# Wo kommen die Objekte her?

# Ontologisch-erkenntnistheoretische Zugänge zum Objektbegriff

Wolfgang Hesse[1] , Hubert v. Braun[2]

[1] FB Mathematik und Informatik, Univ. Marburg, hesse@informatik.uni-marburg.de
[2]ASG München, hubert.vbraun@uumail.de

**Zusammenfassung:** *Auch nach über zwanzig Jahren "Objektorientierung" ist der Objektbegriff in der Informatik weitgehend ungeklärt. Zu seiner Klärung können wir komplementär verschiedene Wege beschreiten. Beim ontologisch-erkenntnistheoretischen Zugang geht es um die Herkunft, Abgrenzung und Definition von Objekten. Hier kann die Philosophie einen wertvollen Beitrag leisten, indem sie z.B. die Grenzen eines naiv-realistischen Ansatzes aufzeigt und Begründungen für einen auf Verhandlungen und Konsensbildung gründenden Objektfindungsprozess liefert. Der Ansatz der objektorientierten Analyse, Objekte der Anwendungswelt und ihre verschiedenen Repräsentationen miteinander zu identifizieren, ist semiotisch fragwürdig und trägt zu einer systematischen Konstruktion eines Objektmodells nichts bei.*

***Schlüsselwörter:*** *Objektbegriff, Ontologie, Erkenntnistheorie, Realismus, Idealismus, objektorientierte Analyse*

## 1 Einleitung: "Objekte" - zur Spannweite eines überfrachteten Begriffs

Seit über 20 Jahren spricht man in der Informatik von "Objektorientierung". Eine kritische Analyse des Objektbegriffs hat jedoch - jedenfalls in der Informatik - in dieser Zeit kaum stattgefunden. Vielmehr ist mit der Übernahme des objektorientierten Paradigmas durch viele Teildisziplinen der Informatik eine geradezu inflationäre Ausweitung des Objektbegriffs einhergegangen. Das wird klar, wenn man Schlagworte wie "objektorientierte Analyse", "objektorientierter Entwurf, "objektorientierte Modellierung", "objektorientierte Programmierung", "objektorientierte Benutzer-Oberfläche" oder "objektorientierte Datenbanken" nimmt und sich über die dort jeweils zur Debatte stehenden "Objekte" Rechenschaft ablegt (vgl. z. B. [4,8,9,11,14]).

Beispiele für solche Objekte reichen von lebensweltlichen (konkreten oder abstrakten) "Dingen" wie *Person X*, *Firma Y* oder *Artikel Z* über Vorgänge und Beziehungen wie *Bestellung* (X bestellt Z) oder *Lieferung* (Y liefert Z), weiter über Modellelemente wie Abstrakte Datentypen, Teile von UML-Diagrammen, DB-Schemata, über Elemente von graphischen Benutzer-Schnittstellen wie Fenster, Menüs oder Ikonen, über Variablen, Konstanten und Typen einer Programmiersprache bis hin zu deklarativen oder ausführbaren Programmeinheiten, Dateien, Agenten oder Datenpaketen in einem Rechnernetz. Natürlich ist diese kurze Beispiel-Liste keineswegs erschöpfend.

Will man sich nicht damit begnügen, "Objekt" - wie etwa das deutsche Wort "Gegenstand" - als Universal-Designator zu gebrauchen und ihm damit jegliche Aussagekraft abzusprechen, so wird es Zeit, sich Klarheit darüber zu verschaffen, was wir in der Informatik unter "Objekten" verstehen wollen. Dazu beizutragen hat sich der Arbeitskreis "Terminologie der Softwaretechnik" zum Ziel gesetzt [1,2].

## 2 Zugänge zum Objektbegriff

Zum Objektbegriff gibt es viele unterschiedliche Zugänge. In der Informatik besonders populär ist die Ansicht, dass die Objekte "von Natur aus gegeben" seien und von den System-Entwicklern nur "aufgelesen" und adäquat "abgebildet" werden müssten. Setzt man sich mit einem solchen Standpunkt kritisch auseinander, so wird sofort die außerordentliche Rolle offenbar, die *Sprache* bei der Identifikation und Behandlung von Objekten spielt. Schließlich setzt jeder Umgang mit Objekten eine *Bezugnahme* voraus - und das geschieht neben der sprachlichen Beschreibung durch *Zeichen* - Worte, Symbole, Bilder - oder was immer es uns ermöglicht, auf Objekte Bezug zu nehmen.

