Katrin Weller

B 6 Ontologien

B 6.1 Grundlagen und Definitionen

Ontologien sind formale, schematische Abbildungen eines Wissensbereichs, bestehend aus einem Vokabular und Regeln zu seiner Zusammensetzung. Sie finden als Wissensmodelle und als inhaltsbeschreibende Metadaten in verschiedenen (Internet-)Diensten Verwendung und etablieren sich damit zunehmend als eine zusätzliche Methode der Wissensorganisation (vgl. B 3 Wissensorganisation), die neben klassischen Methoden (wie Thesauri und Klassifikationssystemen) zunehmend zum Einsatz kommt (Lit. 01, Lit. 02). Eine Besonderheit von Ontologien im Vergleich mit klassischen Ansätzen besteht darin, dass sie und ihr Einsatz eng mit der Entwicklung technischer Standards verbunden sind. Dies hängt mit ihrem primären Einsatzzweck zusammen: Ontologien werden mit Hilfe von maschinenlesbaren Ontologiesprachen verfasst, damit Computersysteme die enthaltenen Informationen automatisch auslesen und möglichst auch interpretieren und daraus Schlussfolgerungen ziehen können. Ontologien als Wissensorganisationsmethode müssen daher immer in Bezug zu den vorhandenen Ontologiesprachen betrachtet werden: die inhaltliche Abbildung eines Wissensbereich ist die eine Seite, ihre Realisierung mit Hilfe einer speziellen Ontologiesprache die andere.

Im Folgenden soll die Ontologie als Wissensrepräsentationsmethode anhand ihrer charakteristischen und wichtigsten Merkmale sowie mit Hilfe von Beispielen erklärt werden. Dieses Kapitel kann jedoch nicht die kompletten Spezifika einer ganzen Ontologiesprache (oder gar mehrerer Sprachen) vorstellen. Wer sich mit einer speziellen Ontologiesprache umfassend vertraut machen möchte, sei daher auf die entsprechende Literatur verwiesen (z. B. Lit. 03, Lit. 04).

B 6.1.1 Ein erstes Beispiel

Die typischen Bestandteile einer Ontologie sind (hierarchisch geordnete) Konzepte (*concepts),* die meist in Form von Klassen (*classes*) umgesetzt und definiert werden, Instanzen (*instances*, *individuals*) als konkrete Elemente dieser Klassen, und Relationen (*properties*) als Beschreibung der Eigenschaften von und Beziehungen zwischen Konzepten. Die Struktur einer Ontologie kann außerdem durch Axiome (*restrictions*) modelliert werden. Diese Ontologiebestandteile können von den gängigen Ontologiesprachen formal erfasst werden, mitunter werden dabei auch andere Bezeichnungen für die einzelnen Elemente verwendet. Auch die genauen Erscheinungsformen (z. B. welche Möglichkeiten zur formalen Definition verfügbar sind) können je nach Ontologiesprache variieren. Im diesem ersten illustrativen Beispiel sowie in der folgenden vertiefenden Betrachtung wird im Wesentlichen die Kapazität und Terminologie der Ontologiesprache OWL zu Grunde gelegt (wenngleich es sich nicht um eine komplette Einführung in OWL handelt). Meist geben Ontologiesprachen auch vor, welche Zeichen für Benennungen innerhalb einer Ontologie zulässig sind. Oft sind beispielsweise keine Leerzeichen zwischen Mehrwortbegriffen möglich, so dass diese (wie auch in den hier verwendeten Beispielen) durch Unterstriche ersetzt oder ausgelassen und durch nachfolgende Großbuchstaben (*CamelCase*) gekennzeichnet werden.

Abbildung 1 zeigt eine schematische und vereinfachte Darstellung der typischen Ontologieelemente und ihrer Zusammenhänge für einen Teilbereich der Domäne *Sport*.

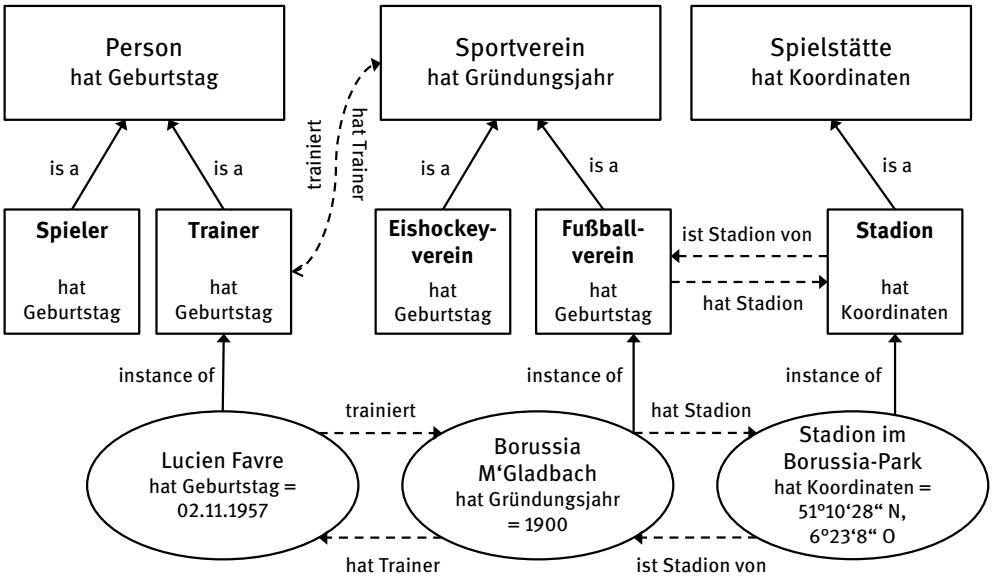

Abb. 1: Schematische Darstellung eines Ausschnitts einer Ontologie zum Thema Sport, bestehend aus Klassen und Klassenhierarchien, Instanzen, Object Properties und Datatype Properties.

Die eckigen Felder stellen Ontologieklassen dar. Klassen repräsentieren Allgemeinbegriffe, also Konzepte eines Interessensgebiets, die reale Objekte anhand von gemeinsamen Eigenschaften bündeln sollen. Sie sind meist grundlegend hierarchisch strukturiert. Im Beispiel sind die Klassen *Spieler* und *Trainer* über die *is_a* Relation (durchgängige Pfeile in der Abbildung) als Unterklassen mit der Klasse *Person* verbunden.

