Wissensrepräsentation
Vorlesung

Sommersemester 2008

6. Sitzung

Dozent

Nino Simunic M.A.

Computerlinguistik, Campus DU

- **Wissen repräsentieren**
  - **Semantische Netze**
  - **Frames**

Definition »Wissen« (in der KI)

»**In AI**, a representation of
knowledge is a combination of data structures and interpretive procedures
that, if used in the right way in a program, will lead to 'knowledgeable' behavior.«

»Remember that

a data structure is no more knowledge than an encyclopaedia is knowledge.

We can say, metaphorically, that a book is a source of knowledge, but without a reader, the book is just ink on paper.«

Barr, Avron, and Edward A. **Feigenbaum**, editors. 1981. *The Handbook of Artificial Intelligence*, Volume 1: 143. (Reading, MA: Addison-Wesley, 1989)

Programmieren != **Wissen repräsentieren**

Aufgabe/Problem

Informell, semi-formal, formal

Wissen akquirieren, identifizieren, modellieren

Algorithmus entwerfen

Wissensrepräsentationsformalismus wählen

Programmiersprache wählen

Semantische Netze, Ontologie, Index, Taxonomie, Thesaurus, Frames, Produktionsregeln, Logik, …

Wissensrepräsentation

Formale Abbildung von Wissen in Wissensbasierten Systemen

Wissen beschreiben (deskriptiv)

Implementieren

Strategische Anwendung, Inferenz

Programm ausführen

Problemlösung

Bisher

- FSM, Tabellen, Regeln … effiziente Repräsentation in vielen Fällen. Auch kognitiv adäquat?
  - Was, wenn Assoziationen, Strukturen, Sequenzen, Situationen, … dargestellt werden sollen?
- Beobachtung am **Menschen**: Hochgradig **vernetztes Wissen**, bestehend aus **Konzepten**, **Individuen**, **Beziehungen**, **Gesetzmäßigkeiten**.
  - Abstraktion, Struktur, Vererbung, Klassifikation, Vernetzung/Assoziationen, …
- Definition im engeren Sinne (insbesondere KI, Computerlinguistik): **Wissensrepräsentation im Rechner soll menschlichem Wissen (kognitiv adäquat) entsprechen!**

Annahme über menschliches »Wissen«

Was ist das?

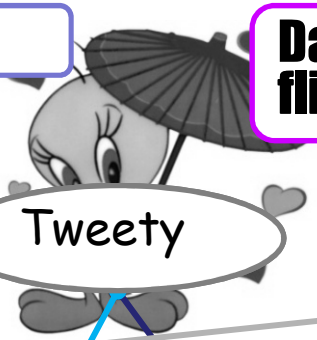
Das sind Vögel. Vögel haben Flügel, Schnabel

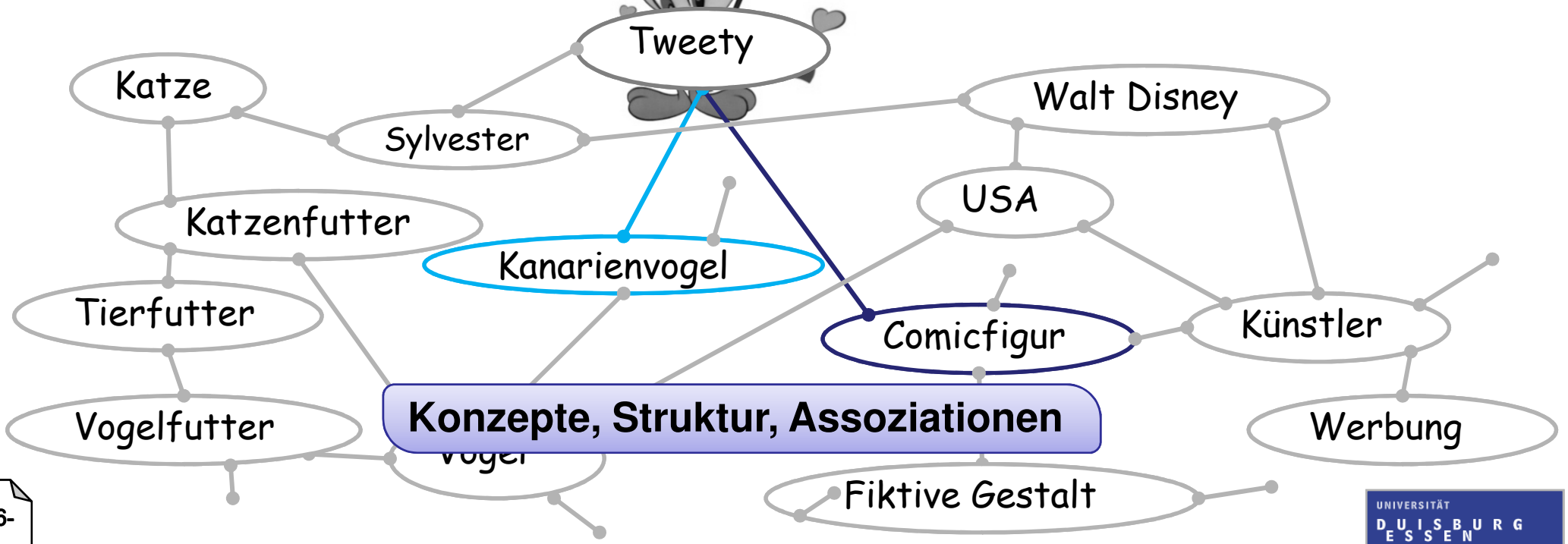
Gruppe von Individuen mit gleichen Eigenschaften.

Objekt mit gleichen Eigenschaften einer Gruppe.

Was ist das?

Das ist ein Vogel, weil es Federn hat, fliegen kann, einen Schnabel hat, ...





Wissen

Gegenstandsbereich

Natur, Comicwelt,...

Erfahrungen führen … zu Informationen

über Objekte, Abstraktes, Ereignisse, …

Organisiert in **Konzepte** und ***Beziehungen*** zu-/untereinander

Tweety *ist ein* **Vogel**, Tweety *ist* gelb, Vogel *hat* **Federn**, Sylvester *mag* Tweety, **Katze** *meidet* Wasser …

… Gesetzmäßigkeiten

Wenn ein Vogel eine Katze sieht, fliegt er weg …

+ Strategische Anwendung/ Inferenz → Problemlösung

Grundlegende Wissenseinheit: **Konzept**

Konzept: Abstrakte **Beschreibung** einer **Menge von Individuen mit gleichen Eigenschaften**

Person

männlich/weiblich, hungrig/satt, Fingerabdruck, Mitarbeiter/Putzfrau/…

Fußballspiel

22 Spieler, Tor, Abseits, …

Vogel

Federn, Nest, fliegt, …

Aspekte

Person ist bzw. **kann sein**

männlich/weiblich, hungrig/satt,
blond/braun/schwarz, weiß/schwarz/gelb/grün,
vorbestraft/sauber, krank/gesund, arm/reich,
lustig/langweilig, dumm/intelligent,
verheiratet/geschieden, lebendig/tot,
arbeitslos/beschäftigt/, betrunken/nüchtern, …

Wenn es regnet →

Straße nass, Zaun nicht streichen,
Regenschirm mitnehmen, Regenwürmer
kommen aus dem Boden, …

Dass Regenwürmer aus dem Boden kommen, ist irrelevant, wenn Sie bei Regen mit dem Auto nach Paris fahren.

