Wissensbasierte Systeme I

Zusammenfassung wichtiger Elemente der Einführung in wissensbasierte Systeme

von Alexander Köster

Student der Universität Siegen, frei nach Vorlesung WBS1 2020
von Prof. Dr.-Ing. Madjid Fathi, Johannes Zenkert

Letzte Aktualisierung: 17. Juli 2020

Dieser Kurs der Informatik befasst sich mit den Grundlagen von wissensbasierten System und Wissensakquisition, den Definitionen und verschiedenen Sichtweisen von Wissen.

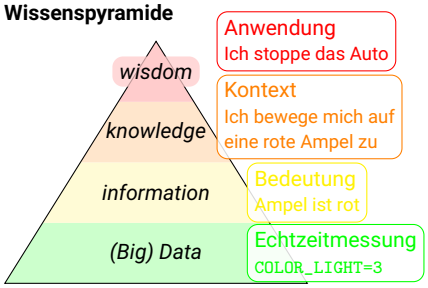
Einzig für Lernzwecke erstellt.

Nicht geeignet als Klausurhilfe.
Das Erstellen einer eigenen Klausurhilfe führt zu einem besonders guten Lerneffekt.
Dieses Dokument sollte nur als Orientierung oder Vergleich dienen.
Jeder sollte seine Klausurhilfe individuell auf seinen Lernstand und seine eigenen Probleme anpassen, gut bekannte Dinge auslassen und schlecht merkbare Dinge hinzufügen.
Dieses Dokument beinhaltet viel mehr, als für eine tatsächliche Klausur durchschnittlich benötigt wird.
Für einen guten Ansatz, welche Inhalte man braucht, sollte sich an der Probeklausur orientiert werden.

1 Einführung

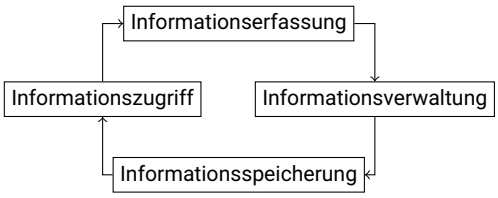
1.1 Definition von Wissen

- „Wissen ist die Bedingung oder die Tatsache, etwas mit Vertrautheit zu wissen, das durch Erfahrung oder Assoziation erworben wurde."
- Zwei Ebenen von Wissen (*knowledge*):
  - ▷ oberflächl. Kenntn. (*shallow knowledge*)
  - ▷ fundierte Kenntnisse (*deep knowledge*)
- Weitere Unterteilungen:
  - ▷ ***declarative knowledge**: knowing what*
    Bsp.: Register über Komponenten/verbaute Bauteile, Logdatei einer Maschine
  - ▷ ***procedural knowledge**: knowing how*
    Bsp.: Effizienteste Durchführung eines Vorgangs, gute Geschäftsbuchführung
  - ▷ ***conditional knowl.**: knowing when & why*
    Bsp.: Anpassung der Temperatur eines Brennofens aufgrund von Materialverfärbung, Verteilung von Arbeitskräften in einer Engpasssituation
- **Wissenspyramide**



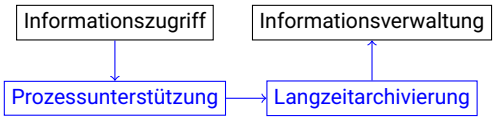
- Weitere Unterteilung:
  - ▷ **explizites Wissen**
    - – „aufschreibbares Wissen"; alles, was formalisierbar ist (Datenbank, wissenschaftliche Publikation, …), sehr gut organisierbar mit **Wissensmanagementsystemen** (KMS)
    - – häufigste Form des Wissens
    - – *"know-what"*
  - ▷ **implizites Wissen**
    - – intuitives, schwer zu definierendes Wissen
    - – Erfahrungswissen, kontextabhängig & von persönlicher Natur
    - – tief verwurzelt in *action, commitment, involvement*
    - – *"know-how"*
    - – wertvollste Wissensquelle, beeinflusst stark die Innovations- & Wettbewerbsfähigkeit von Unternehmen, wenn diese viele Mitarbeiter mit gutem implizitem Wissen haben
- Verschiedene Sichtweisen auf Wissen:
  - ▷ **Alltagswissen**
    Bsp.: Wasser wird unter 0° zu Eis
  - ▷ **Lexikalisches Wissen**
    Bsp.: „Schimmel" ist auch ein weißes Pferd; Nokia hat früher Gummi hergest.
  - ▷ **Fachwissen**
    Bsp.: Programmierung von Algorithmen, fachmathematisches Wissen, kunsthistorisches Expertise, …
  - ▷ **Erfahrungswissen**
    Bsp.: Fassen auf die heiße Herdplatte
  - ▷ **Handlungswissen**
    Bsp.: korrektes Händewaschen, effektive Anwendung von Kampfsport
  - ▷ **räumliches Wissen**
    Bsp.: mentales Vorstellen von Objekten, Vorstellung Bremsweg mit Fahrrad/Auto
  - ▷ **episodisches Wissen**
    erinnerbare Erlebnisse, **Bsp.:** COVID-19
- Wissen wird für **Entscheidungsfindung** in **Entscheidungsprozessen** verwendet
  - ▷ Daten müssen verfügbar & interpretierbar sein → Informationen
  - ▷ Informationen müssen glaubwürdig, qualitativ wertig (keine *„fake news"*) sein → Wissensentdeckung
  - ▷ Wissen muss verfügbar und für die Situation relevant sein → Entscheidung
  - ▷ Entscheidung sollte nützlich sein, dazu relevant: Informationsgehalt/Validität
- **Entscheidungsanomalien**: rational nicht erklärbare Entscheidungen (z. B. emotionale)
- **Wissensingenieur** (***knowledge engineer***): Bindeglied zwischen Experte & Wissensbasis
  - ▷ Durchführung von Interviews, um das Wissen der Experten zu extrahieren
  - ▷ Überführung des Wissens in die Wissensbasis
- **Wissensmodellierung** (*knowledge engineering*):
  1. Vermittlung zwischen Experte und Wissensbasis
  2. Erfassung, Strukturierung von explizitem & implizitem Wissen (**Wissensakquise**) in Befragungen & Interviews
  3. Formalisierung und Abbildung im Computer (**Wissensrepräsentation**, Aufbau einer Wissensdatenbank), d. h. Wissen wird für die Wissensbasis vorbereitet und in Struktur überführt (die korrekte Repräsentationsform zu finden kann schwer sein, **Problem der semantischen Lücke**)
  4. Wissen innerhalb Wissensdatenbank verfeinern zusammen mit Experte für akzeptable Darstellung & Einsetzbarkeit
- **Wissensbasis** (z. B. Datenbank) stellt formalisiertes Wissen bereit zur Lösung klar definierter Probleme (z. B. durch **Inferenzmaschine**).

1.2 Informationslebenszyklusmanagement



- **Informationserfassung:** Autorensysteme (*authoring*) für die Erstellung von Dokumenten, **Bsp.:** Produktangebot für Kunden; außerdem:
  - ▷ Metadaten, **Bsp.:** Angebot-Nr.
  - ▷ Annotationen, z. B. Schlagworte (*tags*), **Bsp.:** Name des Unternehmens

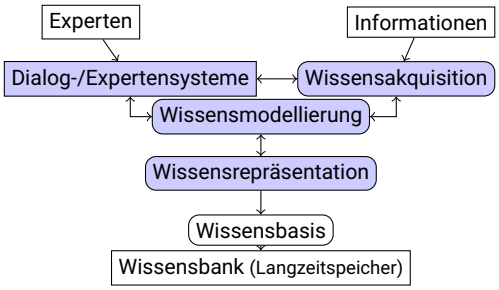
  Markup, Modellierung v. Daten, Digitalisierung:
  - ▷ textuell, z. B. durch Scans, *Optical Character Recognition* (OCR)
  - ▷ nicht-textuell, z. B. Messdaten, Bilder, Video & Audio (*Sampling*)
- **Informationsverwaltung:** Wissensrepräsentation und Organisation der Informationen, **Bsp.:** *Enterprise-Resource-Planning*-System (ERP-System), Aufbereitung für Suche (Index)
- **Informationsspeicherung:** Ablage in RAM/Dateien/Datenbanken/…, je nach der nötigen Speicherdauer und -größe,
  Bsp.: Angebot *vs.* Angebotshistorie
- **Informationszugriff:** Zugriff in Empfehlungssystemen, vorgeschlagene Informationen anhand Zugriffszahlen, ***Information Retrieval***
  - ▷ **Zugriffsmanagement** zum Verhindern unerwünschter Nutzung



- **Prozessunterstützung:** Integration in *Workflow-* & *Business-Process*-Management (BPM)
  - ▷ Bereitstellung von Informationen für Geschäftsprozesse, **Bsp.:** Angebot ist relevant für den Vertrieb/Außendienst
- **Langzeitarchivierung:** dauerhafte elektronische Speicherung von NOTWENDIGEN Informationen und Informationsquellen
- ***Information Retrieval*:** Merkmale:
  - ▷ der Benutzer kann sein „diffuses" Informationsbedürfnis ggf. nicht präzise und formal (z. B. als SQL-Query) ausdrücken, Anfrage enthält daher vage Bedingungen
  - ▷ dem System fehlen Kenntnisse über Dokument**inhalte** ⇒ fehlerhafte Antworten, Probleme bei Textsuchen sind z. B. oft **Synonyme** (**Bsp.:** „Bank", „Geldinstitut")