Eng verwandt mit der *is_a* Beziehung ist die *instance_of* Relation. Sie verbindet *Instanzen* (in der Abbildung als ovale Felder dargestellt) mit Klassen. Instanzen repräsentieren Individualbegriffe, also konkrete Vertreter der einzelnen Klassen. In der Abbildung sehen wir als Beispiele *LucienFavre* als Instanz der Klasse *Trainer* und *BorussiaMönchengladbach* als Instanz der Klasse *Fußballverein*.

Klassen sowie deren Instanzen können weiter in ihrer Bedeutung spezifiziert werden. Dies erfolgt durch sogenannte *Properties*, welche die Klasseneigenschaften über semantische Relationen modellieren. Dabei gibt es zwei grundlegende Möglichkeiten in der Umsetzung: Zum einen kann eine Property die Beziehung zwischen zwei Klassen herstellen. In Abb. 1 stehen gestrichelte Pfeile für solche Beziehungen zwischen Klassen, ein Beispiel ist die Verbindung zwischen den Klassen *Trainer* und *Verein* über die Property *hat_Trainer* bzw. *trainiert*.

Zum anderen können Properties aber auch einzelne Klassen beschreiben, ohne diese mit anderen Klassen zu verbinden. Beispiele hierfür finden sich in der Abbildung innerhalb der Felder *Person*, *Sportverein* und *Sportstätte*. Eine Property wie *hat_Geburtstag* kann dann z. B. einfach mit einer Zahl im Datumsformat versehen werden, und muss nicht mit anderen Elementen der Ontologie verbunden werden. Bei beiden Formen von Properties ist entscheidend, dass sie zunächst auf der Klassenebene formuliert werden, um so die spezielle Klasse näher zu definieren: So ist z. B. ein *Trainer* eine spezielle Form von *Person*, die sich dadurch definieren lässt, dass sie in einer speziellen Beziehung zur Klasse *Sportverein* steht. Zusätzlich erbt die Klasse *Trainer* alle Eigenschaften der Oberklasse *Person*, also. z. B. die Eigenschaft *hat_Geburtstag*. Auf Ebene der Instanzen können diese Relationen dann mit spezifischen Werten versehen werden: bei der Instanz *LucienFavre* können wir ein konkretes Geburtsdatum eintragen sowie eine Verbindung zu der Instanz *BorussiaMönchengladbach* herstellen, dem Verein, bei dem Favre (zurzeit) Trainer ist.

Auf diese Weise werden Fakten aus dem jeweiligen Themengebiet direkt in der Ontologie gespeichert. Die Ontologie ist an dieser Stelle nicht mehr nur reines Indexierungsvokabular, sondern wird selbst zu einer Wissensbasis. Die Instanzebene wird daher oft separat betrachtet und nicht immer als Bestandteil zur Ontologie hinzugezählt.

Nicht in der Abbildung dargestellt ist der Einsatz von *Restrictions*. Er lässt sich dennoch anhand dieser Beispiel-Ontologie bereits skizzieren: Restrictions (oder Axiome) sind Regeln, mit deren Hilfe die Klassen formal definiert werden können. Sie spezifizieren, in welcher Form die Properties angewandt werden sollen. Beispielsweise kann festgelegt werden, dass ein *Trainer* immer eine Beziehung *trainiert* zu einem *Verein* haben muss, damit er tatsächlich als Trainer gelten kann. Umgekehrt kann dann unter gewissen Voraussetzungen (s. u.) geschlossen werden, dass eine *Person* die in der Relation *trainiert* zu einem *Verein* steht, auch in die Klasse *Trainer* eingestuft werden muss.

Die schematische Darstellung aus Abbildung 1 soll nur als sehr vereinfachtes Beispiel zur Einführung der wesentlichen Ontologiebestandteile dienen. Ausführlichere Darstellungen von Muster-Ontologien gibt es in der Literatur z. B. zum Thema *Pizza* (Lit. 05), zum Thema *Wein* (Lit. 06) und zum Thema *Harry Potter* (Lit. 07). In Abschnitt B 6.2 sollen die Ontologie-Elemente noch einmal in Hinblick auf ihre Besonderheiten und damit verbundenen Herausforderungen beim praktischen Aufbau von Ontologien vertiefend erläutert werden.

B 6.1.2 Hintergrund und Einsatzbereich

Die Informatik hat die Bezeichnung *Ontologie* aus der Philosophie entlehnt und ihr eine neue Bedeutung zugewiesen. Ontologie leitet sich ab aus dem griechischen Ontologia, was in etwa mit *Lehre vom Sein* übersetzt werden kann. Als Ontologia/Ontologie bezeichnet man etwa seit dem siebzehnten Jahrhundert eine philosophische Disziplin, die sich mit den Grundstrukturen und dem Wesen allen Seins befasst – damit wird Ontologie in der Philosophie heute weitgehend gleichbedeutend mit *allgemeiner Metaphysik* verwendet.

Im Rahmen der Informatik wurde der Term Ontologie aufgegriffen und mit einer eigenständigen Bedeutung versehen (z. B. Lit. 08). In englischen Texten lässt sich sprachlich gut erkennen, welcher Kontext gemeint ist: Die philosophische Disziplin wird in der Regel mit Großbuchstaben geschrieben und nur im Singular verwendet (*Ontology*), bei den Wissensmodellen der Informatik handelt es sich um einen zählbaren Begriff. Pluralformen sowie Kleinschreibung sind üblich (*ontologies*, *one* ontology). Allgemein versteht man unter einer Ontologie außerhalb der Philosophie eine formale Konzeptualisierung eines Wissensbereichs. Auch hier geht es darum, wesentliche Entitäten der Realität zu erfassen und zu definieren und ihre Beziehungen untereinander zu beschreiben. Die wohl berühmteste Definition von Ontologie im Kontext der Informatik lautet „an ontology is an explicit specification of a conceptualization“ (Lit. 09). Zahlreiche weitere Begriffserklärungen wurden und werden in der Literatur zitiert, eine endgültige und eindeutige Definition, die von allen beteiligten Forschungsdisziplinen geteilt wird, gibt es dabei allerdings bis heute nicht. So fällt es anhand der meisten bisherigen Definitionen schwer, Ontologien von anderen Methoden der Wissensorganisation klar abzugrenzen (Lit. 10). Hinzu kommt die Schwierigkeit, dass die genaue Form der Ontologie auch stark von der verwendeten Ontologiesprache abhängt.