Heterogene Informationen in einer Datenstruktur

Homogene Informationen zu Konzepten: Organisation nach Wissensdomäne / Gegenstandsbereich

Wissensdomänen

Kontext »Polizei« Wenn es regnet → Unfallgefahr ++
Name, Haarfarbe, Fingerabdruck, Vorstrafen, …

Kontext »Finanzen«
(Name,) Einkünfte, Immobilienbesitze, … Haarfarbe irrelevant

Kontext »Medizin« Wenn es regnet → Mehr Leute mit Schnupfen
(Name,) Krankheit, Blutgruppe, …

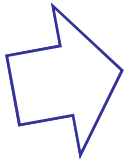
Kontext »Einzelhandel« Wenn es regnet → Weniger Kunden
(Name,) Merkmale im Bereich Kaufpotential, …

Kontext »Biologie« Einkünfte irrelevant
Lebewesen mit biologischen Eigenschaften, …

Kontext »Physik«
Objekt mit Gewicht, Masse, … Fingerabdruck irrelevant

Strukturelle Wissensorganisation: Überblick

Subsumtion



Hierarchische Beziehung, Taxonomie

Klasse X ist allgemeiner als Klasse Y

⇔ Klasse X subsumiert Klasse Y

Person ist allgemeiner als / subsumiert Mitarbeiter

(Instanz-)Klassifikation

Objekt A ist vom Typ (Klasse) X

⇔ Objekt A ist Instanz der Klasse X

Erwin ist vom Typ Mitarbeiter

Aggregation

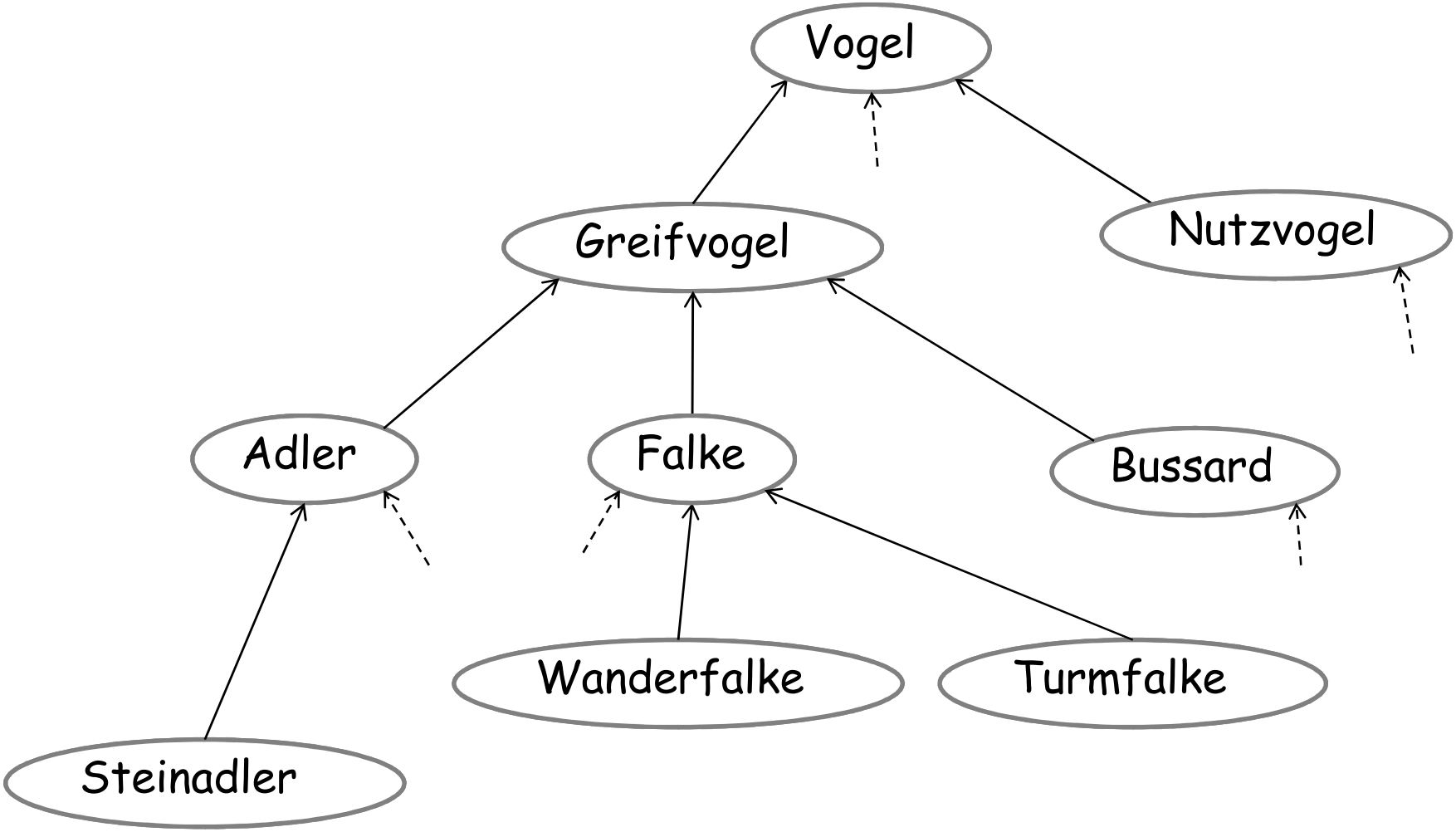
Struktur, aber nicht Im hierarchischen, sondern im partitiven Sinne

Objekt A besteht aus den Teilen T1, T2, ...

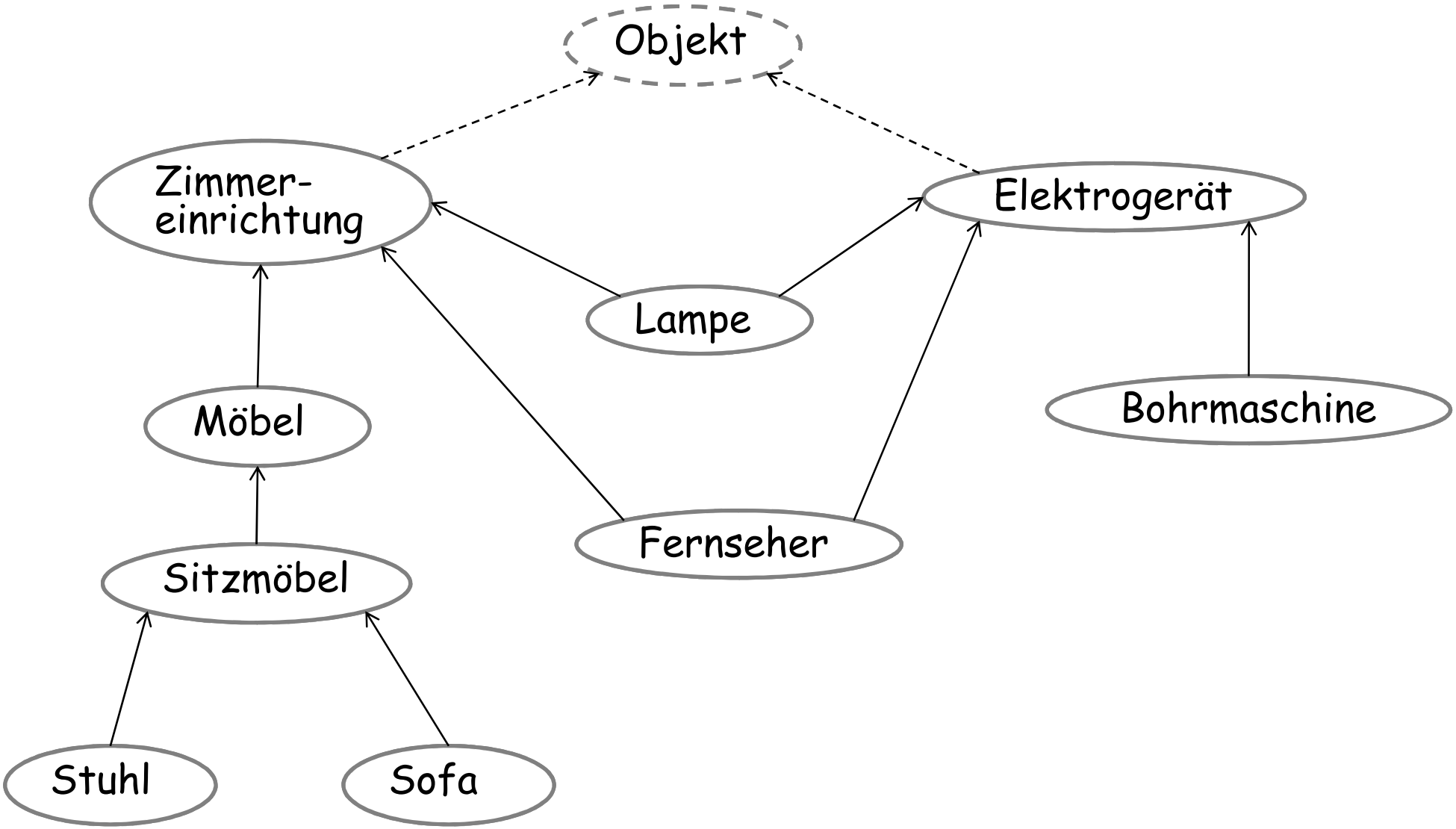
⇔ T1 ist Teil von Objekt A, T2 ist Teil von, …