1.3 Wissensbasierte Systeme

- Ein **WBS** ist ein System, das sowohl **Daten** als auch eine **abgeleitete Logik von Experten** bzw. eine **Zusammenstellung an Wissen aus verschiedenen Quellen** gemeinsam nutzt.
- Unterscheidungen:
  - ▷ **regelbasierte Systeme**, **Bsp.:** Treffen von Entscheidungen anhand fester Regeln
  - ▷ **Expertensysteme**, **Bsp.:** System zum Stellen medizinischer Diagnosen, unterstützt bzw. ersetzt/imitiert Experten
  - ▷ **Software-Agenten**, **Bsp.:** Roboter
- **Wissensbasierte Techniken** sind:
  1. Einbeziehen des gesamten Wissens in die Entscheidungsfindung
  2. Imitation des Experten, Begründung der Problemlösungsstrategie(n)
- Teilaufgaben eines WBS:
  - ▷ **Wissenserwerb/-akquisition** (KAPITEL 2)
  - ▷ **Wissensrepräsentation:** Darstellung des internen Wissens (KAPITEL 4)
  - ▷ **Wissensmodellierung:** Suchverfahren (KAPITEL 3)
  - ▷ **Dialogsystem:** Inferenzmechanismus für die Lösung von Problemen mithilfe des gespeicherten Wissens (KAPITEL 5.2)



- Wissen in der Wissensbasis **deklarativ**, besteht aus **Faktenwissen** (Daten) & **Regelwissen**. **Bsp.:** Produktionsregeln: `if … else …`
- **Inferenzkomponente** (regelbas. Systeme):
  - ▷ wird aktiviert, wenn Benutzer eine neue Session im WBS startet.
  - ▷ Verwende **Fakten** und **Regeln** aus bestehendem Wissen zur **Schlussfolgerung** (***Inferenz***). **Bsp.:** Roboter hat Näherungssensor und erkennt Wand voraus (Fakt). Roboter hat eingespeichert, nicht gegen Wände zu fahren (Regel). Robotor wird anhalten (Schlussfolgerung).
- WBS braucht Benutzerschnittstelle (Eingabemöglichkeit, ggf. Dialogsystem (Spracheingabe), ggf. Erklärungskomponente, …)

2 Wissensakquisition

2.1 Wissensakquisition

- Während Anfänger analytisch & distanziert Entscheidungen treffen, können Experten diese aus Erfahrung intuitiv & involviert treffen (implizites Wissen) – dieses ist zu extrahieren.
- **Phasen der Wissensakquisition:**
  1. **Problemcharakterisierung:** Identifizierung der Problemlösungsstrategie & der Wissensbasis. Wissensingenieur richtet Daten, Probleme & Fragen an Experten, der Informationen, Wissensartefakte, Konzepte & Lösungen zurückgibt.
     Hilfsmittel: Interviewtechniken und Protokolle
  2. ***Shell*-Entwicklung:** Umsetzung der Charakterisierungen in ein Expertensystem-*Shell* durch Wissensingenieur, d. h. Umgebung, in der das Wissen akquiriert werden kann (**Bsp.:** Konsolen-/Desktop-Prog., bestehend oder neu entwickelt)
     Hilfsmittel: allgemeines Expertensystem-Werkzeug
  3. **Aufbau d. Wissensbasis:** Formalisierung d. Expertenwissens durch den Experten, evtl. unterstützt durch Wissensingenieur.
     Hilfsmittel: *Shell*
  4. **Wartung:** Verbesserung/Korrektur von Fehlern, Anpassungen an geänderte Anforderungen durch Experten, evtl. unterstützt durch autom. Analysetechniken.
     Hilfsmittel: Falldatenbank (**Bsp.:** *Case-based Reasoning* KI-/ML-Methoden)
- Starke Abhängigkeit von Wissensakquisition & -modellierung der Wissensbasis.
- Verschiedene potenzielle Wissensquellen:
  - ▷ Experten(gruppen) mit langwieriger Erfahrung zur intuitiven Entscheidungsfindung in seiner Tätigkeit. **Bsp.:** Pizzabäcker, der an Teigfarbe sieht, wie die Temperatur geregelt werden muss.
  - ▷ textuelle Informationen (Berichte, Artikel, Bücher, Gesetze, Richtlinien, . . .)
  - ▷ digitale Informationen (Aufnahmen) **Bsp.:** Röntgenbilder, *Deep Learning* zum Antrainieren verschiedener Situationen
  - ▷ Endbenutzerfeedback **Bsp.:** „Wie gut war diese Entscheidung?"

  Experten wissen auch nicht alles, daher möglichst viele Quellen nutzen!
- **Methoden der Wissenserhebung:**
  - ▷ **indirekt:**
    - Wissensingenieur befragt Experten, formalisiert die Ergebnisse für WBS
    - aufwändig & fehleranfällig
    - Einsetz von Interviewtechniken
  - ▷ **direkt:**
    - Experte formalisiert Wissen selbst.
    - erfordert komfortables Softwaresystem und nicht zu komplexen & gut verstandenen Wissensbereich
  - ▷ **semi-automatisch:**
    - Unterstützung des Experten bei Wissenskonzeptualisierung
    - Matrix-basierte Techniken (Fragebögen, Qualitätsfunktionendarst., . . .)
    - *Expertise Transfer Systems*, **Bsp.:** Software für Fragenkategorien, Dialoge zur Modifikation
  - ▷ **automatisch:**
    - System extrahiert selbst notwendiges Wissen aus Falldaten/Literatur
    - erfodert überschaubaren & sehr gut verstandenen Wissensbereich
    - enormes Potenzial in Lernfähigkeit
    - „Traum jedes Wissensingenieurs"; Rolle von Wissensingenieur & Experte minimiert oder sogar eliminiert
    - heutige, moderne Verfahren (z. B. KI)
    - induktive Verallgemeinerung (d. h. Schließen von Regel aus Einzelfall); Wissen kann fehlerhaft sein
- Manuelle Methoden (direkt/indirekt):
  - ▷ strukturierte & unstrukturierte Interviews
    Fragetypen:
    - direkte Fragen, festgelegte Antwortmöglichkeiten (Fragebogen)
    - indirekte Fragen, offene Fragenstellungen, Freitextantworten
    - primäre Fragest. (neue Themen)
    - sekundäre Fragestellungen (Herausfinden zusätzlicher Infos)
  - ▷ Fallanalyse
  - ▷ kritische Störfallanalyse
  - ▷ Diskussion mit Benutzer
  - ▷ Kommentare & Bewertungen
  - ▷ Konzeptgrafiken & -modelle
  - ▷ *Brainstorming*
  - ▷ *Prototyping*
  - ▷ multidimensionale Skalierung
  - ▷ Sortierungstechniken

  Rahmenbedingungen f. man. Wissenserwerb:
  - ▷ passende Umgebungssituation; Möglichkeiten zur Kommunikation (**Bsp.:** in Person: Raum, Tische, Stühle, . . .)
  - ▷ Ablenkungen vermeiden
  - ▷ Protokollierungsmöglichkeiten (Schreibgeräte), Aufnahmegeräte, . . .
  - ▷ Quellen für Hintergrundwissen, Modelle, Prototypen, Simulationsergebnisse, . . .
- **Probleme des Wissenserwerbs:**
  - ▷ schlechtes Management und mangelnde Organisation in Erwerbsphase (**Bsp.:** schlecht durchgeführte Interviews)
  - ▷ unvollständige oder schlechte Ausbildung des Wissensingenieurs
  - ▷ keine Möglichkeit der Übersetzung des Wissens aus der Befragung in verarbeitbaren Code/keine Ableitbarkeit v. Regeln
  - ▷ Experte hat eine Menge implizites Wissen, kann aber nur schwer extrah. werden
- Strategien zur Überwindung der Schwierigk.:
  - ▷ Software für Wissenserwerb entwickeln
  - ▷ Spracheingabe (z. B. für Schemata/einfache Regeln)

2.2 Systeme für Wissensakquisition

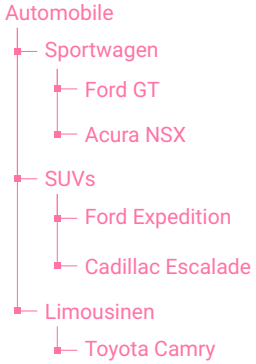
- **Wissenserwerbssysteme:**
  - ▷ betreffen den direkten Wissenserwerb, d. h. durch Experten selbst
  - ▷ sollen eine dem Experten bekannte, verständliche Darstellung der int. Wissensrepräsentation verwenden
  - ▷ komfortable Eingabe, sofortige Überprüfbarkeit von Änderungen, Konsistenztests
- **KADS**/***CommonKADS***: Standard/Leitlinie für Wissensakquisition (etwas in die Jahre gekommen, ca. 20 Jahre alt, eher als Modellierungsphilosophie zu verstehen)
- Heute: steigende Anzahl an Informationsquellen ⇒ automatische Analyseverfahren, lernende Systeme (*machine learning*), *Text Mining*