Ontologien als Methode zur Wissensorganisation werden in der Informatik etwa seit Anfang der 1990er Jahre diskutiert und entwickelt. Seit mit der Arbeit an einem Semantic Web (vgl. B 7 Semantic Web und Linked Open Data) begonnen wurde, erhielten sie einen besonderen Stellenwert. Die Idee einer zusätzlichen Metadatenebene, durch die das World Wide Web zu einem Semantic Web ausgebaut werden sollte, wurde durch eine Veröffentlichung von Tim Berners-Lee, James Hendler und Ora Lassila im Jahre 2001 maßgeblich vorangetrieben (Lit. 11). Das Semantic Web stellt demzufolge eine Erweiterung des klassischen Webs dar, bei der eine zusätzliche Schicht semantischer Informationen zu bestehenden Web-Inhalten hinzugefügt wird. Dabei geht es darum, Daten im Internet so aufzubereiten, dass auch Computer in der Lage sind, einzelne Informationen heraus-

zufiltern, weiterzuverarbeiten und mit anderen neu zu kombinieren. Im Unterschied zur Künstlichen-Intelligenz-Forschung soll hier jedoch dem Computer nicht beigebracht werden, vorhandene Informationen selbst zu *verstehen*, sondern es sollen spezielle Metadaten angewandt werden, mit deren Hilfe kleine Informationseinheiten beschreibbar und damit interpretierbar gemacht werden (Lit. 12). Am Ende steht das Ziel, neue Internetdienste für praktische Anwendungsszenarien zu entwickeln, die beispielsweise Informationen aus mehreren Quellen sammeln und sinnvoll zusammenstellen können – und zwar idealerweise unabhängig von der speziellen Terminologie einzelner Quellen. Gelingen soll dies mit Hilfe von Wissensmodellen, die dem Computer ein umfassendes Hintergrundwissen eines Fachgebietes, bestehend aus Fachvokabular und Beziehungen zwischen den einzelnen Termen, zur Verfügung stellen: Ontologien. Damit dies gelingt, müssen Wissensbereiche vorab in einer Ontologie definiert und formal festgehalten werden. Wichtig ist dabei, dass einerseits die Terminologie und die Zusammenhänge des Fachgebietes sorgfältig aufgearbeitet werden, und dass andererseits diese Informationen in einem speziellen Datenformat abgespeichert werden, mit dem eine zugehörige Suchmaschine umgehen kann. In der Semantic-Web-Forschung arbeitet man intensiv an den technischen Grundlagen, die hierfür notwendig sind, und entwickelt beispielsweise Ontologiesprachen und dazugehörige Ontologieeditoren, Abfragesprachen sowie *Reasoner*, die automatische Schlussfolgerungen basierend auf den Informationen der Ontologie ziehen können.

In der Regel geht man inzwischen davon aus, dass es nicht eine allumfassende Ontologie geben soll, die alle Themenbereiche des WWW abdeckt und so für ein flächendeckendes Semantic Web eingesetzt werden kann. Stattdessen sollen einzelne Themenbereiche und spezielle Fachgebiete mit jeweils eigenen, klar eingegrenzten Ontologien repräsentiert werden, die dann vorwiegend in konkreten Spezialanwendungen (*semantic applications*) eingesetzt werden. Experten entwickeln also maßgeschneiderte Ontologien für spezielle Anwendungsfälle, Beispiele werden in Abschnitt B 6.3 vorgestellt. Ontologien im Sinne des Semantic Web sind also eine Spezialform eines kontrollierten Vokabulars, bei dem Begriffsdefinitionen und -relationen mit maschinenlesbarer Formalisierung kombiniert werden und die für einen speziellen Anwendungsfall erstellt werden.

B 6.1.3 Technische Grundlagen

Die Tatsache, dass eine Ontologie so eng mit ihrer tatsächlichen Darstellung in einer speziellen Ontologiesprache verbunden ist, führt also dazu, dass eine einheitliche Definition von Ontologien und ihren Bestandteilen erschwert wird. Ontologiesprachen ermöglichen eine maschinenlesbare Darstellung der Elemente einer Ontologie – geben aber ihrerseits auch vor, welchen Gestaltungsspielraum man bei der Vokabularkontrolle und der formalen Begriffsdefinition hat. Die ersten Ontologien in der Informatik basierten in ihrer Darstellung auf Frames und *First-Order Logic* (Lit. 13). Nach und nach wurden verschiedene Sprachen zur Wissensrepräsentation entwickelt, die auf Beschreibungslogik (*Description Logics*) basierten, wie beispielsweise OIL und DAML+OIL. Durchgesetzt hat sich schließlich die *Web Ontology Language (OWL)* (Lit. 03), die auch vom World Wide Web Consortium (W3C) als Standard empfohlen wird. OWL gibt es in drei Ausführungen von unterschiedlicher Ausdrucksstärke: OWL lite, OWL-DL (basierend auf Description Logics) und OWL full, sowie in der erweiterten Version OWL 2 (Lit. 14). Der aktuelle Trend geht jedoch eher zu den weniger ausdrucksstarken Varianten – und wenn selbst OWL lite noch zu komplex für eine geplante Anwendung erscheint, wird aktuell meist auf das einfachere RDF(S) (Lit. 15) oder auf SKOS (Lit. 04) zurückgreifen. Einen Überblick über wichtige Sprachen gibt es z. B. bei Lit. 13, Lit. 16.

Klassen, Instanzen und Properties werden in Ontologiesprachen wie OWL oder RDF mit einer eindeutigen Kennung versehen: die URI (*universal resource identifier*) sorgt dafür, dass jedes Ontologieelement eindeutig referenzierbar ist. Beim Anlegen einer neuen Ontologie, erhält diese zunächst ihren eigenen URI-Bereich (den *namespace*), auf dem die URIs für die einzelnen Elemente

aufbauen. Ein Namespace könnte z. B. die Form *http://my.domain.de/SportOntologie* haben, eine Klasse *Sportverein* hätte dann die folgende URI: *http://my.domain.de/SportOntologie#Sportverein*.