Computer besteht aus Tastatur, Monitor, Mainboard, …

Strikte Taxonomie / Hierarchie: Maximal eine Superklasse



Multiple Hierarchie (und i.d.R. Vererbung)



Assoziative Relationen / Beziehungen

- **Nicht-strukturelle Beziehungen (zwischen Konzepten)**
  - Person ist … (männlich-weiblich/hungrig-satt/groß-klein/…)
  - Person kennt … (Person/Geschäft/…)
  - Person hat … (Augen/Fingerabdruck…)
- **Heterarchie**: Nicht-(be-)herrschende Beziehungen

Kognitiv adäquate Repräsentation von Wissen: Semantische Netze, Frames

Semantische Netze und Frames zur Repräsentation von (Domänen-)Wissen

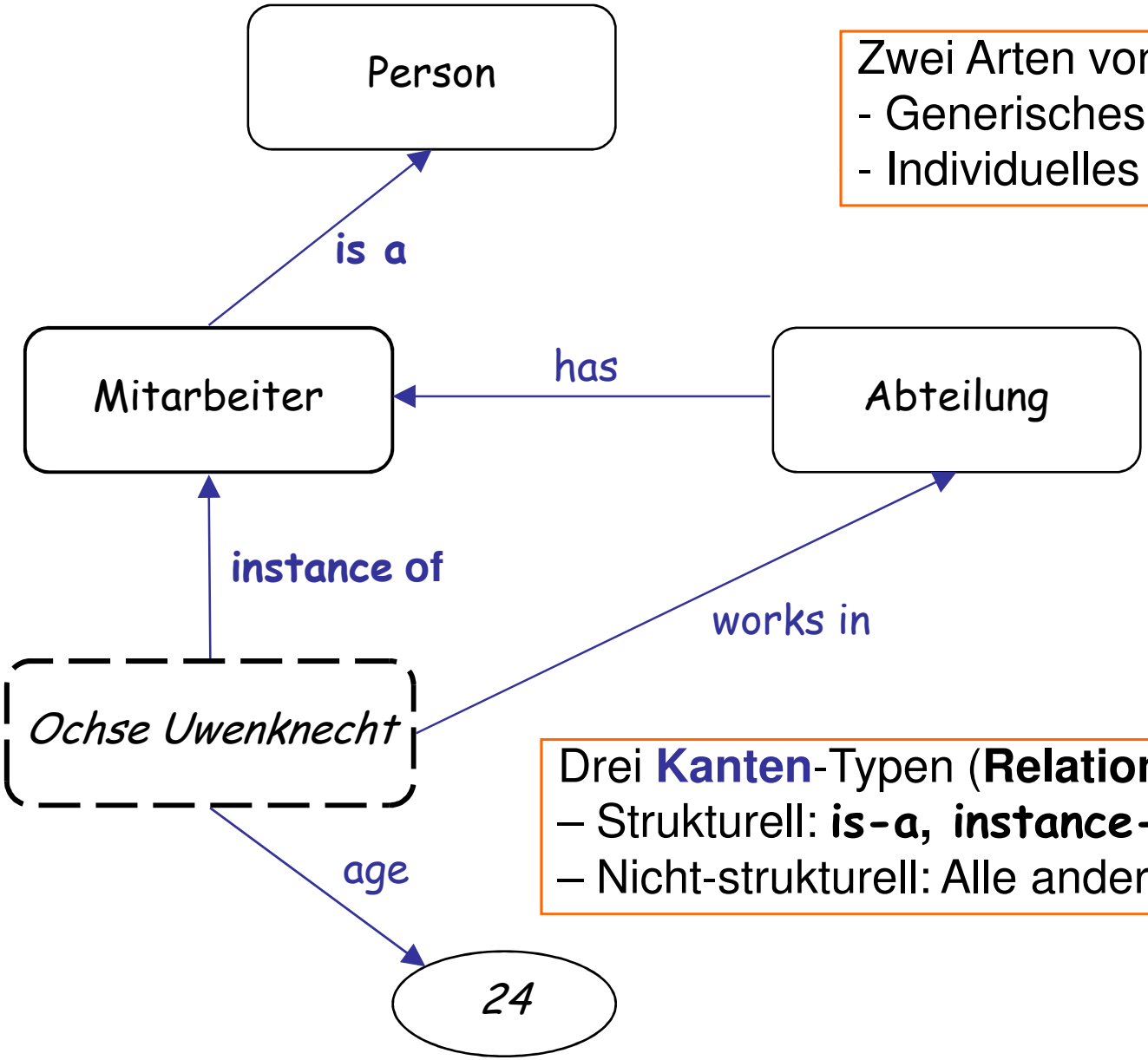
- Basieren auf Erkenntnissen über das menschliche Gedächtnis:
  - Mehrdimensionale Verbindungen/Assoziationen zwischen Informationen
- **Semantische Netze**:
  - Hochgradige assoziative/strukturelle/klassifizierende Vernetzung
  - Besser: Wissensorganisation in Form von Konzepten mit assoziierten Beschreibungen.
- **Frames**:
  - Wissen wird in Form von Konzepten (Datenstruktur: Frame) organisiert

Starke Analogien

Semantische Netze: Graphische Wissensrepräsentation

- Zwei Primitive: Knoten, Kanten
  - **Knoten**: Instanzen / Klassen von Objekten
    - Enthalten keine Information per se
  - **Kanten:** 2-stellige Objekt-Relationen
- Bedeutung eines Knotens bzw. Konzepts
  - Vollständig repräsentiert durch von im ausgehende Kanten

Beispiel-Netz



Zwei Arten von **Knoten**:
- Generisches Konzept (**Klasse**)
- Individuelles Konzept (**Instanz**)

Drei **Kanten**-Typen (**Relationen**)
– Strukturell: **is-a, instance-of**
– Nicht-strukturell: Alle anderen 2-stelligen Relation

2-stellige Relationen

Relation (»Prädikat«) »Subjekt« »Objekt«

- is-a(Mitarbeiter, Person)
- has(Abteilung, Mitarbeiter)
- works in(Mitarbeiter, Abteilung)
- age(*Ochse U., 24*)

Was ist mit dreistelligen Relationen?

Direktes Objekt

Schenken(Wer, Wem, Was)