3 Wissensmodellierung

- Verwendungsszenarien für Modelle:
  - ▷ Werkzeug für Wissensakquisition
  - ▷ **Validierung** (mit dem Experten)
  - ▷ Kreuz-Validierung (mit anderen Exp.)
  - ▷ Formalisierung & Abbildung im Computer (modellbas. Wissensrepräsentation)
  - ▷ Darstellung des Wissens (**Informationsvisualisierung**)
- Verschiedene Arten der Modellierung:
  - ▷ **Geschäftsmodellierung** **Bsp.:** *BPM* (Geschäftsprozess-Analyse)
  - ▷ **Kommunikationsmodellierung** **Bsp.:** Schnittstellen zu Informationssystemen
  - ▷ **System-Design und -Implementierung**
  - ▷ **Konzeptionelle Modellierung** **Bsp.:** wissensintensive Aufgaben durch Modellierung bewältigen (i. F. unser Hauptthema)
- **Konzeptionelle Modellierungsphasen:**
  1. **Wissensidentifikation**
     - ▷ Wissensquellen aus Wissensakquisition heranziehen
     - ▷ gemeinsames „Vokabular" konstruieren, **Bsp.:** Glossar, Fachbegriff-*Wiki*
     - ▷ Schlüsselstellen/Entscheidungspunkte identifiz. (f. späteren Regelentwurf)
  2. **Wissensspezifikation**
     - ▷ Typen, Objekte & Klassen bilden
     - ▷ finde geeignetes Modellierungstool
     - ▷ Relationstypen festlegen, spezifiziere Relationen zw. Objekten, Klassen
     - ▷ def. Eigenschaften der Klassen
  3. **Wissensverfeinerung**
     - ▷ zusammengetragene Informationen nochmal ansehen
     - ▷ ggf. Rücksprache mit Experten
     - ▷ Validierung des Modells auf Fehler
     - ▷ Simulation & Test der Modelle anh. Szenarien aus Wissensakquisition

  ggf. iterativ wiederholen.
- Bestandteile der Wissensmodellierung:
  - ▷ **Konzepte:** übergeordnete Begriffe, Begriffssammlung, Objekte
  - ▷ **Instanzen:** Ausprägungen eines Objekts
  - ▷ **Prozesse:** Vorgangsbeschreibungen
  - ▷ **Attribute & Werte:** Objekt-Spezifikationen
  - ▷ **Regeln:** aus dem Kontext abgeleitete IF–THEN-Bedingungen
  - ▷ **Beziehungen/Relationen:** **Bsp.:** „ist-ein"-, „ist-ein-Teil-von"-Beziehungen
- **Techniken d. Wissensmodellierung (*Ladder*):**
  - ▷ ***Concept Ladder* (Taxonomie):**
    - zeigt alle Klassen von Objekten und ihre Untertypen
    - hat nur „ist-ein"-Beziehungen
    - kann in fast allen Bereichen zur Wissensrepräs. eingesetzt werden

    Bsp.:

▷ ***Composition Ladder*:**
- zeigt Bestandteile eines Objektes an
- hat nur „ist-ein-Teil-von“-Bez.
- zum Verstehen komplexer Objekte (**Bsp.:** Maschinen, Organisationen)

Bsp.:
- Automobile
  - Aufhängung
    - Gestänge
    - Federung
    - Querstabilisator
  - Bremsen
    - Bremsscheiben
    - Bremsbeläge
    - ABS
  - Antrieb
    - Motor
    - Getriebe
    - Achsen

▷ ***Decision Ladder*:**
- „Vorstufe von Regeln“
- zeigt Vor- und Nachteile für verschiedene Entscheidungsmöglichkeiten
- für detailliertes Prozesswissen

Bsp.:
- Automobil-Treibstoff
  - Bleifreies Benzin
    - ✓ allgemein
    - ✓ gute Leistung
    - × mittlerer Benzinverbrauch
  - Diesel
    - ✓ allgemein
    - ✓ preiswert
    - ✓ niedriger Benzinverbrauch
    - × geringe Leistung
    - × lauter Motor

▷ ***Attribute Ladder*:**
- zeigt Attribute und deren Werte
- relevante Adjektive für ein Attribut werden als Unterknoten angezeigt

Bsp.:
- Automobile
  - Kosten
    - teuer
    - günstig
    - exorbitant
  - Geschwindigkeit
    - schnell
    - 0-100 km/h in 2 s
    - 0-100 km/h in 9 s
  - Sicherheit
    - sicher
    - unsicher
    - abgeschirmt

▷ ***Process Ladder*:**
- zeigt Prozesse und deren Teilprozesse/Aufgaben/Aktivitäten
- hat nur „ist-ein-Teil-von“-Bez.
- muss nicht geordnet oder durch Maschine interpretierbar sein
- sinnvollste Variante, um Prozesswissen darzustellen

Bsp.:
- kleine Inspektion
  - Materialkauf
    - gehe zum Teillager
    - finde Material
  - Ölwechsel
    - hebe das Auto an
    - lasse das Öl ab
    - auffüllen mit neuem Öl
  - Zündkerzen austauschen
    - alte Zündkerzen entfernen
    - Elektrodenabstand einstellen
    - neue Zündkerzen einsetzen

▷ ***Concept Map*:**
- sehr freie Gestaltungsmöglichkeit
- Objekte als Knoten, Beziehungen als Verbindungen (Pfeile)
- ähnlich dem semantischen Netzwerk (dort haben Kanten jedoch semantische Bedeutungen und müssen von einer Maschine interpretierbar sein)
- Erstellung ist iteratives Verfahren, jeder Durchgang soll eine bessere Lösung liefern
- **Bsp.:** *PreSERVe* (Methode von NASA & US-Militär für Wettervorhersagen)
- **Bsp.:** *IHMC Map*, *CmapTool*: simples Tool zur Erstellung v. *Concept Maps*

4 Wissensrepräsentation

4.1 Wissensrepräsentation

- Arten der Wissensrepräsentation:
  - ▷ **deklarative Wissensrepräsentation:**
    - als Fakten gespeichertes Wissen
    - Wissen, das interpretiert werden muss (Inferenz)
    - ineffizient zur direkten Problemlösung (alle Fakten durchgehen)
  - ▷ **Prozedurale Wissensrepräsentation:**
    - Algorithmen (Programmcode)
    - nur nutzbar in speziellen Kontexten, aber dort hocheffizient
- Analog: Arten der Programmierung:
  - ▷ **prozedurale Programmierung:**
    Aufspalten von Programmen in Teilaufgaben (Prozeduren). Fokus auf „Wie?“.
    Bsp.: alle imperativen Programmiersprachen (C, C++, Java, …)
  - ▷ **deklarative Programmierung:**
    Beschreibung des Programms steht im Vordergrund (Fokus auf „Was?“). Lösungsweg wird automatisch ermittelt.
    Bsp.: Haskell, Lisp, Prolog
- **Wissensdarstellungssprache** (*knowledge representation language*, ***KRL***) zum Codieren von Wissen; Standard aus den 80ern/90ern
  Bsp.: ***GRAIL*** (*GALEM Representation & Interpretation Language*) ist **frame-basierte *KRL***, d. h. limitierte Aussagekraft, Fokus auf schnellen Antworten auf einfache Fragen
- **Methoden der Wissensrepräsentation:**
  - ▷ ***Frames*:**
    - Datenstruktur zur Darstellung einer stereotypischen Situation
    - verschiedene Art der Informationen beigefügt: Verwendung, nächster *Frame*, „Erwartungen bestätigt“?
    - Vergleiche mit UML: Netzwerk von Knoten & Beziehungen
    - hierarchische Vererbungsstruktur
    - Attribute & Beziehungen = ***Slots***
  - ▷ ***Scripts*:**
    - Situationen und beobachtete Ereignisse interpretieren; Herausfinden von Gründen für beobachtete Aktion; Planungsinstrument
    - Bestandteile:
      - Szene, Szenentyp (*track type*)
      - Requisiten (im Skript manipulierte Objekte)
      - Akteure (Agenten, die Zustand der Welt verändern können)
      - Ereignisse
      - Vorgänge: Aktionen der Akteure
    - Darstellung als Baum oder Netzwerk von Zuständen, die durch Ereignisse verändert werden
  - ▷ **Semantische Netzwerke** (Wissensnetz)**:**
    - formales Modell von Begriffen und ihren Beziehungen
    - Knoten des Graphen stellen Begriffe dar, Kanten die Beziehungen
    - ähnlich der *Concept Map*, aber mit maschinell interpretierbaren Bez.
  - ▷ **Prädikatenlogik:**
    - Erw. der Aussagenlogik um **Prädikate** (Begriffe, Eigenschaften, Relationen)
    - Einsatz: Wissen über Objekte, deren Eigenschaften und Beziehungen deklarativ beschreiben
    - Einfluss auf prozedurale Beschreibung des Programmcodes
    - es haben sich **Repräsentationssprachen** etabliert, die die **Beschreibungslogik** (*description logic*) nutzen. **Bsp.:** *Web Ontology Language (OWL)* mit *Resource Description Framework (RDF/RDFS)*
  - ▷ **(Produktions-)Regeln:**
    - Simulation des kognitiven Verhaltens der menschlichen Experten
    - Diagnose von IF – ELSE-Problemen
    - Wissensregeln (deklarative Regeln) **Bsp.:** WENN Angebot > Nachfrage DANN sinkt der Preis.
    - Inferenzregeln (prozedurale Regeln) **Bsp.:** WENN erfolderliche Daten nicht im System DANN fordere sie an.