Beim Speicherprozess wird die Ontologie als eine Zusammenstellung vieler sogenannter Triple *(triples)* angesehen, also als dreistellige Informationseinheiten, bestehend aus Subjekt, Prädikat und Objekt (Lit. 12). Basierend auf dem Muster aus Abbildung 1 wäre ein solches Triple zum Beispiel: *Trainer* (Subjekt) – *trainiert* (Prädikat) – Sportverein (Objekt).

Beim Aufbau von Ontologien helfen Ontologieeditoren (*ontology editors*), so dass man nicht die komplette Ontologiesprache im Kopf haben muss, um eine Ontologie zu formulieren. Die Prinzipien der Ontologiemodellierung sollte man dennoch kennen, denn die aktuellen Ontologieeditoren richten sich an Nutzer mit Vorkenntnissen.

Ontologieeditoren unterscheiden sich zum einen dadurch voneinander, welche Ontologiesprachen sie unterstützen. Davon hängt dann wiederum auch der Funktionsumfang bei der eigentlichen Ontologieerstellung ab. Zum anderen sind jeweils folgende Punkte zu beachten: Handelt es sich um ein freies oder kostenpflichtiges Programm; wird der Editor lokal installiert oder läuft er Web-basiert; kann man mit mehreren Nutzern an der gleichen Ontologie arbeiten; gibt es erweiterte Projektmanagement-Funktionen (z. B. Versionskontrolle, Kommentierung von Arbeitsschritten, Task Management) und können zusätzliche Plug-Ins (z. B. zur Visualisierung der Ontologie) eingebunden werden? Der wohl am weitesten verbreitete kostenfreie Ontologieeditor ist Protégé (URL1). Zu den etablierten kostenpflichtigen Angeboten gehören unter anderem TopBraid Composer und OntoStudio.

Abfragesprachen (*query languages*) können eingesetzt werden, um alle Elemente einer Ontologie zu ermitteln, die bestimmte Kriterien erfüllen und nutzen dafür besonders die Triple-Struktur von Ontologien für die Suche aus. Für RDF-basierte Ontologien wird zumeist SPARQL als Abfragesprache eingesetzt (Lit. 17), für OWL gibt es auch spezielle Sprachen wie OWL-QL (Lit. 18). Die Hauptaufgabe von Reasonern (*reasoners*, auch *inference engines*) war ursprünglich nicht die Suche nach Informationen, sondern nach Unstimmigkeiten in der Ontologie. Sie können aber nicht nur Modellierungsfehler und logische Inkonsistenzen aufspüren, sondern auch Schlussfolgerungen ziehen und so Informationen bereitstellen, die (noch) nicht explizit in die Ontologie eingegeben wurden (Lit. 13). Beispielsweise können Instanzen unter bestimmten Voraussetzungen automatisch in Klassen eingruppiert werden (s. u.).

B 6.2 Struktur und Bestandteile von Ontologien im Detail

B 6.2.1 Das Grundgerüst: Klassen

Konzepte können als grundlegende Elemente der Ontologie angesehen werden, bei denen alle übrigen Elemente anknüpfen. Meist werden sie in Form von Klassen und Unterklassen hierarchisch strukturiert und stellen so die Basis des Ontologievokabulars dar. Die Bezeichnungen ‚Klasse' und ‚Konzept' werden beim praktischen Ontologieaufbau oft synonym verwendet. Dabei ist ein Konzept im eigentlichen Sinne die Abstraktion einer Menge von Objekten anhand von gemeinsamen, charakteristischen Eigenschaften, die durch eine (Klassen-)Bezeichnung repräsentiert wird. Konzepte können nicht nur physikalische Objekte erfassen, sondern reichen vom Abstrakten (etwa *Raum, Zeit, Wissenschaft*) bis zum Spezifischen (etwa *Riesenhonigbiene*). Ontologieklassen sollten möglichst eindeutig definiert werden. Dies kann einerseits über eine Aufzählung aller Elemente (extensionale Begriffsbestimmung, Lit. 01) geschehen: die Klasse *Wochentage* könnte so über ihre Bestandteile *Montag*, *Dienstag*, *Mittwoch*, *Donnerstag*, *Freitag*, *Samstag*, *Sonntag* definiert werden. In den meisten Fällen ist eine solche Aufzählung jedoch wenig sinnvoll, da die Menge der Bestandteil zu groß oder nicht konstant ist.

Stattdessen kann eine Klasse auch anhand ihrer charakteristischen Merkmale definiert werden (intensionale Begriffsbestimmung, Lit. 01). Die Klasse *DeutscheFußballmeister* könnte so als

eine Menge von *Fußballvereinen* beschrieben werden, die in Deutschland aktiv sind, und die mindestens einmal die deutsche Fußballmeisterschaft gewonnen haben. Beide Definitions-Varianten können mit OWL formal umgesetzt werden, so dass das entsprechende Wissen vom Computer ausgelesen werden kann.

Idealerweise teilt eine Unterklasse alle charakteristischen Merkmale ihrer Oberklasse und wird durch mindestens ein weiteres Merkmal spezifiziert. Beim praktischen Aufbau von Ontologien stößt man hier auf ähnliche Herausforderungen wie auch in anderen Wissensordnungen: Meist bieten sich mehrere Möglichkeiten für die hierarchische Strukturierung von Objekten in Klassen an, es muss eine sinnvolle sowie pragmatische Wahl bezüglich der Anordnung getroffen werden. In der Regel ist gewünscht, dass die Ontologie monohierarchisch strukturiert wird, dass also eine Klasse nicht mehrere Oberklassen hat. Formal lassen sich Klassen als disjunkt definieren. Das bedeutet, dass diese Klassen keine gemeinsamen Elemente haben können, dass also eine Instanz nicht beiden Klassen gleichzeitig zugewiesen werden darf.