Daraus ergeben sich mindestens drei mögliche Zugänge zum Objektbegriff, die anhand der folgenden Leitfragen kurz charakterisiert werden sollen:

- Wo kommen die Objekte her - wer bestimmt, was ein Objekt ist und was nicht?

→ Ontologisch-erkenntnistheoretischer Zugang

- Welche sprachlichen Mittel werden benutzt, um Objekte zu bezeichnen und zu charakterisieren?

→ Sprachlogischer Zugang

- Wie verhalten sich Zeichen und die durch sie bezeichneten Objekte zueinander?

→ Semiotischer Zugang

In diesem Artikel soll es vorrangig um Fragen nach der *Herkunft*, der *Erkennbarkeit* und der *Abgrenzbarkeit* von Objekten, d. h. um den ***ontologisch-erkenntnistheoretischen Zugang*** gehen. Solche Fragen gehören seit jeher zu den Kernfragen philosophischer Diskussion. Im einzelnen sind zwei Teildisziplinen der Philosophie angesprochen:

- die Lehre von den Gegebenheiten des Seins (*Ontologie*)

- die Lehre vom Wissen und von den Prozessen und Grenzen der Erkenntnisgewinnung (*Epistemologie*).

Beide Teildisziplinen sind durch die oben gestellten Fragen in gleicher Weise betroffen: "Objekte" wie die Sterne oder Planeten, aber auch *Kunden* oder *Artikel* werden von uns in der Regel als "gegeben" angesehen, eine Aussage ihrer bloßen Existenz ist also *ontologischer* Natur, während das Studium ihrer Beziehungen, Bewegungen und aufeinander wirkenden Kräfte Erkenntnisprozesse und den Rückgriff auf Wissen voraussetzt und damit dem *epistemischen* Bereich zuzuordnen ist.

Um die o. g. Fragestellung weiter zu präzisieren, haben wir für den ontologisch-epistemischen Bereich fünf weitere Leitfragen formuliert:

- Wo sind Objekte verankert, welcher Art ist das "Universum", welchem sie angehören? (*ontologischer Hintergrund*)
- Wie kommen Objekte zustande, wer ist für ihre Definition und Abgrenzung verantwortlich? (*Genese* von Objekten)
- Wie werden bestehende Objekte erkannt und mit Bezeichnungen versehen? (*Identifikation* von Objekten)
- Wie werden bestehende Objekte klassifiziert und aufgrund bestimmter Merkmale zu Kategorien oder Gruppen zusammengefasst? (*Typisierung* von Objekten)
- Wie sind Objekte miteinander verbunden und welche Wirkungen üben sie aufeinander aus? (*Beziehungen* zwischen Objekten)

# 3 Philosophische Grundpositionen

In der Philosophie haben sich in einer langen Tradition Denker verschiedener Denkschulen mit den genannten Fragestellungen beschäftigt und charakteristische Positionen herausgebildet. Dazu kommt die vor ca. 100-150 Jahren etablierte mathematische Logik mit ihrem eigenen Objektbegriff. In jüngster Zeit sind die "objektorientierten" Disziplinen der Informatik hinzugekommen, denen es allerdings bislang noch nicht gelungen ist, einen eindeutigen, klar definierten und allgemein anerkannten Objektbegriff herauszubilden.

Will man sich einem solchen Begriff nähern, so ist es nach unserer Überzeugung vorteilhaft, auf die philosophischen Grundlagen zurückzugreifen und Definitionen bzw. Begriffsklärungen im Einklang mit lange etablierten und bewährten Denktraditionen zu suchen statt (wie es in der Informatik oft geübte Praxis ist) diese zu ignorieren und sich in "Macher-Manier" allein von kurzfristigen Zweckmäßigkeits-Erwägungen oder gar kommerziellen Erfolgsaussichten leiten zu lassen. Wir plädieren deshalb für einen philosophisch begründeten Objektbegriff in der Informatik und wollen dies als eine erste These formulieren:

> These 1: Ein Objektbegriff in der Informatik (so es überhaupt *einen*, für alle Zweige verbindlichen solchen Begriff geben kann) sollte die Erkenntnisse und Denktraditionen der Philosophie berücksichtigen und kann von diesen nur profitieren.

