Problemlösungswissen
- Programmierer vertieft Problemlösungswissen durch gezielte Nachfragen, besitzt aber auch eigenes Problemlösungswissen.

Expertensysteme, XPS

„. . . ein intelligentes Computerprogramm, das Wissen und Inferenzverfahren benutzt, um Probleme zu lösen, die immerhin so schwierig sind, dass ihre Lösung ein beträchtliches menschliches Fachwissen erfordert . . . “

[Edward Feigenbaum]]

XPS-Architektur aus funktionaler Sicht [Puppe 1991]:



Expertensysteme

Expertensysteme versuchen, die Eigenschaften eines Experten hinsichtlich einer eingeschränkten Tätigkeit möglichst gut nachzubilden.

Gewünschter Nutzen:

- Experten sollen von Routinetätigkeiten entlastet werden
- Aufheben der örtlichen und zeitlichen Abhängigkeit von Experten
- kostengünstiger als menschliche Experten
- Erstellung einer ausgewogenen (objektiven?) Expertise
- Schulungsaspekte (Unterstützung bei der Ausbildung)
- unbegrenzte Verbreitbarkeit (ist das immer gewünscht?)

Expertensysteme

Der ideale menschliche Experte

- besitzt Wissen über Problembereich (Domäne)
  - Aneignung durch Ausbildung und Erfahrung
  - kennt Begriffswelt und die Zusammenhänge zwischen den Begriffen
  - arbeitet mit vagem Wissen
  - kennt die Zusammenhänge zu anderen Wissensgebieten
  - hat Hintergrundwissen über Sachverhalt (Common Sense)
  - kennt Verweise auf Wissensquellen
- kann Wissen erwerben und falsches Wissen korrigieren
- kann Probleme durch Verknüpfen von Wissen lösen
- kann seine Ergebnisse durch Verdeutlichen seines Lösungsweges erklären
- handelt effizient:
  - schnelle Problemlösung durch Anwendung von Heuristiken
  - Wahl geeigneter Vorgehensweisen
- handelt menschlich (Das steht in keinem KI-Buch!)

Expertensysteme

Expertensystem versus Experte

	XPS	Experte
Wissenserhaltung	+	–
zeitliche/örtliche Unabhängigkeit	+	–
Unabhängigkeit von Umwelteinflüssen	+	–
Hintergrundwissen	(–)	(+)
Lernfähigkeit	o	+
Größe des Problemfeldes	–	+
Kreativität	–	+

- Expertensysteme sind keine Allheilmittel.
- Es existieren viele Probleme, die mittels der Expertensystemtechnologie (noch) nicht lösbar sind.