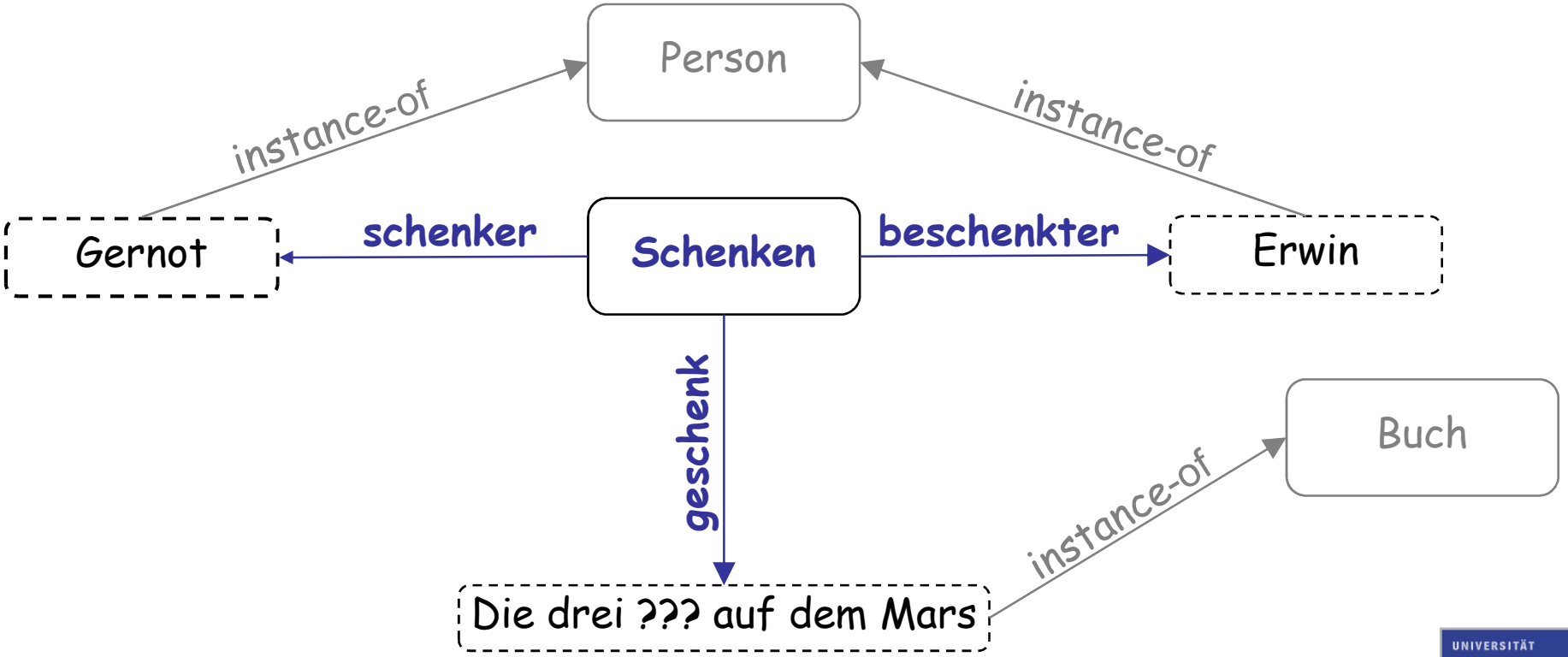
Subjekt Indirektes Objekt

3-stellige Relation → 2-stellige Relation

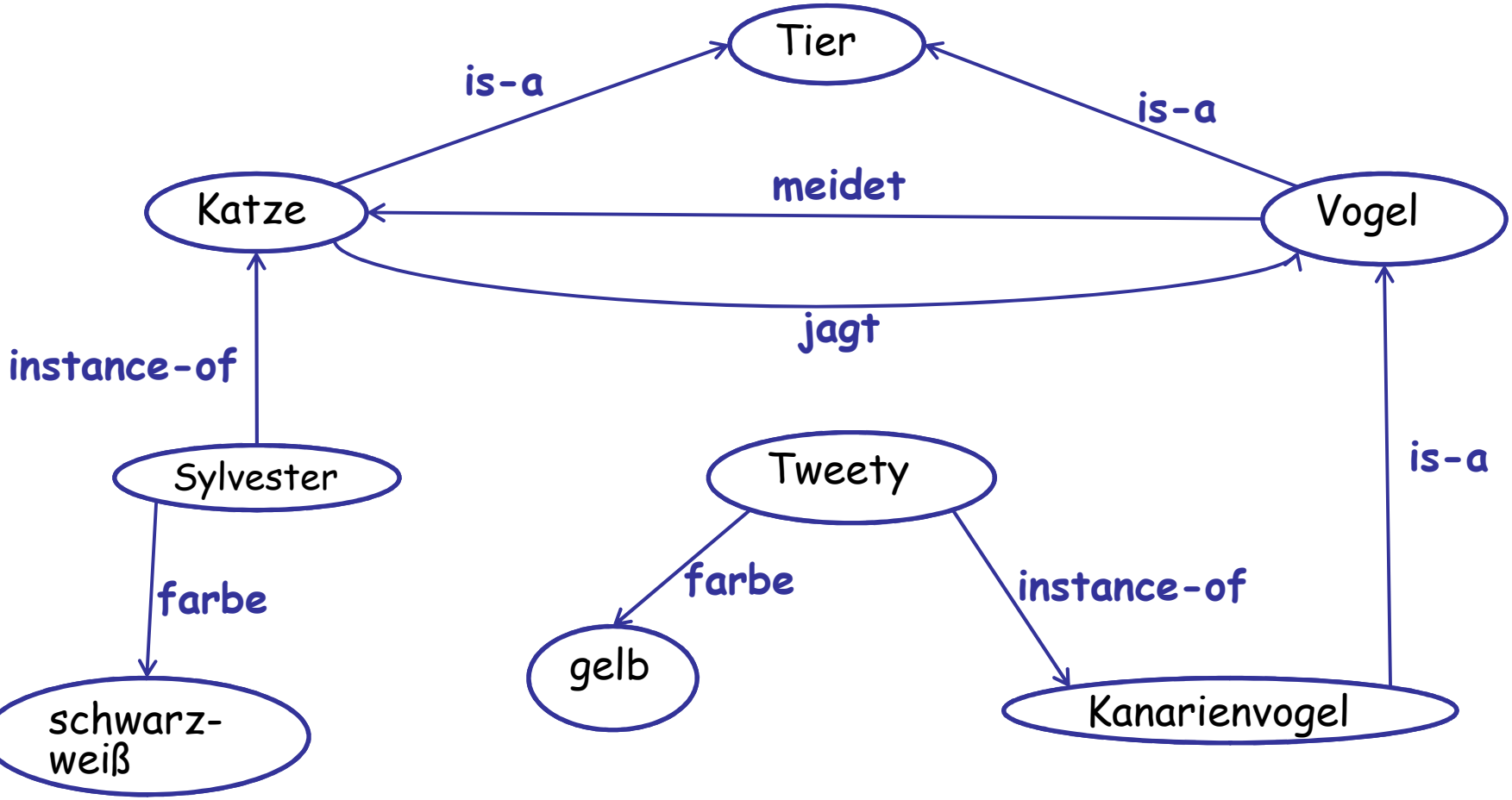
Mehrstellige Relationen (n>2) müssen in assoziativen Netzen auf 2-stellige abgebildet werden

→ Knoten, der die Relation selbst definiert.

Beispiel: Gernot schenkt Erwin ein Buch.