Die Beziehungen zwischen Ober- und Unterklassen werden von Ontologiesprachen in der Regel standardmäßig unterstützt, und beispielsweise durch Relationen wie *subClassOf* oder *is_a* repräsentiert. Wenn die Beziehung zwischen Ober- und Unterklasse ausschließlich von der Art *is_a* (oder sinngemäß ‚ist eine Art von') ist, gilt es, dies für den praktischen Ontologieaufbau besonders zu beachten, da in diesem Fall nur Abstraktionsrelationen für die hierarchische Struktur eingesetzt werden können. Klassenbezeichnungen müssen also entsprechend so gewählt werden, dass diese Beziehung nicht verletzt wird (also etwa nicht: *Deutschland* als Unterklasse von *Europa*, da dies bedeuten würde, dass Deutschland „eine Art" Europa ist). Für alle weiteren Relationen zwischen Klassen, insbesondere auch für die Bestandsrelation (Teil-Ganzes-Relation) wird dann mit Properties gearbeitet.

B 6.2.2 Verbinden und beschreiben: Properties

Besonders für die Klassen-Definition anhand von charakteristischen Merkmalen ist es zunächst erforderlich, dass die Eigenschaften von Konzepten auch in der Ontologie erfasst werden können. Hierzu kommen semantische Relationen ins Spiel. In Ontologiesprachen wie OWL werden die Beziehungen zwischen Konzepten und damit auch die Eigenschaften von Klassen mit Properties erfasst. In klassischen Wissensorganisationsmethoden spricht man in der Regel von (semantischen) Relationen, wenn die Begriffsbeziehungen zwischen verschiedenen Konzepten gemeint sind (etwa Synonymie, Assoziationsrelation, Lit. 19), deswegen wollen wir diese Bezeichnung auch hier anwenden. Properties in Ontologien sind jedoch deutlich umfangreicher, als die Relationen, die standardmäßig in Thesauri oder Klassifikationssystemen zum Einsatz kommen. Es gibt keinerlei Richtlinien, die vorgeben, welche Properties in welcher Form genutzt werden dürfen. Das bedeutet, der Ontologiedesigner kann hier völlig frei entscheiden.

So können Klassen in Ontologien über beliebige Relationen miteinander verbunden werden. Nehmen wir an, die vorhandenen Klassen *Fußballspieler* und *Fußballverein* sollen verbunden werden. In einem Thesaurus würde hier vermutlich eine Assoziationsrelation gewählt. In der Ontologie ist die Beziehung frei benennbar und spezifischer. Sie könnte also etwa lauten *Fußballspieler – spielt_für – Fußballverein*, oder auch *Fußballspieler – ist_Teil_von* / *ist_Mitglied_in* / *gehört_zu* / *hat – Fußballverein*. In der Regel würde es zudem ergänzend eine *inverse* Property geben, die den gleichen Zusammenhang aus Sicht des Konzepts *Fußballverein* darstellt, etwa: *Fußballverein – hat_Mitglied – Fußballspieler*.

Bislang gibt es keine einheitlichen Standards, wie Relationen, die typischerweise und regelmäßig in Wissensordnungen auftreten, in Ontologien einheitlich benannt werden sollen. Davon betroffen ist beispielsweise die Teil-Ganzes-Relation (*Meronymie*), für die es beim Thesaurusaufbau sehr wohl standardisierte Ausdrücke gibt. Das bedeutet, dass beim Zusammenspiel von mehreren verschiedenen Ontologien auch immer eine Übersetzung zwischen den verschiedenen Properties

berücksichtigt werden sollte (etwa: die Property *hat_Bestandteil* in Ontologie 1 enstpricht der Property *besteht_aus* in Ontologie 2). Im Gegenzug bietet die Ontologie hier flexiblen Spielraum für die präzise Beschreibung und Modellierung von Wissen, dass je nach Anwendungsfall und Wissensdomäne konkretisiert dargestellt werden kann. In der Praxis ebenfalls wichtig ist die Möglichkeit, mit Properties auch Beziehungen zwischen Konzepten aus verschiedenen Ontologien miteinander zu verbinden – und somit Konkordanzen herzustellen.

Wie oben bereits kurz angesprochen heißen die (wechselseiteigen bzw. inversen) Relationen zwischen Ontologieklassen (in OWL) *Object Properties*. Daneben gibt es (ebenfalls in OWL) weitere Möglichkeiten, Eigenschaften von Klassen mit Properties zu erfassen. *Datatype Properties* verbinden eine Klasse nicht mit einer anderen, sondern weisen ihr einen Wert, etwa in Form einer Zahl oder eines Textstrings zu. Bleiben wir bei der Beispiel-Klasse *Fußballspieler*, so könnten zur Beschreibung der Eigenschaften Datatype Properties wie *hat_Geburtsjahr* oder *hat_Trikotnummer* (mit Möglichkeit, einen Zahlenwert anzugeben) oder *ist_Nationalspieler* (mit der Möglichkeit, diese Information als wahr oder falsch einzustufen) hinzukommen. Wegen der triple-förmigen Struktur sind die Relationen innerhalb der Ontologie immer nur zweistellig, das heißt, dass immer nur genau zwei Klassen über eine Property verbunden werden können (oder eine Klasse und ein Wert bei Datatype Properties). Um Aussagen zu modellieren, die sich gleichzeitig auf drei Klassen oder Werte beziehen, müssen daher spezielle Notlösungen gefunden werden. Problematisch wird es zum Beispiel bei der Umsetzung des Zusammenhangs ‚*Fußballverein* gewinnt *Wettbewerb* in *Jahr*'.

Auf der Klassenebene benennen Properties zunächst allgemein die Eigenschaften der jeweiligen Klasse. Sie können beispielsweise besagen, dass sich die Mitglieder der Klasse *Fußballspieler* durch Eigenschaften wie Vereinszugehörigkeit oder Trikotnummern auszeichnen, die Property *hat_Geburtsjahr* könnte die Klasse *Fußballspieler* von einer Oberklasse *Person* geerbt haben. Konkrete Werte – und damit Faktenwissen – werden erst später auf der Instanzebene relevant.

Klassen und Properties stellen das kontrollierte Vokabular bzw. die Terminologie der Ontologie bereit. Man spricht hier auch von der TBox (*Terminology Box*), welche das Wissen über das Vokabular und die Zusammenhänge der jeweiligen Domäne enthält. In Abgrenzung dazu enthält die ABox (*Assertion Box*) Faktenwissen auf Ebene der Instanzen. Nicht alle Ontologiesprachen unterscheiden explizit zwischen diesen beiden Ebenen.