Expertensysteme

Expertensystem versus Standardprogramm

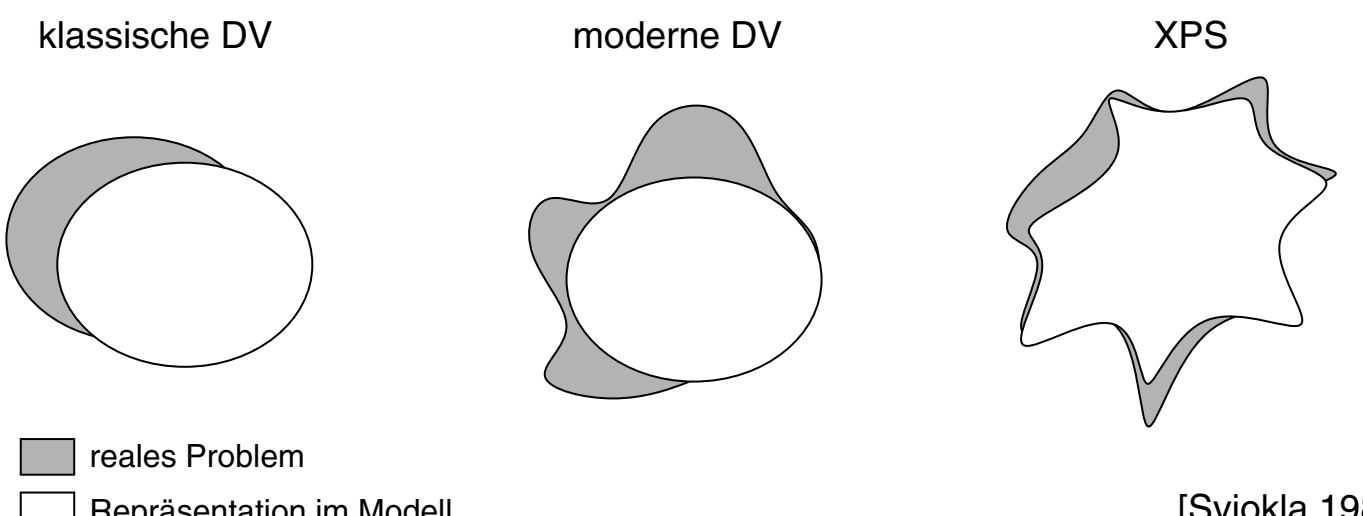
Expertensysteme	**konventionelle Programme**
Repräsentation und Verarbeitung von Wissen	Repräsentation und Verarbeitung von Daten
Algorithmen, Heuristiken	Algorithmen
inferenzielles Bearbeiten	iteratives Bearbeiten
schlecht strukturierte Probleme	gut strukturierte Probleme
viele unterschiedliche Wissensarten	große Mengen ähnlicher Daten
individuelle Verarbeitung und Interpretation der Daten	kanonische Verarbeitung, z. B. Number Crunching
keine Patentrezepte	

Expertensysteme

Expertensystem versus Standardprogramm

Für gut strukturierte Probleme gilt:

- Es existiert ein Leitfaden zur Problemlösung.
- Die Zielfunktion ist bekannt.
- Problem und Lösung sind in numerischen Größen beschreibbar.
- Formalisierung ist möglich, sowohl für Modell als auch Zielfunktion(!)
- Algorithmen zur Problemlösung sind bekannt $\Leftrightarrow$ Kalkülisierung ist möglich.

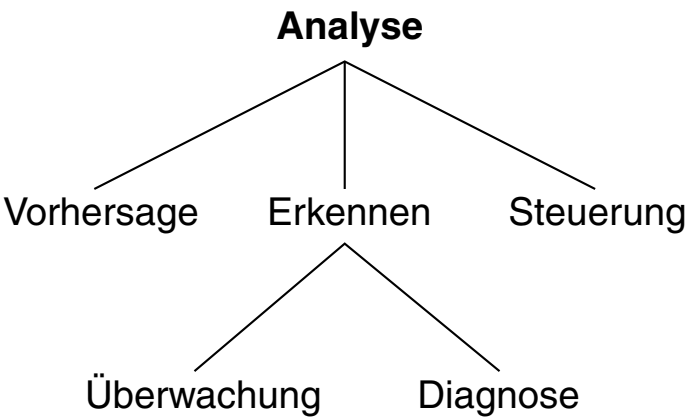


[Sviokla 1986]

Problemklassen für Expertensysteme

Analyse

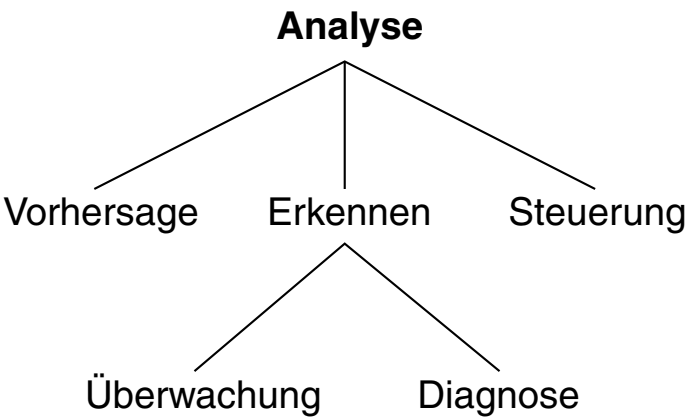
Beim analytischen Problemlösen werden aufgrund von Eingaben Lösungen selektiv bestimmt. Die charakteristische Eigenschaft dieser Problemklasse ist die endliche und verhältnismäßig kleine Anzahl potentieller Lösungen. [Karbach/Linster 1990]