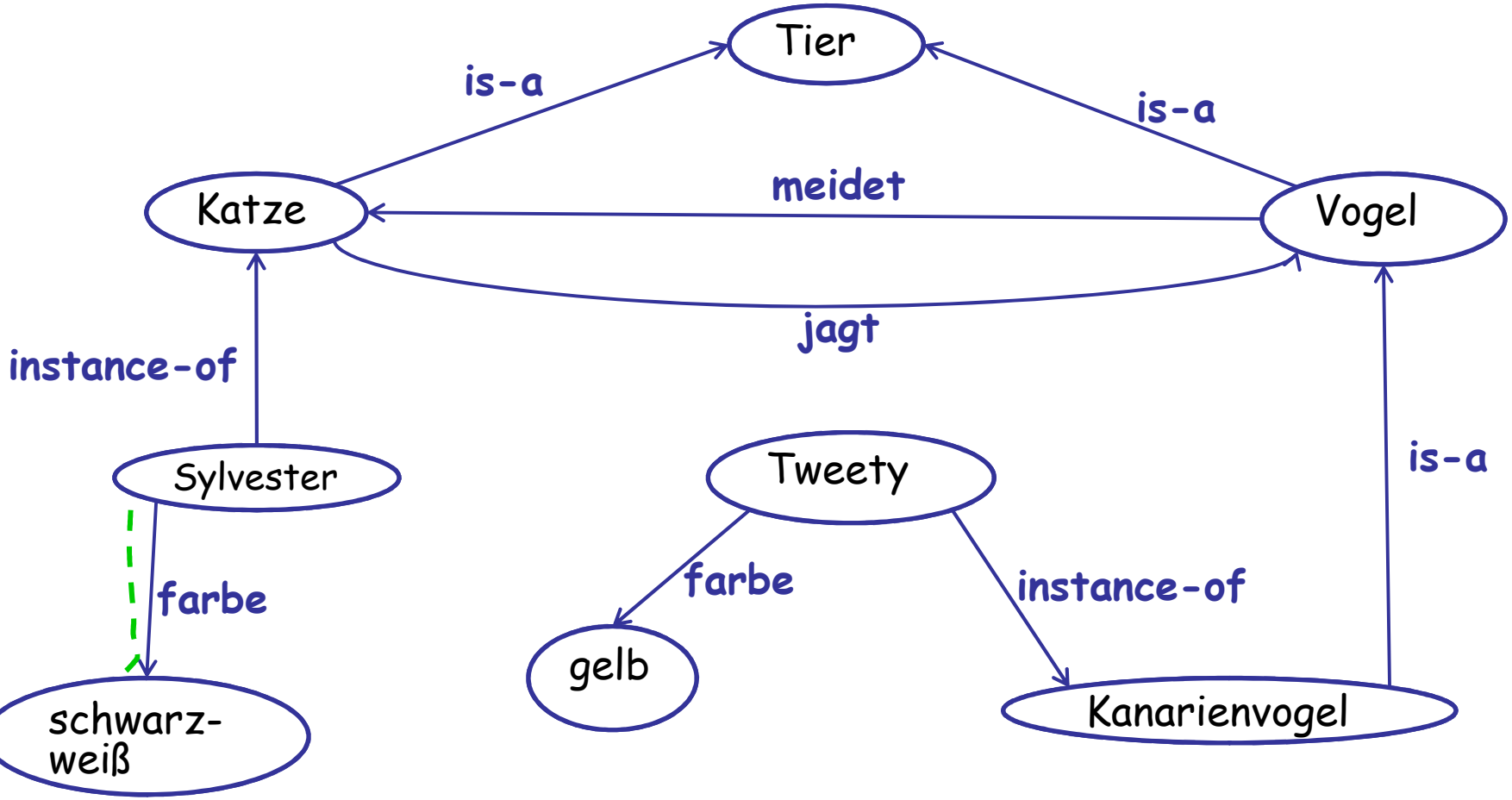


»Lesen« und »Verstehen«
Semantischer / Assoziativer Netze



- **Q**(uestion)**& A**(nswer)
- ***Spreading Activation***
- **Vererbung**

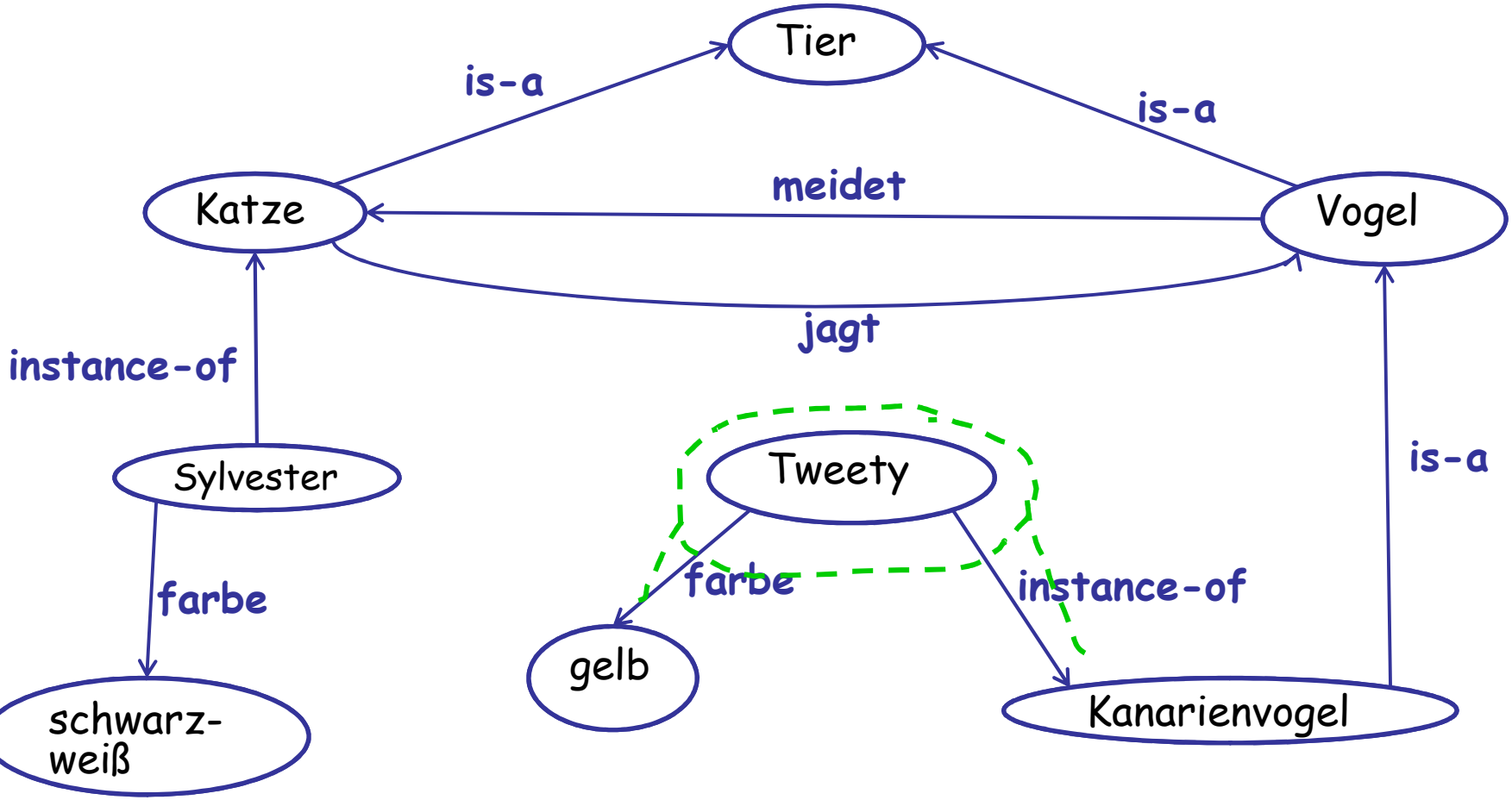
Q(uestion)**& A**(nswer)



»Wie ist der Wert der **Relation** farbe von Sylvester?«

Verfolgen von Kanten: Wert der Relation ***R*** eines Knoten ***K***?

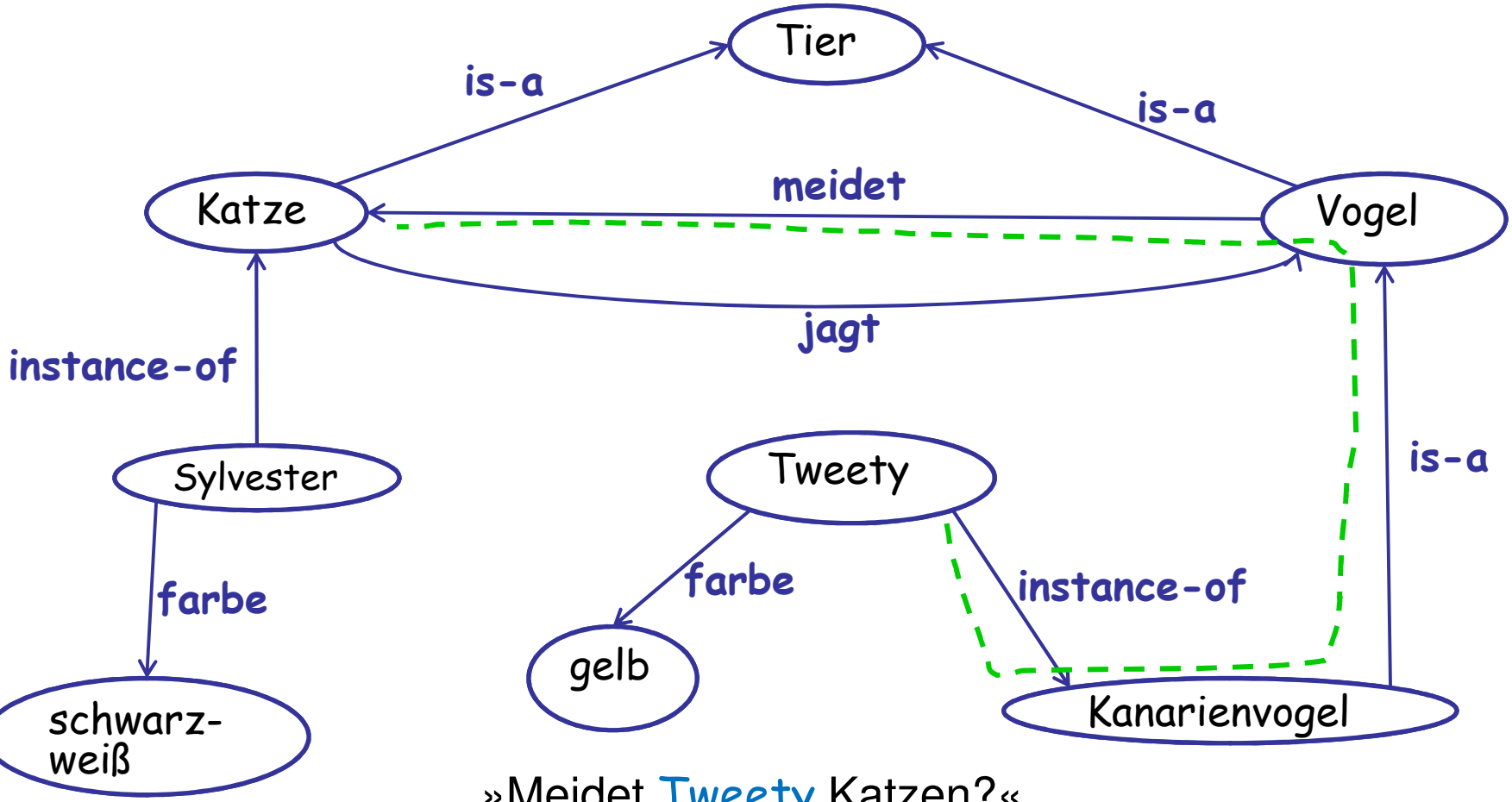
Spreading Activation (*Intersection Search*): Beziehungen zwischen Knoten



»Welche Beziehung haben **gelb** und **Kanarienvogel**?«
(*gelb* ist die Farbe von **Tweety**, und **Tweety** ist ein **Vogel**.)