B 6.2.3 Die Wissensbasis: Instanzen

Instanzen sind konkrete Vertreter einer Klasse, keine Unterklassen, und werden teilweise auch *Individuals* genannt. Sie stehen in der Regel zu den zugehörigen Klassen über eine spezifische Relation in Verbindung, die von der jeweiligen Ontologiesprache vorgegeben wird und nicht frei gewählt werden kann, etwa *instance_of*. Instanzen verfügen über genau die Eigenschaften der Klasse, zu der sie gehören, zusätzliche Properties können auf Instanzebene nicht hinzugefügt werden. Beim praktischen Ontologieaufbau ist hier – ähnlich wie bei der Modellierung von Unterklassen – Vorsicht geboten: Instanzen stehen nicht in einer Teil-Ganzes-Beziehung zu der jeweiligen Klasse sondern sind immer konkrete Vertreter der Klasse mit den entsprechenden klassendefinierenden Eigenschaften. Die Instanzen *MiroslavKlose*, *PhilippLahm* und *LukasPodolski* sollten demnach nicht als Instanzen der Klasse *FußballNationalmannschaft* angelegt werden, da ein einzelner Spieler als Instanz nicht die Eigenschaften einer Klasse *Mannschaft* teilt (etwa: *hat_Mitglieder*). Vielmehr könnten diese Instanzen einer Klasse *Nationalspieler* oder *Fußballspieler* (z. B. als Unterklasse zu *Spieler*) zugeteilt werden, welche dann mit der Klasse *FußballNationalmannschaft* über eine Property verbunden werden kann.

Für die einzelnen Properties werden jetzt auf Ebene der Instanzen konkrete Werte angegeben: *NorbertMeier* (Instanz der Klasse *Trainer*) – *trainiert* (Property der Klasse *Trainer*) – *FortunaDüsseldorf* (Instanz der Klasse *Fußballverein*, die als Unterklasse von *Sportverein* über die *trainiert*-Relation mit der Klasse *Trainer* verbunden ist). Oder *LukasPodolski* (Instanz von *Fußballspieler*) – *ist_*

Mitglied_in (Property) – *DeutscheFußballNationalmannschaft* (Instanz von *FußballNationalmannschaft*).

Auf diese Weise reichert die Instanzebene eine Ontologie um konkretes Faktenwissen an (ABox) und lässt damit die Grenze zwischen abstraktem Wissensorganisationsmodell und konkreter Wissensbasis verschwimmen. Mitunter wird daher auch die Instanzebene nicht zur Ontologie dazugezählt, sondern als eigenständiges Konstrukt wahrgenommen. Das liegt auch daran, dass nur die TBox der Ontologie Anspruch auf Allgemeingültigkeit erhebt. Auf unser Beispiel bezogen bedeutet das: Fußballvereine bleiben immer eine Art von Verein und stehen immer in einer bestimmten Beziehung zu Trainern. Bei der ABox kann es hingegen problematisch werden, wenn Fakten auf der Instanzebene nicht permanent gültig sind – und dies ist in unserer gewählten Beispieldomäne häufig der Fall: Ein Verein bekommt immer wieder andere Trainer, Spieler wechseln den Verein oder beenden die Karriere. Mitunter kann dieses Problem mit der Angabe von Jahreszahlen gelöst werden. So könnte ein Beispiel lauten: *LukasPodolski – ist_Mitglied_in – DeutscheFußballNationalmannschaft2004, DeutscheFußballNationalmannschaft2005, ..., DeutscheFußballNationalmannschaft2012*.

B 6.2.4 Definieren und Schlussfolgern: Formale und informale Semantik

Es ist wünschenswert, dass innerhalb einer Ontologie alle Bestandteile formal (also für den Computer interpretierbar) und zusätzlich informal (für Menschen verständlich) definiert werden. Die informale Definitionsebene wird in der Regel mit Hilfe von Textfeldern realisiert, die es erlauben, für jedes Konzept, jede Instanz und jede Property eine natürlichsprachige Beschreibung hinzuzufügen (in OWL über *rdfs:comment*). Diese Beschreibungen sind während des Ontologieaufbaus nützlich, um den Ontologiedesignern untereinander ein gemeinsames Verständnis zu vermitteln, helfen aber auch dem späteren Nutzer einer Ontologie bei der Interpretation.

Damit der Computer Schlüsse aus der Ontologie ziehen kann, sind neben der Modellierung von Konzepten in Hierarchien, dem Einfügen von Instanzen und der Verbindung von Konzepten über Properties noch weitere Maßnahmen sinnvoll. Während die Zuweisung von Properties zu Konzepten bereits ein erster Schritt in Richtung formaler Definition ist, bieten die unterschiedlichen Ontologiesprachen hier noch vertiefende Werkzeuge. In OWL können neue Klassen beispielsweise explizit als Vereinigungsmenge (*unionOf*) oder Gegenteil von (*complementOf*) bestehender Klassen definiert werden. Die auf diese Weise entstehenden Klassen können auch anonym bleiben, müssen also keinen eigenen Klassennamen erhalten (Lit. 03), bei einer UnionOf-Klasse weiß der Reasoner automatisch, dass sich diese aus der Menge der Instanzen der beiden Ursprungsklassen zusammensetzt. Auch die Angabe, dass zwei Klassen disjunkt sind (s. o.), gehört zur formalen Semantik innerhalb der Ontologie.

Weiter ist es möglich, Klassen mit Hilfe der extensionalen Begriffsbestimmung zu definieren: sogenannte *Enumerated Classes* definieren sich über eine präzise Auflistung aller Instanzen, aus denen diese Klasse zusammengesetzt ist (Lit. 05). Die Klasse *Monate* kann also komplett darüber definiert werden, dass sie exakt aus der Menge der Instanzen *Januar*, *Februar*, *März*, *April*, *Mai*, *Juni*, *Juli*, *August*, *September*, *Oktober*, *November* und *Dezember* – und aus nichts anderem – besteht.

Für möglichst präzise intensionale Begriffsbestimmungen sollten Klassen nicht nur mit Properties versehen werden, sondern diese auch für die Formulierung von notwendigen oder hinreichenden Bedingungen (*necessary and sufficient conditions = restrictions*) genutzt werden. Necessary Conditions werden eingesetzt um zu definieren, dass etwas nur dann Mitglied einer Klasse sein kann, wenn es diese Bedingung erfüllt.