Problemklassen für Expertensysteme

Analyse

Beim analytischen Problemlösen werden aufgrund von Eingaben Lösungen selektiv bestimmt. Die charakteristische Eigenschaft dieser Problemklasse ist die endliche und verhältnismäßig kleine Anzahl potentieller Lösungen. [Karbach/Linster 1990]



Weitere Einteilung analytischer Problemlösemodelle:

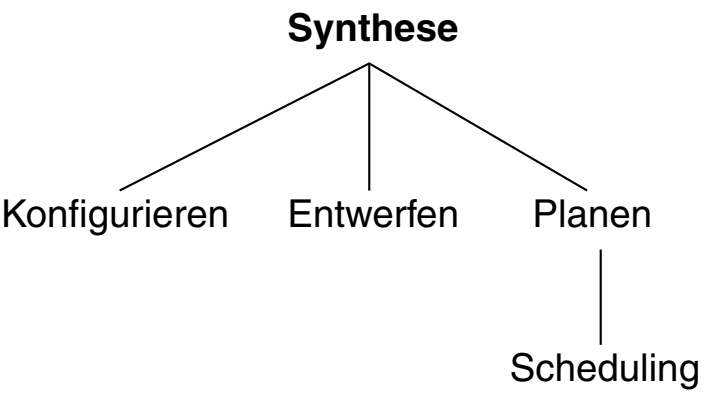
- diagnostisches Problemlösen: Zuordnung von Beobachtungen zu erklärenden Ursachen.
- assoziatives Problemlösen: Zuordnung eines Ausgabewerts für einen Eingangsvektor; typisch ist das Fehlen von Zwischenstufen.

Problemklassen für Expertensysteme

Synthese

Im Gegensatz zu analytischen Problemlöseverfahren, bei denen eine Lösung aus einer bekannten Lösungsmenge ausgewählt wird, muss beim Entwurf eine vorher nicht bekannte Lösung konstruiert werden.

[Karbach/Linster 1990]

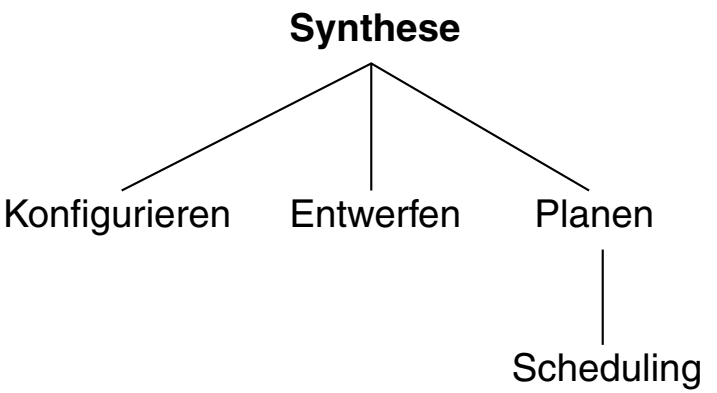


Problemklassen für Expertensysteme

Synthese

Im Gegensatz zu analytischen Problemlöseverfahren, bei denen eine Lösung aus einer bekannten Lösungsmenge ausgewählt wird, muss beim Entwurf eine vorher nicht bekannte Lösung konstruiert werden.