Finde einen Knoten **K**, der von weiteren Knoten ($K_1, \ldots, K_n$) erreichbar ist.

Vererbung



»Meidet Tweety Katzen?«

instance-of und is-a Pfaden folgen um zu sehen, ob fehlende Informationen in der Hierarchie weiter oben gefunden kann.

Tweety→Kanarienvogel→Vogel→meidet→Katze.

Knowledge Engineering Tipps: **Identifizieren, Modellieren**

- **Relevante Entitäten** identifizieren
  - Als Instanzen von **Klassen** modellieren.
    Klassen(-strukturen) aufbauen.
  - Zusammenhänge zwischen auftretenden Klassen frühzeitig erkennen und ggf. mitmodellieren
    Gemeinsame Oberklassen, Links etc.
  - Komplexe Begriffe wenn möglich zerlegen
    »Landschaftsfoto« → *Foto hatMotiv Landschaft*
- **Eigenschaften** identifizieren
  - Attributive, assoziative
  - Als Kanten / Relationen
- **Werte** der Eigenschaften
  - Als Objekte von Klassen (String, Integer, Katze, …)
  - Evtl. neue Klassen nötig

Knowledge Engineering Tipps: **Allgemeine Guidelines**

- **Zu vermeiden**
  - **Zu allgemeine Konzepte** ohne weitere Verfeinerung

    »Zeug«
  - **Zu spezifische Konzepte** ohne Superklassen

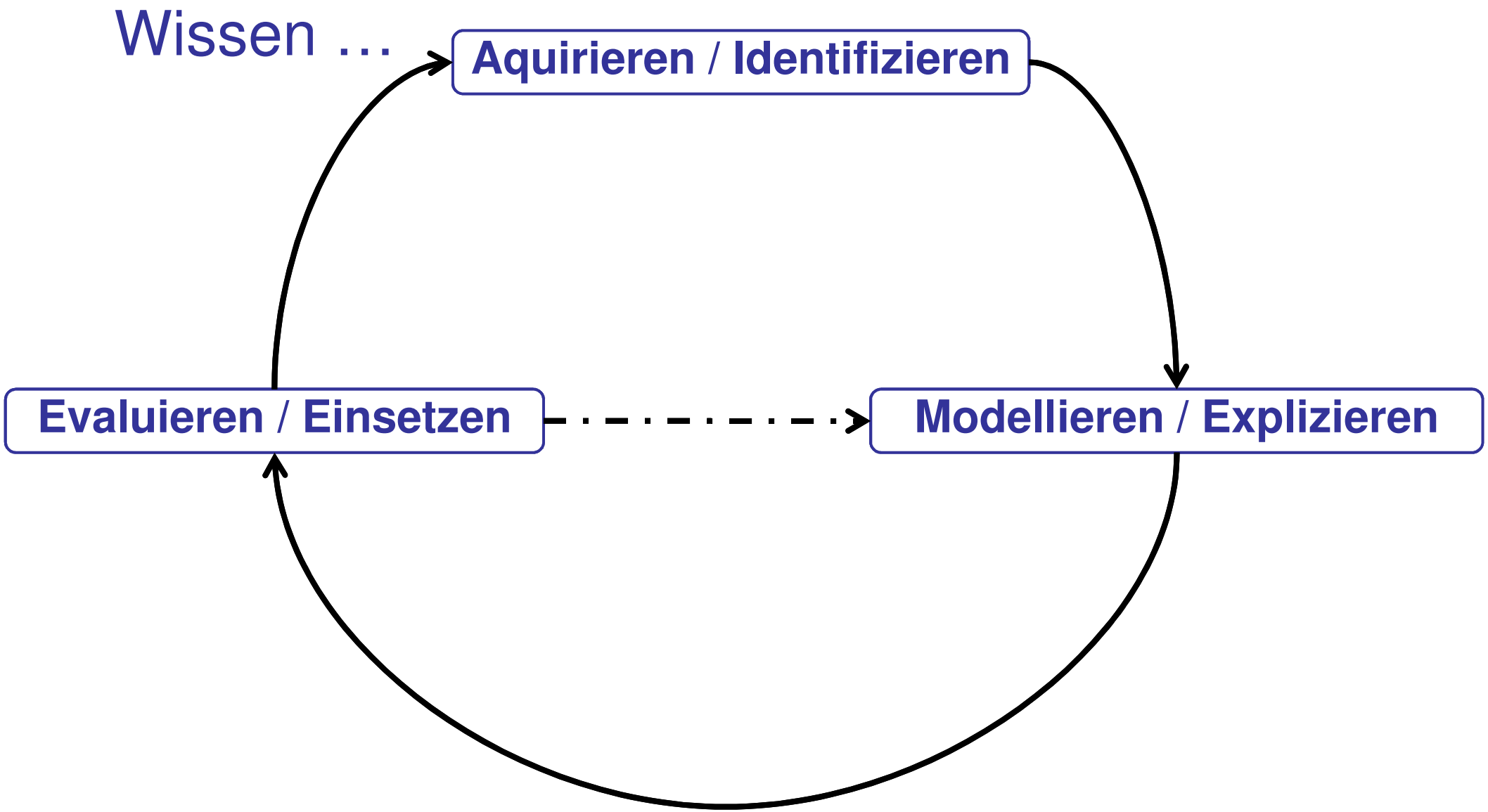
    »Dampfschiffahrtsgesellschaftskapitänsanwärter«
- **Homogene** Repräsentation essentiell
  - **Einheitliche Modellierung** gleichartiger Dinge

    Anwendungsdomänen-orientiert modellieren, nicht mischen.
    Tweety als Comicfigur oder als Vogel, nicht beliebig mischen.
  - **Wiederholung** von Werten und Klassen vermeiden

    Lieber Wiederverwendung/Mehrfachnutzung
  - Relationen/Bezeichner i.A. **standardisieren**

    PKW, *Karre*, *Flitzer*, .. für **ein** Konzept (Auto) vermeide*n*

Knowledge Engineering: **Zyklik**

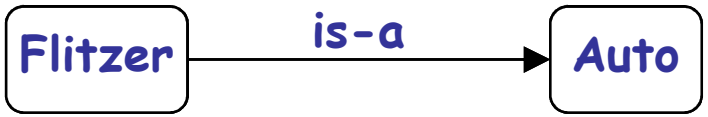


Semantik Semantischer Netze

- Lesen/Interpretieren für Menschen bzw. Eingeweihte unproblematisch: Knoten und Kanten tragend sinnhafte Bezeichner
  - Knoten *Hund* → Bedeutung (Konzept) *Hund*
- **Aber**: Semantik ist in der Form maschinell verwertbar

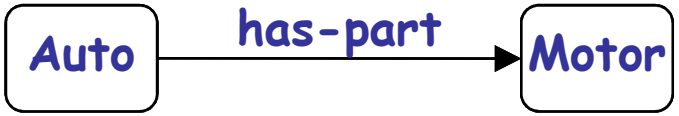
  → Irreführender Begriff »Semantisches Netz«
- (Maschinell verwertbare) Semantik assoziativer Netze (Formalismus) ergibt sich durch Übersetzung in z.B. PL1 (Sprache)

Beispiele



$(\forall x)\,[\,\text{Flitzer}(x) \rightarrow \text{Auto}(x)\,]$

Flitzer(batmobil)