Im folgenden werden die Positionen einiger philosophischer Denkschulen (einschließlich der mathematischen Logik) näher betrachtet und im Hinblick auf ihre Objekt- bzw. Gegenstandsbegriffe und deren (möglichen ) Einfluss auf die Informatik miteinander verglichen. Dabei wird versucht, jeden der betrachteten Ansätze durch eine - hier als "Paradigma" bezeichnete - Leit-Metapher kurz zu charakterisieren (vgl. Tabelle 1).

Die Grundaussage des ***Realismus*** besteht darin, dass es eine - außerhalb unseres Bewusstseins existierende und daher prinzipiell objektiv erfassbare - Welt gibt. John Searle drückt es folgendermaßen aus [16]:

> *The world (or alternatively, reality or the universe) exists independently of our representations of it. This view I will call "**external realism**".*

Dieses unabhängig von uns existierende "Universum" muss noch nicht unbedingt mit "vorgefertigten Objekten" bevölkert sein - in diesem Punkt unterscheiden sich gerade verschiedene Spielarten des Realismus. Nach Searle bildet es aber einen

*Hintergrund* für die Definition, Abgrenzung und Beschreibung von Objekten. Diese kann durchaus Beobachter-spezifisch erfolgen - allerdings sind bestimmte Eigenschaften - wie z.B. die Verortung bestimmter materieller Objekte im Universum *intrinsisch*, d. h., vom Beobachter unabhängig.

Diese Position, die z. B. auch im FRISCO-Bericht [6] aufscheint, wird zuweilen auch als *"God's Eye View"* charakterisiert: Der Beobachter versetzt sich gedanklich in eine außerhalb der Welt befindliche Warte oder nimmt sich ganz aus dieser heraus. Wir könnten das auch mit einem *"Garten Eden"* vergleichen: Der Mensch steht außerhalb, betrachtet die darin für ihn wahrnehmbaren "Früchte" und versucht diese mit den ihm zur Verfügung stehenden Mitteln zu erfassen und zu beschreiben.

In der Spielart des ***naiven Realismus*** wird nicht nur das als Hintergrund dienende Universum, sondern es werden auch die darin befindlichen Objekte (oder jedenfalls ein großer Teil davon) als "objektiv gegeben" (= vom Betrachter unabhängig) angesehen. Das Universum ist also bereits mit vorgefertigten "Objekten" bevölkert und diese müssen vom Betrachter nur in geeigneter Weise erkannt, aufgelesen und beschrieben werden. Eine "Erfassung" der uns umgebenden Welt ist dann nur eine Frage der Qualität unserer Repräsentationen.

Besonders treffend hat B. Meyer diese Position charakterisiert. Unter der Überschrift *"Finding the objects"* lesen wir [11, S. 51].:

> *"... object-oriented design is a natural approach: the world being modeled is made of objects - sensors, devices, airplanes, employees, paychecks, tax returns - and it is appropriate to organize the model around computer representations of these objects. This is why object-oriented designers usually do not spend their time in academic discussions of methods to find the objects****: in the physical or abstract reality being modelled, the objects are there just for the picking****! The software objects will simply reflect these external objects. "*

Selbst wenn wir Meyer zu Gute halten, dass er diese Passage heute anders schreiben würde, so halten wir die dort ausgedrückte (und im täglichen Leben kaum in Frage gestellte) Sicht auf ein "Schlaraffenland der Objekte" auch für die Mehrheit der Informatiker für symptomatisch: Die "Objekte" sind die gebratenen Tauben, die dem Betrachter in den Mund fliegen.

Eng verknüpft mit der naiv-realistischen Position ist die ***Abbildtheorie***. Das Gelingen unserer Erkenntnis lässt sich daran messen, wie genau es uns gelingt, die als objektiv gegebenen Dinge zu repräsentieren. Unsere Abbilder der Welt stellen also gewissermaßen ein Spiegelkabinett dar, in dem speziell die Software-Modellierer für immer neuere und bessere Spiegel sorgen.