[Karbach/Linster 1990]



Weitere Aufteilung der Entwurfsarten:

- kreativer Entwurf: völlige Neuschöpfung, patentierbare Erfindung
- innovativer Entwurf: Teillösungen bekannt, Kombination/Anpassung neu
- Änderungsentwurf: eine vorhandene Lösung wird verändert
- Routineentwurf: Auswahl und Dimension gemäß bekannten Varianten

Bemerkungen:

- Die Problemklassen bestimmen die in Exeprtensystemen zum Einsatz kommenden Problemlösungsverfahren.
- Die Einteilung in Problemklassen ist nicht exakt. Zum Beispiel gehört die Wettervorhersage zur
  - Interpretation (Eingrenzung Hoch- und Tiefdruckgebiete),
  - Simulation (Abschätzung zukünftiger Wetterlagen),
  - Beobachtung (Sturmwarnung) und
  - Diagnose (Fehlerermittlung bei falschen Voraussagen).

Erstellung wissensbasierter Systeme

Die Erstellung von Programmen, die eine Lösung schlecht strukturierbarer bzw. wissensintensiver Probleme automatisiert, erfordert:

1. Lösung des Akquisitionsproblems.
   Identifikation und Erwerb des Wissen, das zur Problemlösung notwendig ist.

2. Lösung des Repräsentationsproblems.
   Entwicklung einer geeigneten Formalisierung/Codierung des Problemlösungswissens.

3. Lösung des Inferenzproblems.
   Auswahl und/oder Entwicklung von Algorithmen zur Verarbeitung des Problemlösungswissens.

Bemerkungen:

- Die Lösung des Akquisitions-, Repräsentations- und Inferenzproblems ist keine Garantie für den Erfolg eines wissensbasierten Systems.
- Man unterscheidet zwei grundsätzlich verschiedene Ansätze für die methodische Entwicklung von Expertensystemen:
  - Rapid Prototyping
  - Modellierung von Expertise

Erstellung wissensbasierter Systeme

Lösung des Akquisitionsproblems [Karbach/Linster 1990]

Wissenserhebung mit Bleistift und Papier.

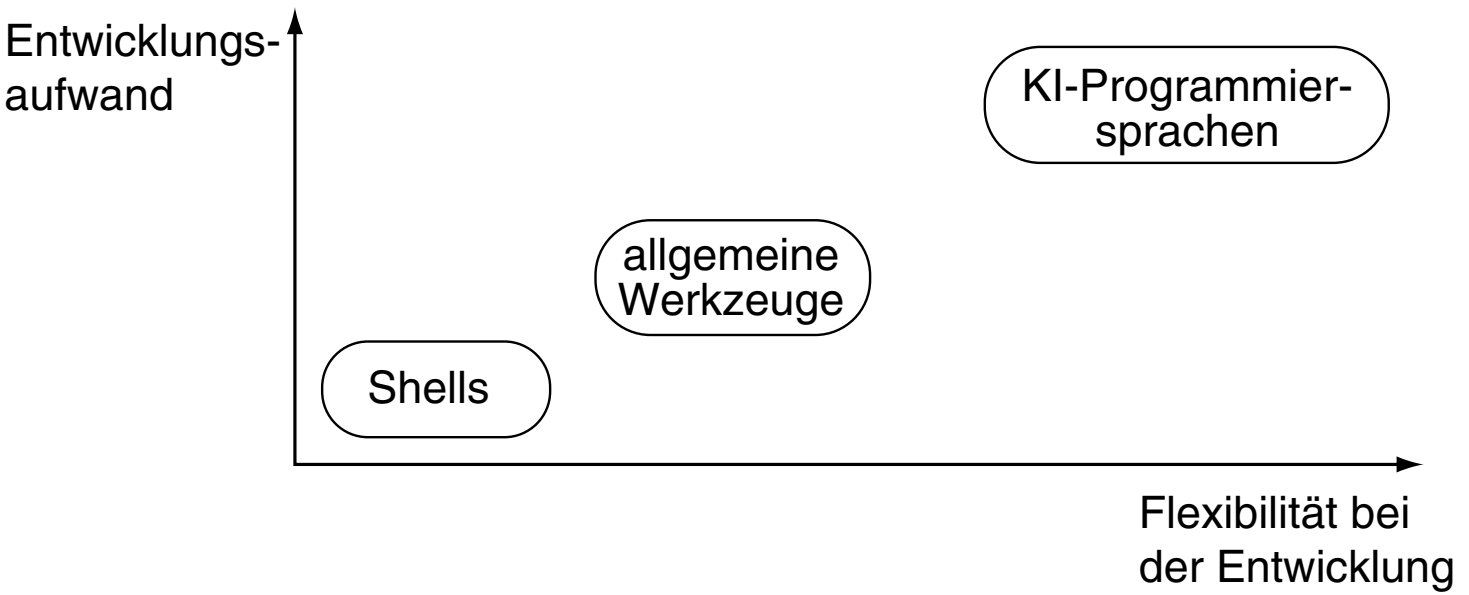
1. Interviewtechniken:
   - unstrukturiertes Interview
   - strukturiertes Interview
   - fokussiertes Interview