$(\forall x)\,[\text{Auto}(x) \rightarrow (\exists y)\,[\text{Motor}(y)\ \&\ \text{has-part}(x,y)]]$

$(\forall x)\,[\,\text{Auto}(x) \rightarrow \text{farbe}(x, \text{schwarz})\,]$

farbe(batmobil, schwarz)

Motivation für Frames

- Basiert auf der Annahme, dass unsere **Informationsverarbeitung** wesentlich von unseren **Erwartungen** abhängt [Minsky, 1981].
- Minsky: Neue Situation / Sicht auf ein Problem ändert sich gravierend
  - → Suche nach passenden Situationen im Gedächtnis
  - → Gespeicherte Situation wird auf neue Situation angepasst
  - → Aus bekannter Situation ergeben sich Erwartungen über die neue Situation.
    - Beispiele: Restaurantbesuch, Geburtstagsparty, Wohnzimmer
  - → Sonst: Erstellen neuer Situationsstruktur
- **Frames**: Datenstruktur zur Repräsentation stereotyp. Situationen
- **Frame-Inferenz:** Vorwiegend Erkennungsprozess

Frames: Objektbasierte Wissensrepräsentation

Bezeichner		
Slot 1	**Slotbeschreibung 1**	**Slotwert 1**
Slot 2	**Slotbeschreibung 2**	**Slotwert 2**
Slot 3	**Slotbeschreibung 3**	**Slotwert 3**
...		

- **Frames** fassen alle Eigenschaften eines Objekts in einer Datenstruktur zusammen
- Bestandteile eines Frame
  - **Bezeichner** des Frame
  - **Slots** mit **Slotbeschreibungen**, **Slotwerten**

Frames: Objektbasierte Wissensrepräsentation

Slotbeschreibungen (Facetten) zum Einschränken von Slots. Z.B. durch

- **Typ**
- **Kardinalität**
- **Vererbung**shinweise

Mögliche **Slotwerte**

- **Primitive Werte**, z.B. Strings, Integer,...
- **Referenzen** auf andere Frames
- **Prozeduren** (*procedural attachment*)

Erwin

Name	**Typ:** `String` **Min. Kardinalität:** `1`	**Schuster**
Vorname	**Typ:** `String` **Min. Kardinalität:** `1`	**Erwin**
Alter	**Typ:** `Integer` **Kardinalität:** `1`	**31**
member-of		`Mitarbeiter`
...		

Syntax: Allgemeine Primitive in Framesystemen

Zwei Arten von Frames:

- **Klasse** (Konzept)
- **Instanz** (Individuum)

Unterscheidung von mindestens drei Arten von Slots

superclass

Relation zwischen Klassen (⇔ is-a)

member-of:

Relation zwischen Instanz und Klasse (⇔ instance-of)

assoziativ:

Beliebige Slots für Attribute

Beispiel Frames

```
[Mensch
        superclass: lebewesen
        geschlecht:
                type: männlich ODER weiblich
                maxKardinalität: 1
        vorname:
                type: string
        name:
                type: string
        alter
                type: integer
        eltern
                type: mensch
                max Kardinalität: 2
]
```

```
[Mann
        superclass: Mensch
        geschlecht: männlich
]
[Vater
        superclass: Mann
        kinder:
                type: Mensch
                min Kardinalität: 1
...
```

Beispiel Frames

```
[Reise
        reisender:
                type: mensch
        ermäßigung:
                type: [0..100]
        proc:   if reisender.alter < 18
                then 50
                else if reisender.alter > 60
                then 25
                else 0
        exec:   if-needed
]
```

```
[EK
        member-of: Vater
        vorname: »Erwin«
        nachname: »Krämer«
        alter: 61
]
```

```
[reise123
        member-of: Reise
        reisender: EK
]
```

Beispiel Frames

```
[rechteck
   seite1
      type: real
   seite2
      type: real
   fläche
      type: real
   exec: if-needed
   proc: (seite1 * seite2)
]
```

```
[quadrat
   superclass: rechteck
   seite2
      type: real
   proc: seite1
      exec: if-added
]
```

```
[q1
   superclass: quadrat
   seite1: 5
]
```

Procedures

- In object-oriented programming languages such as C++ or Java, classes (and hence objects) have **methods associated with them.**
- **This is also true** with frames. Frames have methods associated with them, which are called **procedures. Procedures associated with frames are also called procedural attachments.**
- A procedure is a set of instructions associated with a frame that can be executed on request. For example, a **slot reader procedure might return the** value of a particular slot within the frame. Another procedure might insert a value into a slot (a **slot writer). Another important procedure is the instance constructor, which creates an instance of a class.**
- Such procedures are called when needed and so are called **WHEN NEEDED procedures. Other procedures can be set up that are called automatically** when something changes.

Demons

- A **demon is a particular type of procedure that is run automatically whenever** a particular value changes or when a particular event occurs.
- Some demons act when a particular value is read. In other words, they are called automatically when the user of the system, or the system itself, wants to know what value is placed in a particular slot. Such demons are called **WHEN-READ procedures. In this way, complex calculations can be made** that calculate a value to return to the user, rather than simply giving back static data that are contained within the slot. This could be useful, for example, in a large financial system with a large number of slots because it would mean that the system would not necessarily need to calculate every value for every slot. It would need to calculate some values only when they were requested.
- **WHEN-CHANGED procedures (also known as WHEN-WRITTEN procedures)** are run automatically when the value of a slot is changed. This type of function can be particularly useful, for example, for ensuring that the values assigned to a slot fit within a set of constraints. For example, in our example above, a WHEN-WRITTEN procedure might run to ensure that the “number of legs” slot never has a value greater than 4 or less than 1.
- If a value of 7 is entered, a system message might be produced, telling the user that he or she has entered

Frames interpretieren (Q&A, Vererbung)

Q&A:

*Der Slot **reisender** von **r123** hat den Wert **EK***

Wert von Slot X von Frame Y?

```
[reise123
        member-of: Reise
        reisender: EK
]
```

Auswertung von Prozeduren

if-added: Prozedur eines Slots wird ausgewertet, sobald ein Wert für den Slot eingetragen wird (daten-orientierte Auswertung)

if-needed: Prozedur wird ausgewertet, sobald auf den Slot zugegriffen wird (ziel-orientierte Auswertung)

```
[rechteck
    seite1
        type: real
    seite2
        type: real
    fläche
        type: real
    exec: if-needed
    proc: (seite1 * seite2)
]
```

Beispiel:

fläche von ***q1*** ist ***25***

```
[q1
    superclass: quadrat
    seite1: 5
]
```

Vererbung

Falls eine Information nicht im aktuellen Frame gespeichert ist, folge den **member-of** und **superclass** Slots, um zu sehen, ob die Information in einer Superklasse gefunden werden kann.