4.2 Regelbasierte Systeme

- Aufbau:
  - ▷ **Faktenbasis/Arbeitsbereich**:vorliegende Fakten der Situation
  - ▷ Wissensbasis mit Regeln (**Regelbasis**)
  - ▷ **Kontrollsystem**: Regelauswahl und Konfliktresolution (Auswahl aus mehreren zutreffenden Regeln):
    - erste passende (übliche Variante)
    - zufällig

  - Metaregeln (Prioritäten, „am besten passende“, Vermeiden von wiederholten Regelanwendungen, …)
- wir sagen eine Regel „**feuert**“, wenn diese zutrifft & von Konfliktresolution ausgewählt wird
- Hauptaufgabe:
  - ▷ neue Fakten ableiten
  - ▷ feststellen, ob ein Sachverhalt mit vorgegebenen Regeln und bekannten Fakten abgeleitet werden kann
- Kontrollregime/Inferenzmechanismen: (Wichtig für Klausur!)
  - ▷ **Vorwärtsverkettung** (***forward chaining***): mit Fakten beginnen, anwendbare Regeln festlegen und auf die Fakten anwenden (datenorientiert)
    - **–** ***Recognize-Act Cycle*****:** Grundprinzip
      1. ***Match*****:** welche Regeln treffen zu?
      2. ***Select*****:** wähle nützlichste Regel(n) (inkl. Konfliktresolution)
      3. ***Execute*****:** Aktualisieren des Arbeitsbereiches mit Schlussfolg. d. ausgeführten Regeln als neue Fakten
    - **–** Wird verwendet, wenn es eine hohe Anzahl Ziele und relativ kleine Anzahl an möglichen „*Inputs*“ gibt.
  - ▷ **Rückwärtsverkettung** (***backward chaining***): gegebene Hypothese; nach Regeln suchen, die das Ziel zerlegen; kleinere Ziele lösen (zielorientiert)
    1. Wähle erstes Ziel der Hypothese.
    2. Ist das aktuell gewählte Ziel bereits als Fakt im Arbeitsbereich, gebe TRUE zurück. Sonst: Betrachte alle Regeln, deren Schlussfolgerung das gegebene Ziel ist. Gibt es keine passenden Regeln mehr, gebe FALSE zurück.
    3. Gehe passende Regeln nacheinander durch, wähle deren Fakt (Voraussetzung) als neues Ziel; wiederhole damit rekursiv Schritt 2. Ist das Ergebnis TRUE, so trifft das aktuell gewählte Ziel zu und ist als Fakt dem Arbeitsbereich hinzuzufügen.
    4. Treffen alle Ziele der Hypothese zu, so stimmt die Hypothese.

    – Wird verwendet, wenn es eine hohe Anzahl von möglichen „*Inputs*“ und relativ kleine Anzahl von Zielen gibt.
  - ▷ Hybridverkettung (nicht Teil der Vorlesung)

5 Experten- & Dialogsysteme

5.1 Expertensysteme

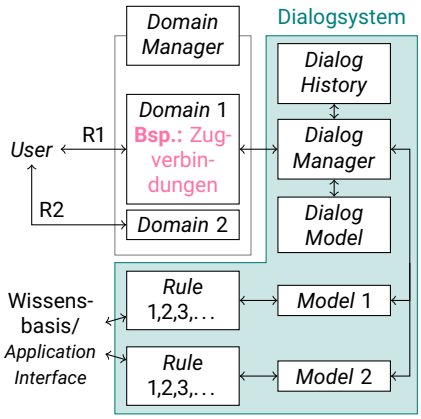
- Def.: **Expertensysteme** sind WBS, die einen bei der Lösung komplexerer Probleme wie ein Experte durch aus der Wissensbasis abgeleitete Schlussfolgerungen unterstützen.
- Es geht hier nicht darum, den Experten zu ersetzen, sondern zu unterstützen.
- Expertensysteme sind spezialisierte Form von regelbasierten Systemen.
- Ensteht durch:
  - ▷ direkt (Experte → Computersystem)
  - ▷ indirekt (Experte → Wissensingenieur)
  - ▷ Experte → Nicht-Experte (**Bsp.:** Kommunikationskanal in Expertenportal)
- **Hauptkomponenten** eines Expertensystems:
  - ▷ **Wissensbasis:** Langzeitspeicher
  - ▷ **Inferenzmaschine** (Problemlösungsstrategien)
  - ▷ **Benutzeroberfläche** **Bsp.:** Sprachverarbeitung, Menüs

  Andere Komponenten:
  - ▷ **Editor** für die Wissensbasis (ggf. *live*)
  - ▷ **Erklärungskomponente:** Auskunft für Benutzer über Lösungsfindung (→ KI)
  - ▷ **fallspezifische Daten:** Falldatenbank, historische Daten (u. a. für *Case-based Reasoning*)
- menschliche Elemente von Expertensystem:
  - ▷ **Experte**
  - ▷ **Benutzer** („Nicht-Experte“)
  - ▷ **Wissensingenieur**
  - ▷ Andere: Systementwickler, Entwickler von Expertensystemwerkzeugen
- **Entwicklungsphasen** von Expertensystemen: Es gibt hier viele Entwicklungsmodelle, hier nur exemplarisch, ist nicht immer so wie hier.

  Hier: klassisches Wasserfall-Modell.
  1. **Identifikation** dringender Parameter: Beteiligte, Probleme, Zielvorstellungen, Ressourcen, Kosten, Zeitrahmen
  2. Anforderungsanalyse: externe Voraussetzungen, Form von Input & Output, Umgebung, Zielgruppe
  3. **Konzeptualisierung:**
     - ▷ Initiierung der Wissensakquisition
     - ▷ Interaktion & Kommunikation zw. Experte & Wissensingenieur
     - ▷ Identifikation & Erhebung v. Wissen
     - ▷ Bestimmung von Schlüsselkonzepten, Beziehungen zwischen Objekten & Prozessen, Kontrollmechanismen
  4. **Formalisierung** und Modellierung:
     - ▷ Organisation der Schlüsselkonzepte, Teilprobleme & Informationsfluss
     - ▷ Modellierung und formale Repräsentation des Expertenwissens
     - ▷ Design der Programmlogik
  5. **Implementierung:**
     - ▷ formalisiertes Wissen in *Framework* des Entwicklungstools integrieren, in spezifische Sprache übersetzen
     - ▷ Dokumentation d. Expertensystems
     - ▷ weiterführende Fragestellung: Wie kann das Expertensystem mit anderen Programmen, Datenbanken & Wissensquellen interagieren?
  6. **Testen:** Verifizierung, Validierung der Inferenz & Performance, Evaluierung der Software: Erfüllung der definierten Anforderungen, Bewertung der Qualität
- Def.: **Informationssystem:**
  - ▷ System, das den Datenbedarf eines betrieblichen Systems verwaltet.
  - ▷ Datenerfassung und -verwaltung, um nützliche Informationen zu erhalten
  - ▷ führt Aufzeichnungen, pflegt Daten, die zur zur Führung des Unternehmens erforderlich sind
  - ▷ besteht aus Daten, Personen, Verfahren und Maschinen (Computer und andere)
  - ▷ verschiedene Funktionen:
    - **–** *Transaction Processing System* (*TPS*)
    - **–** *Management Information Sys.* (*MIS*)
    - **–** *Decision Support System* (*DSS*)
    - **–** *Executive Information System* (*EIS*)
    - **–** **Expertensysteme** sind eine Art von Informationssystem. (Nicht andersrum!)
    - **–** Kommunikations-, Kollaborationssys.
    - **–** Büroautomatisierungssysteme

5.2 Dialogsysteme

- Def.: **Dialogsystem** ist Hauptkommunikationsschnittstelle zwischen Benutzer und WBS.
- Dialogsystem nutzt Informationen aus einer Wissensbasis zur Kommunikation mit dem Benutzer anhand von (Dialog-)Modellen.
- Nicht jedes WBS hat ein Dialogsystem.
- **Dialogmodelle** bilden einfache Frage-Antwort-Dialoge ab. Auch komplexere Eingabe-/Ausgabe-Strukturen (z. B. Fragestellungen) mgl.
- für effektive Kommunikation: Verständnis erleichtern und durch Anpassung Kommunikationsproblemen begegnen (z. B. Rückfragen)
- vielfältige Einsatzszenarien (**Bsp.:** Abfrage von Flug-, Bahn-, Wetterinformationen, Navigationssysteme, *Call Centers*, …)
- **klassische Struktur eines Dialogsystems**:



  - ▷ **Domäne/*Domain*:** „Vorkategorisierung“ des Dialogsystems
  - ▷ ***Dialog Manager*** kann zwischen verschiedenen **Modellen** zur Verarbeitung wählen, je nach Fragestellung; ermöglicht, Regeln über das Modell dann zu benutzen.
  - ▷ ***Dialog History*****:** zum Lernen aus vergangenen Konversationen (wenn Nutzer zufrieden war mit einer Antwort)
  - ▷ *Application Interface* zum Zuhilfenehmen externer Quellen, wenn Regeln zum Verstehen des Benutzers nicht ausreichen.