Für die Klassenmitgliedschaft ist es also notwendig, eine entsprechende Bedingung zu erfüllen. Reasoner können mit Hilfe von notwendigen Bedingungen überprüfen, ob z. B. Instanzen fälschlicherweise einer Klasse zugeordnet wurden: Nehmen wir an, wir definieren die Klasse *Trainer* damit, dass es eine Unterklasse von *Person* ist und als notwendige Bedingungen eine Relation

zu einem *Verein* anhand der Property *trainiert* vorliegen muss: nur wenn diese Bedingung erfüllt ist, kann eine Instanz der Klasse *Trainer* zugehörig sein. Wird nun in der Klasse *Trainer* eine Instanz vorgefunden, die nicht mit einer *Vereins*-Instanz verbunden ist, so meldet der Reasoner ein Problem.

Bedingungen können in OWL beispielsweise so formuliert sein, dass eine Property mindestens einmal erfüllt sein muss (*quantifier restrictions*). Daneben gibt es weitere Möglichkeiten für Bedingungen, z. B. genauere Vorschriften, wie oft eine Bedingung erfüllt werden muss (*cardinality restrictions*). Beispielsweise könnte man vorgeben, dass ein Mitglied der Klasse *Erfolgreicher_Verein* mindestens fünfmal einen bestimmten Titel gewonnen haben muss. Und schließlich kann definiert werden, dass innerhalb einer Klasse auf Instanzebene für eine Property immer der gleiche Wert gelten muss (*value restrictions*): Für Instanzen einer Klasse *DeutscheFußballnationalspieler* könnte z. B. gelten, dass für eine Property *hat_Staatsangehörigkeit* immer der Wert *Deutsch* vorhanden sein muss.

Wenn nicht nur notwendige, sondern auch hinreichende Bedingungen (*necessary and sufficient conditions*) definiert sind, kann der Reasoner nicht nur falsch zugeordnete Instanzen (bzw. fehlende Angaben bei Instanzen) diagnostizieren, sondern auch Instanzen automatisch in passende Klassen eingruppieren. Notwendige und hinreichende Bedingungen für eine Klasse geben an, dass sofern ein Element diese Bedingungen erfüllt, es auch zwingend dieser Klasse zugeordnet werden muss. Klassen, die mit notwendigen und hinreichenden Bedingungen versehen sind, werden auch *Complete Class* oder *Defined Class* genannt (Lit. 05). Beispielsweise könnte man festlegen, dass jede Instanz, für die sowohl die Property *ist_Mitglied_von* mit einer Instanz aus *Nationalmannschaft* und die Property *hat_Staatsangehörigkeit* mit *Deutsch* angegeben wurde, automatisch der Klasse *DeutscheNationalspieler* hinzugefügt werden soll.

B 6.2.5 Besonderheiten gegenüber anderen Wissensorganisationsmethoden

Eine Ontologie, die alle strukturellen Gestaltungsmöglichkeiten ausschöpft, unterscheidet sich deutlich von klassischen Methoden der Wissensrepräsentation wie Thesauri und Klassifikationen. Vor allem die Möglichkeit, Klassen formal zu definieren und Klassenzugehörigkeiten oder (vererbte) Eigenschaften automatisch zu ermitteln, machen Ontologien zu einer besonderen Form der Wissensrepräsentation und zu einem essentiellen Baustein für semantische Web-Anwendungen. Aber auch andere Besonderheiten gilt es noch einmal hervorzuheben: Mit Klassenhierarchien und Properties können Ontologien nicht nur das Standard-Repertoire semantischer Relationen abdecken (Abstraktions-, Bestands-, Äquivalenz- und Assoziationsrelation) (Lit. 19), sie ermöglichen auch den Einsatz weiterer spezifizierter Verbindungen zwischen den Konzepten eines Interessensgebietes und bilden so die Grundlage für verfeinerte semantische Repräsentationen. Der Einsatz von Datatype Properties eröffnet weitere Möglichkeiten für die präzise Definition einzelner Konzepte und die Einbindung von Fakteninformationen über Instanzen. Die formale Unterscheidung zwischen Klassen und Instanzen ist so bislang in keinem klassischen Wissensrepräsentations-Typus realisiert.

In der Praxis werden diese Möglichkeiten jedoch bislang so gut wie nie voll ausgeschöpft. Hier unterscheiden sich Ontologien vor allem dadurch von anderen Methoden, dass Sie durch den Einsatz einer standardisierten Ontologiesprache maschinenlesbar und interpretierbar werden. Um diesen Vorteil auszunutzen, werden derzeit auch einzelne bewährte Thesauri in Ontologiesprachen überführt: der Standard Thesaurus Wirtschaft wird in SKOS und RDF aufbereitet, auch AGROVOC ist inzwischen als RDF-Version verfügbar. Derartige Bemühungen zur Formalisierung bestehender kontrollierter Vokabulare sind vor allem auch im Rahmen der Linked Data-Initiativen zu verorten (vgl. B 7 Semantic Web und Linked Open Data).

B 6.3 Beispiele für Ontologien im praktischen Einsatz

Neben diesen Erweiterungen bestehender Wissensorganisationsmodelle wollen wir abschließend verschiedene Beispiele für Ontologien und Ontologieprojekte kurz betrachten. Nur wenige Projekte, darunter die Cyc Ontologie (URL2), haben versucht, detailliert universelles Weltwissen im Sinne der philosophischen Ontologie abzubilden. Praktisch eingesetzte Ontologien befassen sich eher mit der Erstellung von Wissensmodellen für spezialisierte, eng abgegrenzte Themengebiete, die sich genau auf ihren Einsatzbereich beziehen. Auch formal sind die folgenden Ontologien eher schlicht gehalten. Wie beim Aufbau anderer Wissensorganisationssysteme sind auch bei der Erstellung solcher Fachontologien (*domain ontologies*) sowohl Expertenwissen im jeweiligen Themenbereich als auch Spezialkenntnisse im Ontologieaufbau gefordert.