Als Gegenposition zum Realismus gilt seit alters her (personifiziert durch Plato, in der Neuzeit vor allem durch Kant) der ***Idealismus***. Eine seiner Kernthesen (die der sog. *kopernikanischen Wende*) lautet (I. Kant, zit. n. [10])

> *"Die Erkenntnis richtet sich nicht nach den Gegenständen, sondern die Gegenstände nach der Erkenntnis."*

Eine solche Position negiert nicht die grundsätzliche Existenz von "Gegenständen" - relativiert sie jedoch: Gegenstände sind insoweit vorhanden und so beschaffen, wie wir sie erkennen (können und wollen). Josef Mitterer bringt die beiden Positionen folgendermaßen auf den Punkt [12]:

> *"Idealismus: Wir machen Unterschiede, daher gibt es sie.*
>
> *Realismus: Es gibt Unterschiede, daher machen wir sie."*

Das Ergebnis eines Erkenntnisprozesses könnte man jetzt mit dem Bild in einer Brille oder einem Fernrohr vergleichen: Es zeigt uns die Welt, wie wir sie erleben - aber möglicherweise verzerrt oder sogar gänzlich irreal. Für den Software-Entwickler bedeutet dieser Standpunkt zuerst einmal den Gewinn großer Freiheit, aber auch ein gehöriges Stück Mehrarbeit: Objekte sind keineswegs *a priori* gegeben und "nur zum Pflücken" da, sondern wollen im Laufe (möglicherweise schwieriger und langwieriger) Analyse- und Erkenntnisprozesse erarbeitet werden.

Neuere Spielarten des Realismus versuchen, die offensichtlichen Lücken und Schwächen seiner naiven Form und der Abbildtheorie zu überwinden , indem sie - statt eine externe Betrachtungsposition ("God's Eye View") einzunehmen - die Rolle des Betrachters und sein Involviert-Sein in den Erkenntnisprozess betonen. Für H. Putnam werden die Objekte vom Betrachter durch sein Handeln "geformt" [13]. In seiner Metapher vom *"cookie cutter"* vergleicht er den Hintergrund mit einem Teig - also einer zunächst amorphen, unstrukturierten Masse, aus der der Betrachter beim Erkenntnisprozess die Objekte wie Plätzchen heraussticht. Die "Förmchen", die er dabei benutzt, sind die im Laufe der Geschichte und der Sprachentwicklung geformten Begriffe, Kategorien und Normen.

So bestechend diese Metapher zunächst anmutet, so stößt auch sie (wie von Putnam selbst bemerkt) an

ihre Grenzen, und zwar, wenn man versucht, die Beschaffenheit des "Teigs" näher zu ergründen. Um aus ihm überhaupt etwas "ausstechen" zu können, muss man seine Zusammensetzung kennen, was wiederum Förmchen benötigen würde, die man ja erst gewinnen will. ...

Putnam folgert daraus, dass sich ontologische und epistemische Phänomene letztlich nicht trennen lassen und fordert, den Wahrheitsgehalt von Aussagen immer an den betrachteten Kontext zu binden. Seine Position bezeichnet er als ***internen Realismus***. Wir möchten sie in Anlehnung an die Metapher vom *cookie cutter* als "Töpferwerkstatt" bezeichnen: Objekte werden aus dem "Ton" der vorhandenen Kontexte geformt ("konstruiert") und je nach Bedarf für den Gebrauch "gebrannt".

Danach lässt sich z. B. die Frage, wie viele "Objekte" sich in einem Raum befänden, nicht eindeutig beantworten, da dies immer davon abhängt, was man als "Objekt" ansieht und was nicht. Für den System-Analysten bedeutet dies, dass er seine Systeme und Objekte nie unabhängig vom Kontext - von früheren oder zugrunde liegenden Systemen, Akteuren, Betroffenen etc. formen und definieren kann.

Auch im **Konstruktivismus** wird die zentrale Rolle des Betrachters betont und eine "externe" Position wird abgelehnt: wir sind immer "inmitten". Objekte werden durch unser Handeln - und dabei in erster Linie durch unser Sprachhandeln - geformt. Trotzdem sind "objektive" Aussagen über die Welt oder ihre Gegenstände nicht ausgeschlossen - sie sind vielmehr das Ergebnis von erfolgreichen Konsensprozessen: "Objektiv wahr" sind Aussagen dann, wenn eine qualifizierte Gruppe oder Mehrheit von Menschen darüber zum Konsens gelangt sind.

Dies stellt die Möglichkeit "letzter Wahrheiten" grundsätzlich in Frage - *de facto* jedoch zeigt die Erfahrung, dass es genügend "Tatsachen" gibt, über die im o. g. Sinne zweifelsfrei Konsens besteht - zumindest was den gegenwärtigen Erkenntnisstand der Menschheit betrifft.