2. Beobachtung des Experten:
   - Protokollanalyse. Experte schildert sein Vorgehen bei der Problemlösung.
   - Introspektion. Experte schildert, wie er denkt, dass er vorgeht.
   - Dialoganalyse. Experte wird beim Dialog mit Klienten beobachtet.

3. Indirekte Erhebung:
   Weil Experten nicht immer beschreiben können, was sie tun.
   Weil Wissensingenieure Experten beeinflussen bzw. deren mentalen Prozess falsch darstellen.

Erstellung wissensbasierter Systeme

Werkzeuge



Bemerkungen:

(a) Shells (EMycin, Nexpert Object, Twaice):
- sind spezielle Programme zur Entwicklung von XPS
- Konzepte für Wissensrepräsentation und Inferenz vorgegeben
- Entwickler konzentriert sich auf die Erfassung, Strukturierung und Eingabe des Wissens aus dem Anwendungsgebiet

(b) allgemeine Werkzeuge (KEE, LOOPS, Knowledge Craft):
- viele Konzepte zur Wissensrepräsentation und Inferenz, Elemente für den Bau von Benutzeroberflächen
- mehr Flexibilität als bei Shells, hoher Einarbeitungsaufwand

(c) KI-Programmiersprachen (LISP, PROLOG, Smalltalk):
- u. a. leistungsfähige Symbolverarbeitung
- erfordern professionelle Softwareentwickler

Erstellung wissensbasierter Systeme

Aus meiner Erfahrung

- Für Anfänger: Werkzeuge können mehr schaden als nutzen.
- Für erfahrene Softwaretechniker: Werkzeuge schaden nicht.
- Der Nutzen von Werkzeugen?
  Wenn überhaupt, dann in der Prototyp-Entwicklung.
- Beherrschung von Methoden und Techniken ist notwendig – nicht hinreichend.