  Bsp.: *MALIN* (modulares Dialogsystem, 2000)
- Die meisten Dialogsysteme haben eine Komponente zur Verarbeitung der Spracheingabe und nutzen Methoden des ***Natural Language Processing*** (*NLP*) bis hin zur Untersuchung der Grammatik (Wortarten, Satzbau).
- **Dialogsysteme heute:** vielfälgie Zwecke

  Bsp.: *Amazon Alexa*, *Siri* (*Apple*), *Google Assistant*, *Cortana* (*Microsoft*)

6 Semantische Wissensbasis

6.1 Semantische Grundlagen

- Def.: **Semiotik:** „Lehre der Zeichen, Zeichensysteme und Zeichenprozesse“
  - ▷ **Syntax:** (unterste Ebene) Definition aller zulässigen Zeichen(folgen) im Kontext
  - ▷ **Semantik:** Bedeutungslehre; Sinn & Bedeutung von Sprache/sprachl. Zeichen
  - ▷ **Pragmatik:** Bedeutung für den Benutzer (Einbettung in den Kontext und Intention)
- Arten der semantischen Wissensrepr., aufsteigend nach semantischer Reichhaltigkeit:
  1. **Glossar** (simple Begriffserklärungen)
  2. ***Folksonomy*****:** Verschlagwortung der Begriffserklärung zum besseren Zuordnen
  3. **Taxonomie:** hierarchische Klassifikation mit Beschreibungen
  4. **Thesaurus:** Beziehungen zwischen Begriffen (**Bsp.:** Synonyme, Antonyme, …)
  5. ***Topic Map*****:** Wissenslandkarten; Begriffe noch weiter erläutern, annotierte Beziehungen (wie?, warum?), …

6. **Ontologie:** Abb. eines geschlossenen „Weltausschnitts“, d. h. ein bestimmtes Thema so modellieren, dass man Beziehungen in IT-System abbilden kann

- **Semantische Mehrdeutigkeit:**
  - ▷ **Lexikalische Mehrdeutigkeit:** Synonyme, Homonyme, …; **Bsp.:** die Bank (🪑 oder $)
  - ▷ **Strukturelle Mehrdeutigkeit:** nicht eindeutige Struktur eines Satzes/einer Aussage, **Bsp.:** die Bank vor der Bank (?)
- **Semantische Konflikte**, die die Interoperabilität verschiedener Systeme beeinträchtigen:

  Bilaterale Konflikte:
  - ▷ **Integritätskonflikte:** verschiedene *Identifier* für das gleiche Objekt
  - ▷ **Datentypkonflikte:** verwendung unterschiedl. Datentypen für gleiche Werte
  - ▷ **Namenskonflikte:** Verwendung unterschiedl. Namen für die gleichen Objekte

  Multilaterale Konflikte:
  - ▷ **Attributskorrespondenzen:** Zuordung von einer vs. Zuordnung mehrerer Ressourcen zu einem Attribut
  - ▷ **Entitätskorrespondenzen:** unterschiedliche Darstellung einer Entität
  - ▷ **Fehlende Werte:** Pflichtangaben zu einem Sachverhalt fehlen in einer Quelle

  Meta-*Level*-Konflikte: Es werden unterschiedliche Meta-Modelle verwendet, unterschiedliche Datenstrukturierung & -ablage.

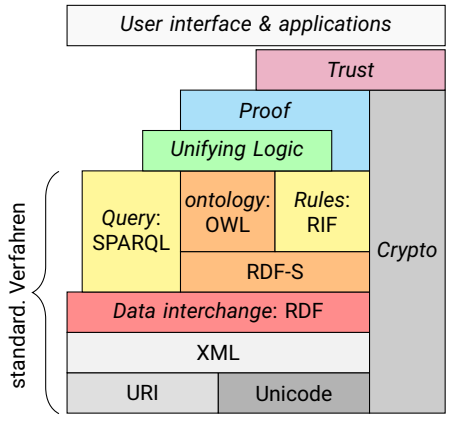
  Datenkonflikte:
  - ▷ **Verschiedene Einheiten**, **Bsp.:** Maßeinheiten (km ↔ mi), Währungen ($ ↔ €), …
  - ▷ **Verschiedene Wertebereiche**, **Bsp.:** Schulnoten in DE (1-6) vs. in CH (6-1)
  - ▷ **Subjektives *Mapping***, **Bsp.:** Luxushotel entspricht 4 oder 5 Sterne

  Domain-Konflikte:
  - ▷ **Unterordnung**
  - ▷ **Überlappung**, **Bsp.:** Hotel und Hostel (günstiges Hotel?)
  - ▷ **Inkonsistenz:** Unklarheit über den Ausschluss von Überlappungen
  - ▷ **Aggregation:** unterschiedliche Abstraktionslevel; unterschiedl. Zusammenfassung von ähnlichen Sachverhalten (**Bsp.:** „Berlin liegt in DE“ vs. „Berlin liegt in EU“)

  Lösungen semantischer Konflikte:
  - ▷ Ansatz A: logische Abbildung der konzeptuellen Struktur der Datenbank + automatisches *Reasoning* (**Bsp.:** Algorithmus zum Umrechnen von km in mi)
  - ▷ Ansatz B: logisches Modell angelehnt an konzeptionelle Struktur, aber durch zusätzliche Definitionen anreichern (**Bsp.:** speichere Entfernung in km und in mi)

6.2 Wissensrepräsentation mittels *Semantic-Web*-Technologien

- 2000: 1. *Semantic Web Stack* (*SWS*) von W3



- ständig in der Weiterentwicklung; Teile sind Realität, andere Teile noch Forschungsgebiet
- Struktur der Formate:
  - ▷ **XML:** Baumstruktur
  - ▷ **RDF:** Graph (Beziehungen der Objekte als Tripel „Subjekt, Prädikat, Objekt“)
  - ▷ **Ontologien:** „abgeschlossene Welt“

6.3 Semantische Suche

- **Bsp.:** Bei Eingabe von „Wetter in Siegen“ in *Google* erhält man eine Wettervorhersage.
- Ausgangssituation & Zielsetzung:
  implizites *Reasoning*: Wie können wir die Suchanfrage interpretieren, und wie können wir die Suchergebnisse liefern, die dem mentalen Modell des Benutzers entsprechen?
- Semantische Technologien interpr. & beschreiben Wissen formal & maschineninterpretierbar (statt „dummer“ Vergleich von Zeichenketten wie z. B. bei reiner Schlüsselwortsuche).
- Eigenschaften der semantischen Suche:
  - ▷ Bedeutung der Suchworte/-phrasen heranziehen; Bedeutung der Suchanfrage im Dialog mit Benutzer/Kontext
  - ▷ anhand d. Ergebnisse die Suchworte konkretisieren, Ergebnismenge erneut durchsuchen lassen
  - ▷ Suche nach Dokumenten
  - ▷ Ziel: Verbesserung des *Rankings* erzielen (passendste Treffer zuerst)
  - ▷ Interpretation der Suchanfragen anhand semantischer Annotationen (Zusatzinfo)
  - ▷ Bereitstellung der maschineninterpretierbaren Informationen zur Bedeutung von Termen, Ausdrücken, Dokumenten
  - ▷ Realisierung höherer Abstraktionsebene (dem mentalen/abstrakten Modell des suchenden Benutzers annähern)
- Es gibt hybride Modelle (Schlüsselwortsuche + semantische Suche).
- Optionen der semantischen Suche:
  1. Verwenden einer **semantischen Anfragesprache** (z. B. SPARQL für RDF).
     - ▷ Problem: „Normalanwender“ haben keine Erfahrung in der Formulierung von semantisch formalen Anfragen.
  2. Eingabe von **Freitext**
     - ▷ Anfrage muss zuerst semantisch interpretiert & formalisiert werden.
  3. Hybrid: **formularbasierte Suche**
     - ▷ Vorstrukturierung der Eingabe durch Web-Formulare
     - ▷ Nutzung von Wertebereichen und Metadaten
- SPARQL ist vergleichbar mit SQL.
- Erweiterung: ***Semantic Web Rule Language*** als Teil des SWS; zum Verwenden von semantischem Wissen für logische Abfragen (traditionelle Techniken wie *Forward Chaining*, etc.). Wird aktuell noch ausgearbeitet.
- Ebenen der semantischen Techniken/Probleme:
  - ▷ *GUI*-Probleme/Anfragen
    Bsp.: Abfragesprachen (SPARQL), Freitext-Suche, formularbasierte Suche
  - ▷ *GUI/Frontend* ↔ *Backend*:
    - – ***Faceted Browsing*:**
      - · Kombination aus Suche & *Browsing*
      - · Benutzer kann Suche eingeben und dann anhand von „Facetten“ filtern
      - · Nutzung von Metadaten der Dokumente als Facetten
      - · neben Suchergebnissen auch zugehörige Facetten als Ergebnis
      - · kontinuierliche Einschränkung des Suchraums möglich
      - · Weiterentwicklung der formularbasierten Suche
    - – **semantische Nachverarbeitung:** wie Schlüsselwortsuche + Extraktion & Strukturierung des Wissens aus den ermittelten Inhalten
    - – **intelligente Visualisierung:** wie Schlüsselwortsuche + intelligente Aufbereitung durch Visualisierung, **Bsp.:** Graphen, Diagramme, …
  - ▷ *Backend* ↔ Daten
    - – **semantische Schlüsselwortsuche**
      - · nutzt neben Schlüsselworten verfügbare semantische Daten
      - · keine Anfragesprache notwendig, meist natürlich-sprachige Eingabe
      - · keine Kenntnis über Aufbau der Wissensbasis nötig
      - · Ablauf: Bestimmung der zur Abfrage gehörende Konzepte, Ermittlung der damit verwandten Instanzen
    - – ***Question-Answering*:**
      - · Teilgebiet des *Natural Language Processing*
      - · Beantwortung von natürlich-sprachigen Fragen auf Basis hinterlegter semantischer Informationen
      - · Eingabe → intern formale Anfrage → Ergebnis → natürlich-spr. Ausgabe