Ist der interne Realismus eine Töpferwerkstatt, so haben wir es hier (wegen der Betonung des sprachlichen Handelns) mit einer *Dichterwerkstatt* zu tun. Für die Software-Entwicklung bedeutet das, dass (auch und gerade bei der "objektorientierten" Entwicklung) die Objekte zu Projektbeginn weder - wie bei B. Meyer - in Erntekörben bereitstehen noch - wie bei I. Jacobson [9] - aus den Anwendungsfällen (*use cases*) herauspurzeln, sondern in oft langwierigen und mühsamen Verhandlungsprozessen (z.B. zwischen Anwendern und Entwicklern) modelliert, analysiert und zum Konsens gebracht werden müssen.

## 4 Positionen der mathematischen Logik und der Informatik

Während alle genannten Ansätze bei der Beschreibung des Erkenntnisprozesses von der zu beschreibenden Realität (*"jenseits des Diskurses"*, vgl. [12]) ausgehen und dann zu Modellen (Repräsentationen, *"diesseits des Diskurses"*) gelangen, ist es beim Ansatz der ***mathematischen Logik*** umgekehrt. Sie stellt Mechanismen zur Verfügung, mit denen man Axiome und Schlussregeln formulieren und aus diesen formal Theoreme (Sätze) ableiten kann. Einem solchen formalen Axiomensystem kann man einen Sinn (eine "Semantik") unterlegen, indem man ein "Modell" mit einem "Individuenbereich" zugrunde legt und den Variablen des Systems Elemente dieses Bereichs als "Objekte" zuordnet. Ein Theorem ist dann z.B. "allgemeingültig", wenn es für alle Objekte jedes denkbaren Modells gültig ist.

Einen solchen Individuenbereich können wir uns wie eine *Briefmarkensammlung* vorstellen: Alles ist wohldefiniert, wohlgeordnet, katalogisiert und überschaubar - einen Bezug zur Realität herzustellen bleibt allerdings vollständig dem Betrachter überlassen.

Zum Erkenntnisprozess trägt ein solcher Ansatz offenbar direkt nichts bei. Indirekt gibt er dem Modellierer allerdings das Rüstzeug an die Hand, seine Axiomensysteme soweit zu entwickeln und zu verfeinern, bis sie geeignet sind, die (ihm relevant erscheinenden) Fakten und Zusammenhänge des betrachteten Weltausschnitts zu beschreiben und bei geeigneter Interpretation wahre Aussagen darüber zu liefern.

Was ein Objekt sein kann, entscheidet jetzt der Formalismus zu seiner Modellierung. W. Quine formuliert es so:

> *"To be is to be the value of a bound variable." (zit. n. P. Schefe, pers. Kommunikation: W.V. Quine: From a logical point of view , p. 103, Cambridge 1964)*

Putnam hält dem allerdings entgegen.

> *"But, from my "internal realist" perspective at least, there is no such totality as All The Objects There Are, inside or outside science. "Object" itself has many uses, and as we creatively invent new uses of words, we find that we can speak of "objects" that were not "values of any variable" in any language we previously spoke. ".*

Das bedeutet nichts weniger, als dass der "closed world"-Ansatz der Mathematik den Anforderungen einer offenen, dynamisch sich entwickelnden mensch-

lichen Erkenntnis- und Lebenswelt nicht vollständig genügen kann.

In der ***Informatik*** stoßen wir - wie oben bereits angesprochen - auf eine Vielzahl von sehr unterschiedlich definierten "Objekten". In diesem Zusammenhang hervorzuheben scheinen uns zwei Objektbegriffe - die der objektorientierten Analyse und Programmierung. Bei der (historisch älteren) ***objektorientierten Programmierung (OOP)*** haben wir es mit einem Objektbegriff zu tun, der dem der mathematischen Logik sehr ähnlich ist: Eine OO-Programmiersprache stellt Sprachkonstrukte wie *Klassen- und Objektdefinitionen, Attribute, Methoden* und *Beziehungen* zur Verfügung, mit denen man Programme schreiben kann. Diese simulieren bei geeigneter Anwendung und Interpretation Sachverhalte und Vorgänge in einem realen Weltausschnitt. Was ein "Objekt" ist oder sein kann, bestimmt die Programmiersprache [8]:

> *An object represents a component of the Smalltalk-80 software system. .. An object consists of some private memory and a set of operations.*

Eine Metapher für diesen Ansatz ist das *Sandkastenspiel*: Es repräsentiert eine geschlossene Welt und ist bei geeigneter Anlage und Interpretation durchaus nutzbringend zur Simulation eines realen Weltausschnitts einsetzbar, trägt aber zum Erkenntnisgewinn kaum bei - gibt allerdings auch nicht vor, dies es zu tun.