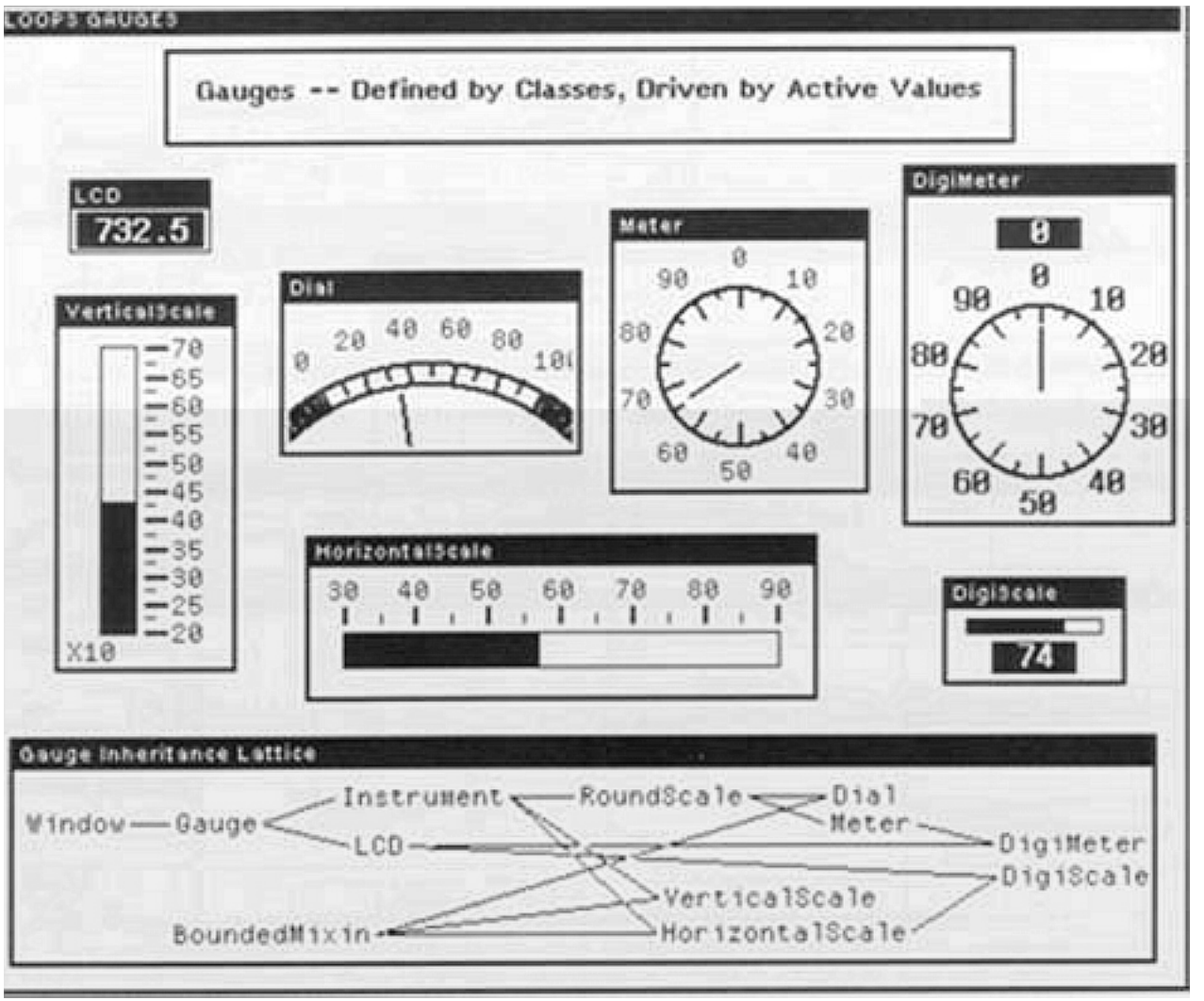
  Intelligenz, Können, Lernbereitschaft und Hartnäckigkeit des Expertensystem-Entwicklers (Wisseningenieurs) sind die wichtigsten Erfolgsfaktoren.

Dabei bestimmt

- das Anwendungsgebiet (Domäne) die Komplexität,
- das Problem den Lösungsaufwand.

Erstellung wissensbasierter Systeme

LOOPS [PARC 1982-1986]



Erstellung wissensbasierter Systeme

Colab [PARC 1982-1986]

Bemerkungen:

- Blickt man zurück, kann man von dem Fortschritt enttäuscht sein.
- Hintergund zum LOOPS-Projekt:
  - Main Participants: D. Bobrow, Sanjay Mittal, Stanley Lanning, Mark Stefik.
  - Object-oriented programming: Classes and objects, class variables, instance variables, methods, multiple-inheritance, interactive class browsers
  - Access-oriented programming: Nestable active values that can be attached to variables, procedures specified in the active value are triggered.
  - Rule-oriented programming.