    Bsp.: *WolframAlpha*, *Answer The Public*

7 Ontologie-Entwicklung

7.1 Ontologie

- abgeschlossener Weltausschnitt, Konzepte & Beziehungen darin
- verschiedene Def.: Eine **Ontologie**…
  - ▷ definiert die Grundbegriffe & Beziehungen, die das Vokabular eines Themenbereichs umfassen, sowie die Regeln für die Kombination von Begriffen & Beziehungen, um Erweiterungen des Vokabulars zu definieren. (*AI Magazine, Winter 1991*)
  - ▷ ist eine explizite Spezifikation einer (gemeinsamen) Konzeptualisierung. (*Knowledge Acquisition, Vol. 5, 1993*)
  - ▷ bietet die Möglichkeit, die Konzeptualisierung hinter dem in einer Wissensbasis dargestelltem Wissen explizit zu beschreiben. (*ECAI96*)
- Wichtig: Ontologie ist maschinell lesbar.
- Einsatz und Nutzen der Ontologie:
  - ▷ Strukturierung/Organisierung von Metadaten informeller Wissensquellen
  - ▷ Informationsbeschaffung in abgeschlossenen Weltausschnitten
  - ▷ Integration existierenden Wissens (ggf. sogar automatisch)
  - ▷ Suche in bestehendem Wissen
  - ▷ *Retrieval*: Abfrage von Wissen in geordneten Strukturen
  - ▷ Personalisierung: flexibler/bedarfsorientierter Zugang zum Wissen
  - ▷ Visualisierung: Darstellung von Abhängigkeiten, Beziehungen zw. Konzepten
- Ontologien definieren gemeinsames Vokabular, Bedeutung von Begriffen, Erklärungen der Zusammenhänge zwischen Begriffen
- wichtige Entwicklungsschritte einer Ontologie sind Konzeption & Umsetzung
- Ziele und Aufgaben: Eine Ontologie sollte…
  - ▷ allgemein sein („so generell wie möglich,

aber so spezifisch wie nötig“ – je allgemeiner eine Ontologie, desto mehr Leute und Informationen können sie benutzen)

- ▷ von bestehender Technologie unabhängig sein
- ▷ jeden Prozess/jede Aktivität definieren (Zweck, *Input*, *Output*; Sammlung von Methoden, Techniken und Werkzeugen)
- ▷ Integration & Anpassung von Zusammenhängen durch Entwickler und Praktiker erleichtern (d. h. tatsächliches praktisches *Tool*, nicht nur abstraktes Modell)

7.2 Methoden zur Ontologie-Entwicklung

- **Bsp.: *METHONTOLOGY*:**
  - ▷ **Management-Aktivitäten:**
    - **Terminierungsaktivität** identifiziert auszuführende Aufgaben, deren Anordnung, Zeit- & Ressourcenbedarf
    - **Kontrolltätigkeit** garantiert Erfüllung der Aufgaben wie vorgesehen
    - **Qualitätssicherungsmaßnahme** überprüft Qualität der einzelnen Methodenergebnisse (Ontologie, Software & Dokumentation)
  - ▷ **Entwicklungsorientierte Aktivitäten:**
    - Während **Vorentwicklung** wird das Umfeld untersucht; es gibt eine Machbarkeitsstudie.
    - Während **Entwicklung** gibt Spezifikationsaktivität an, warum die Ontologie aufgebaut wird, welche Anwendungen beabsichtigt sind, welche Endbenutzer existeren. Die **Konzeptualisierungsaktivität** strukturiert das Domänenwissen als aussagekräftige Modelle auf Wissensebene. Die **Formalisierung** wandelt das konzeptionelle Modell in ein formales Modell um. Berechenbare Modelle werden in **Implementierung** erstellt.
    - Während ***Post*-Entwicklung** aktualisiert und korrigiert **Wartungsaktivität** bei Bedarf die Ontologie (Gegenstand kann sich weiterentwickeln) und kann v. anderen Ontologien/Anwend. wiederverwendet werden.
  - ▷ **Unterstützungsaktivitäten (*Support*):**
    - **Wissensakquisitionstätigkeit:** Wissen von Experten oder durch (halbautomatisches) Lernen erwerben
    - **Bewertungsaktivität**, die die entwickelten Ontologien, Software und Dokumentation anhand eines Bezugsrahmens bewertet
    - **Integrationsaktivität**, wenn andere Ontologien wiederverwendet werden; mglw. im Zshg. mit **Fusions-** (erstellt neue Ontologie) oder **Ausrichtungsaktivitäten** (erstellt *Mappings*, die ursprüngliche Ontologie enthalten), wenn mehrere Ontologien wiederverwendet & kombiniert werden müssen.
    - **Dokumentation** beschreibt jede abgeschlossene Stufe und dokumentiert Ontologien, Software- und Dokumentationsversionen (*VCS*)

  Unterschiedliche Auslastung bei Aktivitäten:
  - ▷ Die Management- & Unterstützungsaktivitäten erfolgen zeitgleich mit den Entwicklungsaktivitäten.
  - ▷ Aufwand für Support-Aktivitäten ist entlang des Lebenszyklus nicht einheitlich.
  - ▷ Wissenserwerb, -integration & -bewertung sind bei der ontologischen Konzeption größer, da die meisten Erkenntnisse zu Beginn erworben werden und das Konzept so früh wie möglich bewertet werden sollte, um **Propagationsfehler** zu vermeiden.

  Evaluationskriterien:
  - ▷ **Verifikation:** Ontologie richtig aufbauen
  - ▷ **Validierung:** Aufbau einer korrekten Ontologie
  - ▷ **Bewertung:** Beurteilung aus Anwendersicht – ergibt die Ontologie in der Anwendung Sinn?
  - ▷ **Konsistenz:** kein Widerspruch in den Daten
  - ▷ **Prägnanz:** keine unnötigen Definitionen
  - ▷ **Vollständigkeit:** alle Informationen des Weltausschnitts sind enthalten
- **Bsp.: *On-To-Knowledge*:** Wasserfall-Modell
  1. Machbarkeitsstudie
  2. *Kick-off* (semi-formale Beschreibung)
  3. Verfeinerung (Formalisierung)
  4. Bewertung
  5. Anwendung & *Evolution* (Wartung)
- **Bsp.: *DILIGENT*:**
  1. ***Build*:** kleines Team baut Ontologie auf, die nicht vollständig sein muss.
  2. **Lokale Anpassung:** Benutzer passen Ontologie an ihre eigenen Bedürfnisse in ihrem lokalen Umfeld an.
  3. **Analyse:** auswählen, welche Änderungen zur nächsten Version d. Ontologie gehen
  4. **Revision** wird regelmäßig durchgeführt, um zu vermeiden, dass die Ontologie zu weit von lokalen Anpassungen abweicht.
  5. **Lokale *Updates*** können durchgeführt werden, wenn die Benutzer ihre lokalen Ontologien an die neue Version der Ontologie anpassen möchten.
- In Klausur: genaue Methoden zu kennen nicht zwingend gefordert, aber beispielhaft eine angeben zu können kann gefragt sein.
- ***Ontological Resource Reengineering Model*:** *Reverse Engineering* einer existierenden Ontologie; Ablesen der Konzeption, Formalisierung, Spezifikation; Verwendung dieses Wissens zur Implementierung einer anderen Ontologie.
- Herausforderungen für Ontologien in WBS:
  - ▷ Oft landet ein entwickelter Weltausschnitt am Ende in der Schublade, da es nicht allgemein genug ist oder das Thema sich erledigt hat.
  - ▷ langlebige Systeme in dynamischen Umgebungen, daher müssen ontologische Systeme **kostengünstig** entwickelt werden, um Verschlechterung der Systemleistung zu vermeiden.
  - ▷ **Inferenz:** es müssen leistungsfähige Argumentationsmechanismen und aussagenkräftige Domainbeschreibungen entwickelt werden

7.3 Aufbau einer Ontologie

- Es gibt verschiedene Varianten:
  - ▷ *Top-Level Ontology*
  - ▷ *Domain Ontology*
  - ▷ *Application Ontology*
- Wesentl. Bestandteile in der Domäne:
  - ▷ Definitionen von **Konzepten** in der Domäne (**Klassen/*Classes***)
  - ▷ **Attribute** und **Eigenschaften** (***Relations***) von Klassen und die Einschränkungen von deren Werten
  - ▷ Definitionen von **Personen** (***Individuals***) und Ausfüllen von *Slot*-Werten
- Ontologie muss nicht zwangsläufig alle mgl. Informationen über die Domain und alle Eigenschaften einer Klasse enthalten.

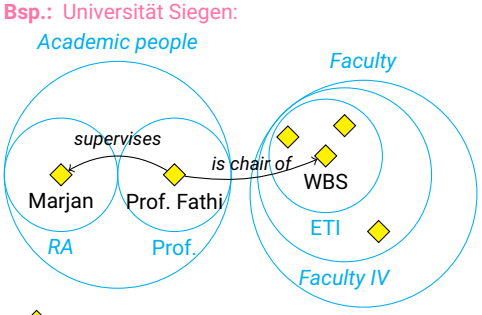
  Ontologien sind erweiterbar.
- Keine Notwendigkeit, sich weiter zu spezialisieren oder weiter zu verallgemeinern, als im Anwendungsfall nötig ist.
- **Klassen:**
  - ▷ typische Kandidaten: **Substantive**
  - ▷ Aber: Akteure von *Use Cases* entsprechen nicht unbedingt den Klassen.
  - ▷ Man definiert Klassen durch Interview mit Experten, Dokumentation, Beobachtung & Reflexion.

  Klassenhierarchie:
  - ▷ Synonyme für das gleiche Konzept stehen nicht für unterschiedliche Klassen.
  - ▷ Alle Geschwisterklassen müssen auf der gleichen **Ebene der Allgemeinheit** sein.
  - ▷ Eine **Unterklasse** stellt eine „Art“ des Konzepts der **Superklasse/Oberklasse** dar (Spezialisierung).
  - ▷ Wenn eine Klasse nur 1 Unterklasse hat, kann es zu einem Modellierungsproblem kommen (oder Ontologie unvollständig).
  - ▷ Zu viele („mehr als ein Dutzend“) Unterklassen einer best. Klasse können zusätzliche Zwischenkategorien erfordern.
  - ▷ Klassen können gemeinsame Instanzen haben (**überlappende Klassen**). Ansonsten heißen sie **disjunkte Klassen**.