Ganz anders verhält es sich mit der ***objektorientierten Analyse (OOA)***. und ihrem Anspruch auf eine ganzheitliche, alle Phasen der Software-Entwicklung umfassende "Objektorientierung". Was könnten deren "durchgängige" Objekte sein? Materielle Objekte oder solche aus Fleisch und Blut wie der in den *use cases* betrachtete Kunde Mustermann? Ein dreigeteilter Kasten in einem UML-Diagramm? Ein Stück Java-Code? Ein ausführbares Programm oder Programmpaket? Alle diese Erscheinungen und Repräsentationen werden auf wundersame Weise unter einem gemeinsamen Objektbegriff zusammengefügt.

Bei zwei der bekanntesten Advokaten des OOA-Ansatzes liest sich das so:

> *Object: Something you can do things to. An object has state, behaviour, and identity; the structure and behaviour of similar objects are defined in their common class. The terms instance and object are interchangeable.* (G. Booch, [3])

> *.. Data is quantized into discrete, distinguishable entities called objects. Objects can be concrete, such as a file in a file system, or conceptual, such as a scheduling policy in a multiprocessing system. Each object has its own inherent identity.* (J. Rumbaugh et al. [14])

Es ist offenkundig, dass hier (reale oder abstrakte, oft nicht-sprachliche) Bezugsobjekte und ihre verschiedenen (sprachlichen) Repräsentationen in semiotisch unzulässiger Weise miteinander identifiziert werden - ein Spagat, der uns bewogen hat, die OOA mit einem *Sensationszirkus* zu vergleichen: Hier wird das Unmögliche möglich gemacht. Der Kunde Mustermann "besteht aus 17 Attributen" und wird auf Knopfdruck "deleted" - *requiescat in pacem*!

## 5 Fazit und Ausblick

Bei einer Zusammenschau der verschiedenen Ansätze (vgl. Tab. 1) zeigt sich also, dass der Objektbegriff der objektorientierten Programmierung im Wesentlichen dem der mathematischen Logik entspricht und damit relativ unproblematisch ist - wenngleich er zur Frage der Herkunft und Bildung von Objekten nichts beiträgt. Dagegen ist der in vielen OOA-Methoden erhobene Anspruch, die Objekte der (abzubildenden) Lebenswelt mit den sie repräsentierenden symbolischen Entitäten zu identifizieren semiotisch unzulässig - nutzt jedoch geschickt die sprachliche Identifikation von Wort und Bezeichnetem aus. Hier könnte ein sprachkritischer und im besonderen ein semiotischer Zugang für weitere Klärung sorgen.

Was die Herkunft und Genese von Objekten und damit letztlich die Qualität von "Objektmodellen" betrifft, kann die Informatik aus der philosophischen Debatte viel lernen: Vor allem, dass in der realen Projektwelt die Objekte in der Regel nicht wie reife Äpfel von den Bäumen fallen, sondern dass sie Ergebnisse eines sozialen Konstruktionsprozesses sind, unter den Beteiligten verhandelbar, einer Qualitätssicherung zu unterziehen und ggf. auch revidierbar sein müssen. Dies wollen wir als Schlussthese(n) zusammenfassen:

These 2: Ein tragfähiger Objektbegriff für die Informatik sollte

(1) semiotisch fundiert sein, d. h. Bezüge und verschiedene Repräsentationsformen für Objekte sauber trennen und

(2) die notwendigen Konstruktions- und Konsensbildungsprozesse bei der Genese und Qualitätssicherung von Objekten bzw. Objektmodellen angemessen berücksichtigen.