Beispiel: Der Slot **geschlecht** von **EK** hat den Wert **männlich**

Matching: Finde Frame, der die aktuelle Situation am besten beschreibt

Beispiel: **[s23 kinder:jens]** passt am besten zu?
Frame **vater**

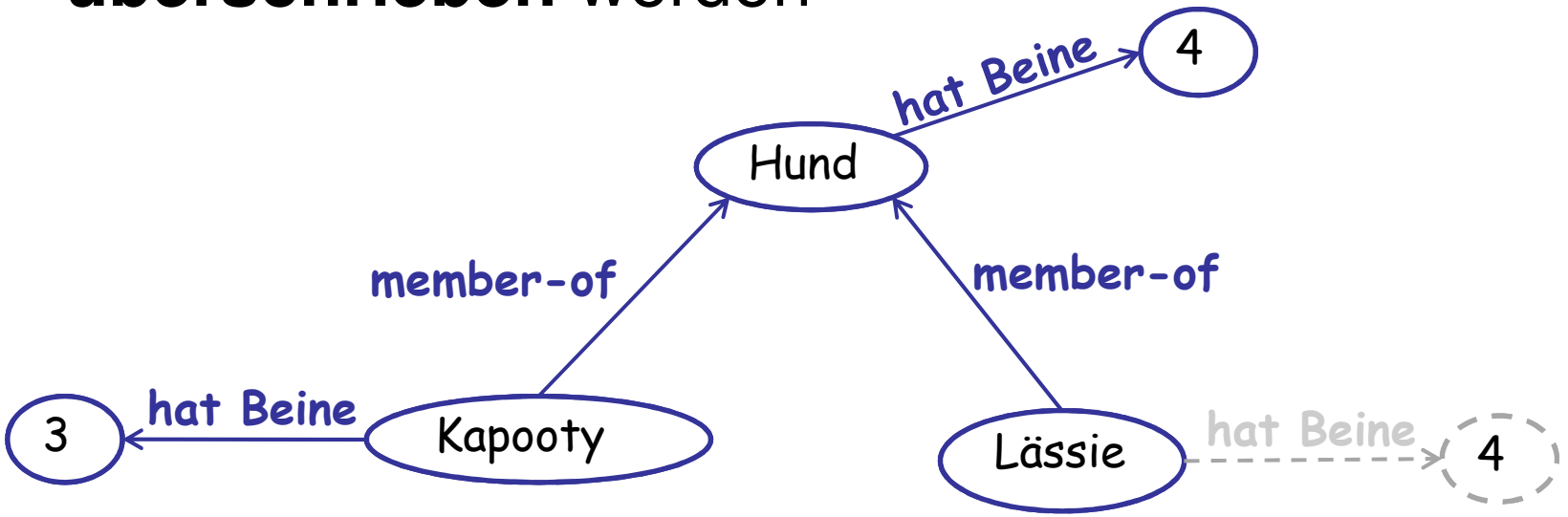
```
[Mann
        superclass: Mensch
        geschlecht: männlich
]
[Vater
        superclass: Mann
        kinder:
                type: Mensch
                min Kardinalität: 1
...
```

```
[EK
        member-of: Vater
        vorname: »Erwin«
        nachname: »Krämer«
        alter: 61
```

```
[Mensch
        …
        vorname:
                type: string
        name:
                type: string
        …
]
```

Default-Werte

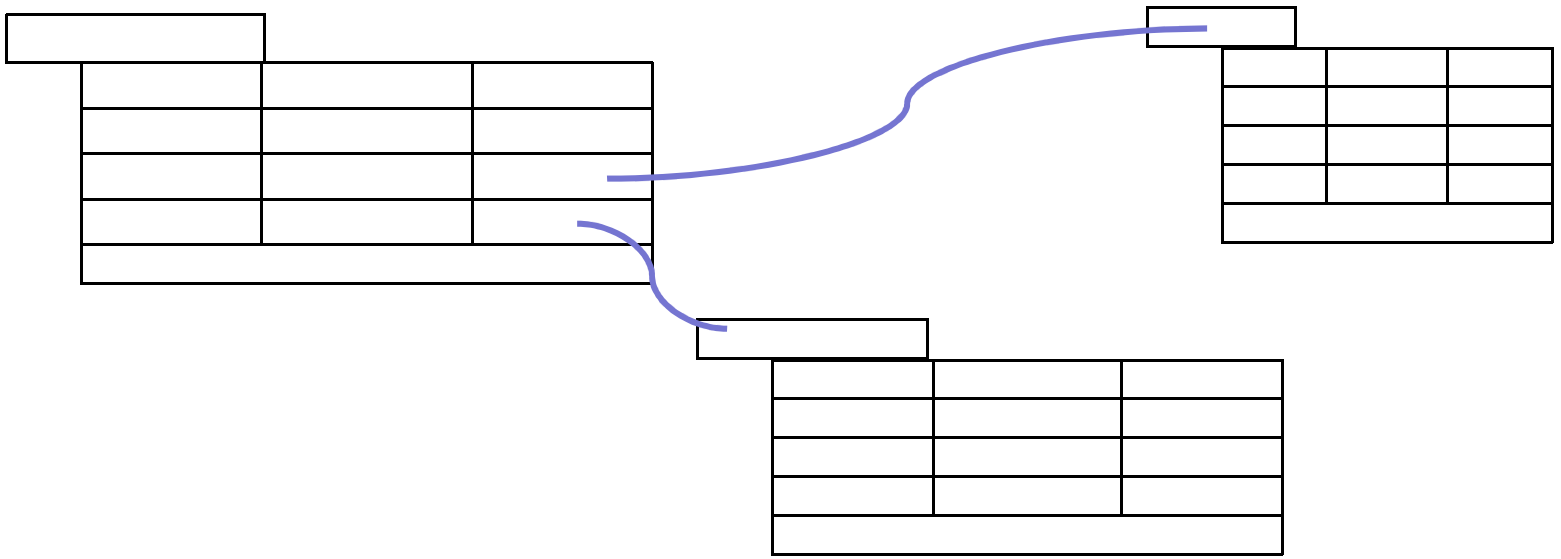
- **Default-Werte** sind **Standard-Werte**
  - **In Klassen** bzw. generischen Konzepten **spezifiziert**
    - Werden vererbt
  - Können bei spezielleren Konzepten bzw. Instanzen **überschrieben** werden



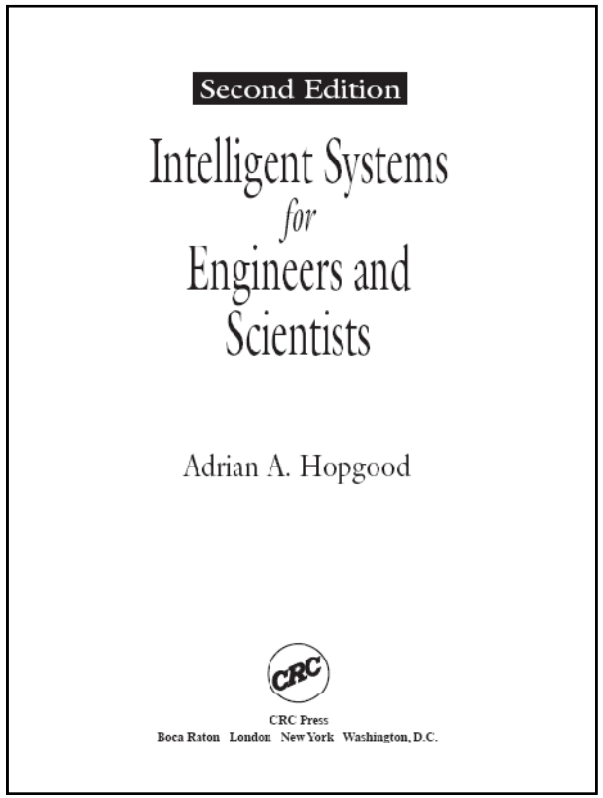
Analogie:
Assoziative Netze - Framesysteme

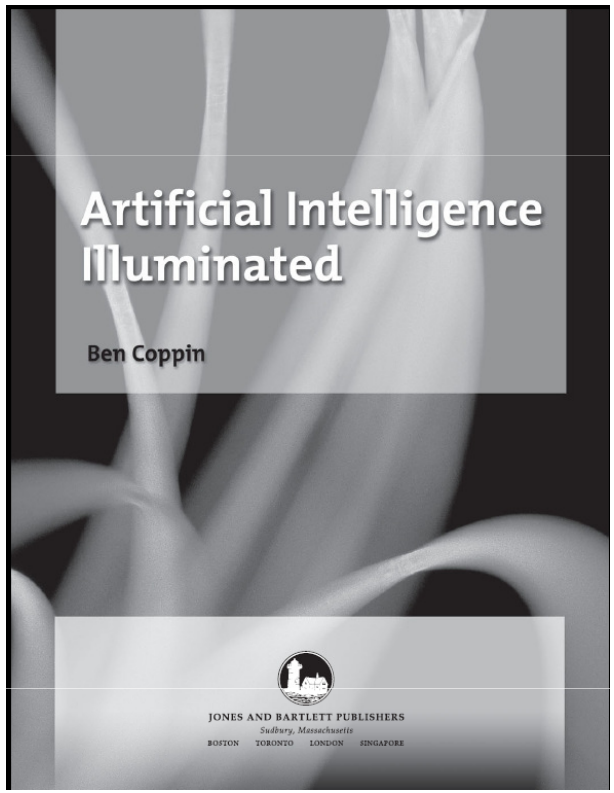
- Frames:
  Objektzentrierte Repräsentation semantischer Netze
  - Slotwerte von Frames verweisen auf andere Frames

  → Netzwerk: Slots entsprechen den Kanten in semantischen Netzen
- Unterschied zu assoziativen Netzen:
  Frame-Systeme erlauben *procedural attachment*



Referenzen, Hausaufgabe





http://www.dfki.uni-kl.de/

Hausaufgabe: Kapitel 3
(2)-Link neben Folien-Download