    Überlappende Klassen können gemeinsame Unterklassen haben.
  - ▷ **Klassenzyklen** (zyklische Vererbung) vermeiden (sonst gelten alle beteiligten Klassen als gleichwertig).
- **Beziehungen** (*Relations*):
  - ▷ typische Kandidaten: **Verben**; verbale Phrasen; Dinge, die Verben sein könnten
  - ▷ $A \rightarrow B$, dann heißt Klasse von A **Domäne** und Klasse von B **Bereich** (***Range***)

  Beziehungscharakteristiken (wichtig!)
  - ▷ **Invers:** Wenn Relation p die umgekehrte Relation q hat, dann $ApB \Rightarrow BqA$.
    Bsp.: p = „stellt an“, q = „angestellt von“
  - ▷ **Funktional:** Für bestimmte Person nimmt Beziehung nur einen Wert an (**Bsp.:** Besitz, Aufenthaltsort)
  - ▷ **Symmetrisch:** $ApB \Rightarrow BpA$
    Bsp.: „arbeitet zusammen mit“
  - ▷ **Transitiv:** $ApB \wedge BpC \Rightarrow ApC$
    Bsp.: „ist Oberklasse von“
  - ▷ **Reflexiv:** mit sich selbst verknüpft
  - ▷ **Teil-von:** Beziehung zwischen Bestandteilen; Existenzabhängigkeit: Bestandteil ist obligatorisch für Existenz anderer Bestandteile
- **Instanz** einer Klasse ist einzelne „Version“ einer Klasse mit ausgefüllten Werten der Eigenschaften & Beziehungen.

Bsp.: Universität Siegen:



◆: Instanz
○: Klasse
↗: Beziehung

in dieser Form in Klausur!

- **Bsp.:** ***Protégé:*** *Open-Source*-Editor zur Ontologie-Entwicklung

8 Wissensextraktion aus Text

- Def.: ***Text Mining*:** Anwendung von statischen Analysen auf Textdaten. (∃ viele andere Def.)
- *Text Mining* ist Überschneidung von
  - ▷ ***Data Mining*:**
    - iterative Anwendung statistischer Analysemethoden
    - Ziel: neue Muster, (Un-)Regelmäßigkeiten in Datenbeständen erkennen
  - ▷ *AI* & *Machine Learning*
  - ▷ *Computational Linguistics*
  - ▷ Statistik/Stochastik
  - ▷ *Library-* & *Information Sciences*
- Im Gegensatz zu *Data Mining*: Beim *Text Mining* unstrukturierte Datenbasis in Textform, die vorverarbeitet werden muss (→ *NLP*)
- *Text-Mining*-**Prozess:**
  1. **Aufgabendefinition:** Problemstellung analysieren, Ziele ableiten, Zusammentragen relevanter Dokumente → **Korpus**
  2. **Dokumentenselektion:** Anhand Zielformulierung relevante Dokumente im Korpus identifizieren, verfügbar machen für Dokumentenaufbereitung
  3. **Dokumentenaufbereitung:** mit *NLP* einzelne Bestandteile des Textes extrahieren → Datenbasis aus Wörtern, Wortstämmen, zusammenges. Wortphrasen
  4. ***Text-Mining*-Methoden:** aufbereitete Form mit statistischen Verfahren untersuchen. **Bsp.:** Häufigkeitsanalysen, Segmentierung, Klassifikation, Ähnlichkeitsanalyse, …
  5. **Interpretation & Evaluation** (Bewertung) der Ergebnisse des *Text Mining*
  6. **Anwendung der Ergebnisse**
- ***Natural Language Processing* (*NLP*)**
  - ▷ Feld der Informatik, Computerlinguistik, künstlichen Intelligenz
  - ▷ Texte in Darstellung umwandeln, die eine maschinelle Verarbeitung, Analyse & Interpretation durch Computer erlauben
  - ▷ **morphologische Analyse:**
    - **Tokenisierung:** Texte in einzelne Bestandteile trennen (häufig *Whitespace* (Wörter), Sätze, Absätze)
    - bei gängigen Tokenisierungs-Verfahren werden numerische Zeichen, Sonderzeichen & Interpunktion entfernt
    - **Entf. von Stoppwörtern:** *Tokens* reduzieren um häufig genannte Begriffe, die keine Bedeutung haben (Stoppfwörter). **Bsp.:** in DE: Artikel, Konjunktionen, Präpositionen, teilw. gehen auch Negationen verloren!
    - **Lemmatisierung:** lexikographische Reduktion der Flexionsformen eines Wortes auf Grundform. (in komplexen Sprachen (**Bsp.:** DE) sehr schwierig, nicht einsetzbar)
  - ▷ **syntaktische Analyse:**
    - ***Part-of-Speech* (*POS*) *Tagging*:** versch. Satzteile mit ihren Wortarten auszeichnen durch Lexika (für Wörter & Wortarten) & syntagmatische Informationen
    - Häufige Technik: Analyse der Sequenz von Satzbestandteilen durch überwachtes Lernen mit *Hidden Markov Models*, Entscheidungsbäumen
    - *Tagsets*
      Bsp.: DE: Stuttgart-Tübingen-*Tagset* (STTS), EN: *Penn Treebank Tagset*
    - überwachtes Lernen: *Tagset* steht nicht vorher fest, sondern entsteht durch stochastische Verfahren
    - ***Phrase Recognition*:** untersucht den Text nach Wortgruppen/Phrasen
    - ***Named Entity Recognition*:** Finden von Eigennamen, Personen, Datum, Währung, Symbolen, Titel, Unternehmensbezeichnungen; Verifiz. durch *Linked Open Data* (Wikipedia, dbpedia)
    - ***Parsing*:** Satzbauanalyse; Annotieren jedes *Tokens* anhand seiner Stellung im Satz
    - ***Shallow Parsing*/*Chunking*:** Wörter zu Gruppen zusammenfassen, die als Satzbestandteil die Satzstruktur erläutern (*NP: noun phrase, VP: verb phrase, ADJP: adjective phrase, ADVP: adverb phrase, PP: prepositional phrase*)
    - ***Constituency Parsing*:** Komponenten-basierte Grammatik verwenden, um interne Struktur von Sätzen in hierarchisch geordneter Struktur ihrer Bestandteile zu modellieren
    - ***Dependency Parsing*:** mit abhängigkeitsbasierten Grammatiken strukturelle & semantische Abhängigkeiten/Beziehungen zw. *Tokens* in einem Satz analysieren/ableiten
  - ▷ **semantische Analyse:**
    - ***Word Sense Disambiguation*:** Kontext eines Wortes erfassen, um dessen Bedeutung/Absicht zu ermitteln (nicht vollst. gelöstes Problem)
- **Integratives *Text Mining*:**
  - ▷ Anwendung, Ergänzung, Kombination von individuellen *Text-Mining*-Methoden unter Verwendung einer Wissensbasis
  - ▷ mehrdimensionale Textanalyse: komplexe Aufgaben & Fragestellungen aus versch. Blickwinkeln betrachten
  - ▷ Bsp. für Dimensionen der Analyse: Entitäten, Stimmungen, Themen, Fakten, …
- **Sentimentanalyse:**
  - ▷ Bewertung eines Textes nach Stimmung, subjektiven Informationen
  - ▷ Polarität auf Skala -1 (negativ) bis 1 (positiv) **Bsp.:** SentiWS (Uni Leipzig)
  - ▷ Bew. auf unterschiedl. Ebenen wie Dokument, Satz, Entität, Assoziation, Aspekt
  - ▷ Zuordn. v. Textinhalten zu Emotionskateg.: Freude, Traurigkeit, Angst, Wut, Vertrauen, Ekel, Überraschung, Antizipation
- ***Topic Detection*:**
  - ▷ Erkennung von Haupt-/Unterthema in einem Text
  - ▷ unterscheide Ansätze der **Klassifikation** (Einordnung d. Textes in Kategorie) und ***Multi-Topic Detection*** (Zuordnung von Textbestandteilen zu versch. Kategorien)
  - ▷ benötigt Merkmale: *Tokens*, Metadaten
  - ▷ ***Term-Frequency Inverse Document Frequency* (*TF-IDF*):** Wie gut charakterisiert ein vorliegender Term ein Gesamtdokument im Vgl. zu allen Dokumenten im Korpus? (anh. Vorkommenshäufigkeit)

    Inverse Dokumentenhäufigkeit: Erfassung aller Dokumente, die das *Token* enthalten.