| Philosophische Position / Disziplin | Paradigma | Erklärung(en) |
|---|---|---|
| Naiver Realismus | Schlaraffenland | Objekte "sind da", "fliegen uns zu" |
| Externer Realismus | Garten Eden, "God's Eye View": | Objekte = Früchte in Nachbars Garten |
| Abbildtheorie | Spiegelkabinett: | Beobachter schafft "Abbild" der "Realität" |
| Interner Realismus | "coockie cutter", Töpferwerkstatt | Objekte werden vom Betrachter durch Handeln "geformt" [13] |
| Konstruktivismus | Dichterwerkstatt | Sprachhandlungen sind maßgeblich zur Objektbildung |
| Mathematische Logik | Briefmarkensammlung | Objekte sind Elemente eines formal definierten Individuenbereichs |
| Objektorientierte Programmierung (OOP) | Sandkasten | Objekte sind formale, symbolische Entitäten (siehe z.B. [8]) |
| Objektorientierte Analyse (OOA) | Sensationszirkus | Ontologische ("realweltliche") Objekte werden mit sprachlichen identifiziert |

**Tabelle 1: Philosophische Positionen und Objekt-Paradigmen**

**Dank**

Unser Dank geht an die weiteren Mitglieder des Arbeitskreises "Terminologie der Softwaretechnik" Urs Andelfinger, Hans-Bernd Kittlaus und Gerd Scheschonk sowie an Peter Schefe für viele und lange - aber nie langweilige -Diskussionen, die in diesen Beitrag eingeflossen sind.

**Literaturhinweise:**

[1] H. v. Braun, W. Hesse, U. Andelfinger, H.B. Kittlaus, G. Scheschonk.: Conceptions are social constructs - Towards a solid foundation of the FRISCO approach. In: E.D. Falkenberg, K. Lyttinen, A.A. Verrijn-Stuart (Eds.): Information System Concepts - An Integrated Discipline Emerging. Proc. ISCO 4 Conference, Kluwer Publ. Comp. 2000

[2] H. v. Braun, W. Hesse, H.B. Kittlaus, G. Scheschonk, G.: Ist die Welt objektorientiert? Von der natürlichsprachlichen Weltsicht zum OO-Modell; in: Natürlichsprachlicher Entwurf von Informationssystemen - Grundlagen, Methoden, Werkzeuge, Anwendungen; Proc. GI/EMISA-Workshop Tutzing, Schriften zur Informationswissenschaft Bd. 25, S. 280-295, Universitätsverlag Konstanz 1996

[3] G. Booch: Object-Oriented Analysis and Design with Applications; 2nd Edition, Benjamin/Cummings Publ. Comp. 1994

[4] P. Coad, E. Yourdon: Object-oriented analysis; 2nd ed., Yourdon Press 1991

[5] R. Ferber: Philosophische Grundbegriffe. Eine Einführung. Beck'sche Reihe 1054. C.H. Beck, München 2000

[6] E. Falkenberg, W. Hesse, P. Lindgreen, B.E. Nilsson, J.L.H. Oei, C. Rolland, R.K. Stamper, F.J.M. Van Assche, A.A. Verrijn-Stuart, K. Voss: FRISCO - A Framework of Information System Concepts - The FRISCO Report. IFIP WG 8.1 Task Group FRISCO. Web version: *ftp://ftp.leidenuniv.nl/pub/rul/fri-full.zip* (1998)

[7] Ch. Floyd, H. Züllighoven, R. Budde, R. Keil-Slawik: Software Development and Reality Construction, Springer-Verlag 1992

[8] A. Goldberg, D. Robson: Smalltalk 80: The language and its implementation; Addison-Wesley 1983

[9] I. Jacobson: Object-Oriented Software Engineering - A Use Case Driven Approach; Revised Printing, Addison-Wesley 1993

[10] P. Kunzmann, F.-P. Burkard, F. Wiedmann: dtv-atlas zur Philosophie. dtv 1991

[11] B. Meyer: Object oriented software construction, Prentice Hall 1988

[12] J. Mitterer: Die Flucht aus der Beliebigkeit. Fischer, Frankfurt am Main 2001

[13] H. Putnam: Representation and Reality. Sixth Printing, MIT Press, Cambridge, Mass. 1998; dt. Ausgabe: Repräsentation und Realität. Suhrkamp Taschenbuch, Frankfurt/M. 1999

[14] J. Rumbaugh, M. Blaha, W. Premerlani, F.Eddy, W. Lorensen: Object oriented modelling and design. Prentice Hall 1993

[15] P. Schefe, Softwaretechnik und Erkenntnistheorie. In: Informatik-Spektrum 22, S. 122-135.

[16] John Searle: The Construction of Social Reality, Penguin Books, London 1995