    (Termfrequenz × Dokumentenfrequenz) $\log$-normalisiert = *TF-IDF*
- **Kookkurenz:** gemeinsames Vorkommen von Wortpaaren (oder *Token-*/n-Gram-Paare) in einer vorgegebenen Textgröße (**Bsp.:** Textfenster von 5 Worten, Satz, Gesamtdokument). (**Wortassoziation**)
  - ▷ **Satzkookkurenz:** Wörter kommen im Satz gemeinsam vor. Kann für Modellierung semantischer Beziehungen genutzt werden.
  - ▷ **Nachbarschaftskookkurenz:** Wörter kommen unmittelbar in direkter Nähe zueinander vor. Kann zum Auto-Vervollständigen eines Suchbegriffs genutzt werden.
- **Koollokation:** signifikante (semantische) Abhängigkeit der Begriffe einer Kookkurenz
- Semantische Informationen: Bei der Extraktion & Nutzung von semant. Zusammenhängen unterscheiden wir i. F. 2 Ansätze:
  - ▷ **Assoziationsanalysen** mithilfe von Kookkurenzen
    - **Bsp.:** *CIMAWA* (*Concept for the Imitation of the Mental Ability of Word Association*): hybrider Ansatz, der symmetrische & asymmetrische Merkmale der menschl. Wortassoziation modelliert („Woran denken Sie, wenn Sie ‚Brot' hören?")
  - ▷ Extraktion von semant. Zusammenhängen in der Form **Subjekt-Prädikat-Objekt** (**Faktenextraktion**) **Bsp.:** „Essen ist eine schöne Stadt im (Präd.) Ruhrgebiet"

9 KI in WBS

9.1 Künstliche Intelligenz

- **Turing-Test**: Testverfahren, um der Frage nachzugehen, ob eine Maschine das Denken eines Menschen nachahmen kann.
  1. Es wird ein Gespräch simuliert, Mensch bedient einen Computer.
  2. Mensch soll nun mit zwei Probanden über Chat-Terminal kommunizieren. Ein Proband ist ein Mensch, der andere eine Maschine.
  3. Anwender kommuniziert über Chat-Terminal, stellt Fragen & erhält Antworten.
  4. Sollte der Anwender am Ende des Gesprächs nicht in der Lage sein, zwischen Mensch und Maschine zu unterscheiden, so geht die Wissenschaft davon aus, dass das Gerät dem menschlichen Denken ebenbürtig ist.
  5. Wenn es dem Computer in >30% einer Serie kurzer Unterhaltungen gelingt, seine menschlichen Gesprächspartner zu überlisten, gilt der Test als bestanden.

  Bsp.: Erster bestandener Turing-Test: Chatbot „*Eugene Goostman*", 2014
- Def.: **künstliche Intelligenz** (KI)
  - ▷ „ist die Lehre davon, wie Computer Dinge tun können, in denen Menschen besser sind – wenigstens im Moment noch." – Elaine Rich, 1983
  - ▷ Automatisierung intelligenten Verhaltens
  - ▷ Nachbildung von menschenähnlichen Entscheidungsstrukturen in einem nicht-eindeutigen Umfeld
  - ▷ Simulation von „intelligentem Verhalten"
- Wissenschaftliche Zielrichtungen von KI:
  - ▷ **Ingenieurswissenschaften:** Entwicklung von Methoden zur maschinellen Lösung von Problemen (u. a. Wissensrepräsentation, Inferenz, Suchalgorithmen, Klassifikation, Lernen)
  - ▷ **Wissenschaftliche Erkenntnisse:** Objektivierung und Modellierung von kognitiven Prozessen (u. a. Wahrnehmung,

Denken, Schlussfolgern, Sprachverstehen, Wissenserwerb)

  - ▷ **Formalwissenschaftlich:** (Weiter-)Entwicklung von Formalismen zur Beschreibung und Bewertung von Problemen & Algorithmen (Logik-Kalküle, Graphtheorie, Lernbarkeitstheorie, usw.)

- Wissensbasierte Systeme modellieren eine Form von **rationaler Intelligenz** für Expertensysteme.
- Aufgaben von KI in WBS:
  - ▷ Antworten auf Fragen des Anwenders liefern auf Grundlage von formalisiertem Fachwissen und daraus gezogenen logischen Schlüssen
  - ▷ Unterstützung für Problemlösen und Planen bieten: Suchstrategien und Techniken der Problemreduktion (Teilzielbildung)

  Bsp.: ***Cyc*** (maschinenauswertbare Alltagswissen-Datenbank), IBM's ***Watson*** (Antworten auf Fragen in natürlicher Sprache)

9.2 „KI-*Hype*"

- Google's ***AlphaGo*: Monte-Carlo-Algorithmen, Lernmethoden für tiefe neuronale Netzwerke**. Verschiedene Netze integriert:
  - ▷ ***policy network*** („Regelnetzwerk") trainiert durch viele Partien, überwachtes Lernen, bestärkendes Lernen; kennt Grundregeln des Spiels.
  - ▷ ***value network*** („Bewertungsnetzwerk") dient der Bewertung der aktuellen Spielsituation; trainiert durch bestärkendes Lernen.
  - ▷ **Monte-Carlo-Baumsuche** rechnet die Varianten durch, kombiniert alle Komponenten.
- ***Convolutional Neural Network*** (***CNN***): können selbst herausfinden, welche Features am besten sind, um eine gegebene Problemstellung zu lösen.
- Automatisiertes Autofahren
  - ▷ ***Level* 0:** Fahrer fährt selbst.
  - ▷ ***Level* 1:** Fahrassistenz. **Bsp.:** Abstandsregeltempomat (*Adaptive Cruise Control*)
  - ▷ ***Level* 2:** Teilautomatisierung. Automatisches Einparken, Spurhalten, Beschl., ... **Bsp.:** Stauassistent, Tesla *Autopilot*
  - ▷ ***Level* 3:** Bedingungsautomatisierung. Fahrer muss das System nicht dauernd überwachen, Fahrzeug führt selbstständig Auslösen des Blinkers, Spurwechsel, Spurhalten durch.
  - ▷ ***Level* 4:** Hochautomatisierung. Führung wird dauerhaft vom System übernommen, kann an Fahrer abgegeben werden wenn nötig.
  - ▷ ***Level* 5:** Vollautomatisierung. Kein Fahrer erforderlich, nur Eingabe des Ziels.

10 Maschinelles Lernen / *Deep Learning* in WBS

10.1 Maschinelles Lernen

- **maschinelles Lernen** (ML): „künstliche" Generierung von Wissen aus Erfahrung.
  - ▷ Künstlich erschaffenes System lernt aus vorgegebenen Beispielen und kann dann Muster in den Lerndaten erkennen.
  - ▷ Vorhersagen machen, diese evaluieren und als neue Trainingsdaten verwenden, falls diese gut war.
- *Deep Learning* (*DL*) ⊂ ML ⊂ *AI*
- **statistisches Lernen:** Menge von Methoden zur Erkennung von Strukturen und Zusammenhängen in Datensätzen.

  Kategorien des statistischen Lernens:
  - ▷ ***supervised learning* (überwachtes Lernen):** Lernen aus Beispielen (gegebene Trainingsdaten)
    - – Ziel: Generalisierung
    - – Grundtypen: **Klassifikation** & **Regression** (s. u.)
  - ▷ ***unsupervised learning* (unüberwachtes Lernen):** Lernen durch automatische Problemlösungsstrategien (automatische Gruppierung, Strukturierung, Suche nach Mustern in der Datenbasis), keine vorgegebene Klassifikation.

    Bsp.: k-*means Clustering* (s. u.)
  - ▷ ***reinforcement learning* (bestärkendes/optimierendes Lernen):** über Belohnung/Bestrafung (Punktesystem)
- Grundlegende Arten von ML:
  - ▷ **Klassifikation:** prädiktives Modellierungsproblem, Klassenbezeichnung für ein gegebenes Beispiel vorhersagen. **Bsp.:** Spamvorhersage, Handschrifterkennung auf Zeichenebene, Nutzerverhalten
  - ▷ **Regression:** Vorhersage eines stetigen (reellwertigen) Wertes. **Bsp.:** Preis eines Hauses anhand Größe, Lage, usw.
    - – **einfache lineare Regression:** häufige Art der Regressionstechnik; Zielvariable y anhand von Eingangsvariable x vorherbestimmen mit Unterstellung einer linearen Beziehung.
  - ▷ ***Clustering*:** Rückschlüsse aus Datensätzen ziehen, die aus Eingabedaten ohne Daten*label* bestehen; Sinnvolle Strukturen/Gruppierungen in einem Datensatz von Beispielen finden.

    Bsp.: k-*means Clustering*: einfachster Algorithmus für unüberwachtes Lernen, unterteilt n Beobachtungen in k *Cluster*.
- **Entscheidungsbäume (*Decision Trees*):** aus gegebenen Traininsbeispielen und einer Klasseneinteilung ein Modell in Form eines Baums erstellen. Ordnet neue Beispiele (mit gewisser Fehlerrate) einer vorgegebenen Klasse zu.

  Grundlage: **Konzepthierarchien** (aufeinanderfolgende, hierarchische Entscheidungen).

  Einsatzgebiete:
  - ▷ Wahrscheinlichkeitsrechnung mit bedingten Wahrscheinlichkeiten $P(A|B)$
  - ▷ *Data Mining*
  - ▷ Entscheidungstheorie
  - ▷ *Business Rule Management Systems*

10.2 *Computer Vision*: *CNNs*

- (Teilgebiet *Deep Learning*)
- *Convolutional Layers* sind rechnerisch viel effizienter als vollständig verbundene *Layer*.
- Komplexe Aneinanderreihung einfacher Rechenoperationen: z. B. nur Skalarprodukte & $\max$-Operationen.

10.3 *DL*-Trends: *Generative Adversarial Networks*

- ***Generative Adversarial Networks* (*GANs*):**
  - ▷ ***Generator*** G**:** erzeugt Daten; versucht D zu überzeugen, dass diese Daten in Originaldatensatz enthalten sind.

    Muss lernen, gute Daten zu „erfinden".
  - ▷ ***Discriminator*** D analysiert, ob diese Daten tatsächlich dazugehören.

    Muss lernen, gut zwischen den Quellen unterscheiden zu können.

  Analogie: Fälscher G und Ermittler/Experte D
- Verwendung von *GANs* in WBS: Füllen von Wissensbasis mit „künstlich erzeugten" Informationen, **Bsp.:** Generieren von weiteren Bildern aus einer textuellen Beschreibung.