Eine Vorreiterrolle im Einsatz von Ontologien kam zunächst den Lebenswissenschaften zu (Lit. 20). Biologie und Bioinformatik entwickelten zuerst ein besonderes Interesse am Nutzen von Ontologien: Einerseits sollten diese dabei helfen, die komplexen Terminologien innerhalb der Fachgemeinschaft einheitlich zu definieren und zu strukturieren (vergleichbar mit dem bewährten Einsatz von Taxonomien in der Zoologie). Andererseits sollten Ontologien helfen, die in der biologischen Forschung stetig wachsenden Mengen von Daten (sowohl Primärdaten als auch Publikationen) zu erschließen und suchbar zu machen (Lit. 10). Besonders beachtet wurden die Open Biomedical Ontologies (OBO), eine Sammlung von verschiedenen Bio-Ontologien, unter anderem die vielleicht populärste: *Gene Ontology (GO)* (Lit. 21). Diese Ontologien gehen oftmals nur wenig über die Ausdrucksstärke von Hierarchien als Wissensorganisationsprinzip hinaus (und werden daher mitunter auch in Einzelfällen als Thesaurus oder Taxonomie beschrieben). Die Gene Ontology wurde entwickelt, um Gene und Gen-Produkte innerhalb verschiedener Fachdatenbanken zu indexieren. Sie umfasst präzise Definitionen für vorwiegend hierarchisch (*is-a* und *part-of* Relationen) strukturierte Konzepte aus den drei Teilbereichen *Cellular Component*, *Biological Process* und *Molecular Function*. Darüber hinaus werden nur sehr wenige spezifizierte Relationen eingesetzt. Die TAMBIS Ontology als weiteres Beispiel wurde als einheitliches Suchvokabular für verschiedene Informationsquellen im Bereich Biologie und Bioinformatik entwickelt. Sie verfügt über viele verschiedene, sehr spezifische Properties, enthält jedoch keine Instanzen. Auch im Unternehmenskontext werden Ontologien als mögliches Hilfsmittel diskutiert und ausprobiert, insbesondere für das Wissensmanagement in Unternehmen. Das aktuell vielleicht populärste Beispiel für angewandte Ontologien stammt aus dem Bereich Industrie und e-Commerce: Die *eClassOWL* (URL3) zusammen mit der *GoodRelations Ontology* (URL4). In Kombination sollen beide dabei helfen, Produkte und Dienstleistungen mit semantischen Metadaten eindeutig zu beschreiben. *GoodRelations* repräsentiert dabei Produkte und Dienstleistungen hauptsächlich mit Hilfe von Merkmalen wie *BusinessEntity*, *Offerings* und *Location*, an die jeweils spezifische Properties gebunden sind. Der Fokus liegt hierbei weniger auf den Produkten an sich als vielmehr bei branchenunabhängigen Zusammenhängen zwischen Angeboten und Anbietern (beispielsweise Bezahlung, Lieferung etc.). Für die tatsächliche Beschreibung der Produkttypen und ihren Eigenschaften wird die *eClassOWL* (Lit. 22) hinzugenommen. Sie enthält mehr als 30.000 Produkttypen und über 5000 Properties, die diese beschreiben.

Ein sehr schlichtes RDF-Vokabular wird derzeit als Ontologie für den Einsatz im Social Web viel beachtet. Die *FOAF Ontology* (Friend Of A Friend, URL5) hilft dabei, Personen, ihre Interessen, von ihnen geschaffene Dokumente wie Bilder und ihre Beziehungen zu anderen Personen einheitlich mit Metadaten zu beschreiben. Ziel des FOAF Projektes ist es, Webseiten mit Hilfe des FOAF Vokabulars zu erschließen, um so Suchen nach Personen anhand verschiedener Kriterien zu ermöglichen. Noch stärker als FOAF zielt das SIOC Projekt (Lit. 23) darauf ab, alle Aktivitäten einer Person im Web zu erfassen und beschreiben zu können. Es geht darum, ein semantisches Modell speziell für Beiträge im Social Web (vgl. D 7 Social Web) bereitzustellen, was einen ersten Schritt in Richtung des Social Semantic Web bedeutet (Lit. 10, Lit. 24): Der Einsatz von Ontologien für die Beschreibung von *User-Generated-Content* im Web 2.0 (Blogbeiträge, Wiki-Artikel, Nutzercommunities) ist einer der Trends, die derzeit die Richtung für die Ontologie- und Semantic-Web-Forschung vorgeben.

B 6.4 Fazit: Perspektiven und Herausforderungen

Wir haben in diesem Kapitel einen Einblick in die Ausdrucksmöglichkeiten von Ontologien zur Wissensorganisation vermittelt. Abschließend soll kurz zusammengefasst werden, an welchen Herausforderungen Ontologiedesigner und Semantic-Web-Community derzeit noch besonders arbeiten. Die technischen Grundlagen und die Infrastruktur für die Ontologieerstellungen sind vorhanden, werden aber nach und nach weiter verfeinert. Im praktischen Einsatz befinden sich bereits einzelne Ontologien, aber flächendeckend durchgesetzt haben sich diese noch nicht. Ein Ziel für die Zukunft wird es daher sein, weitere Anwendungen zu erstellen, die nah an den Nutzerbedürfnissen orientiert sind und den Nutzen semantischer Metadaten in Form von Ontologien deutlich machen. Auch sollten Ontologien einen festen Platz in informationswissenschaftlichen Curricula haben. Die Verbindung von Social Web und Ontologien ist ein diesbezüglicher Trend, die Erweiterung bestehender Indexierungsvokabulare um formale Definitionen ein anderer. Da der Arbeitsaufwand für die Ontologieerstellung hoch ist, wird auch an Methoden gearbeitet, die es erlauben, größere Ontologien kollaborativ, vor allem mit Hilfe von Web-Communities aufzubauen, die dabei auch ihr kollektives Wissen in die Abbildung eines Themenbereichs einfließen lassen.

Die aber wohl größte Herausforderung besteht darin, zwischen verschiedenen einzelnen Ontologien sowie zwischen Ontologien und anderen Wissensrepräsentationsmodellen zu vermitteln und Querverweise zu ermöglichen. Hier sind in Zukunft vor allem Ansätze zur Standardisierung auf Ebene der konzeptionellen Modellierung von Ontologien sowie zur Verbindung von Konzepten und Properties über die Grenzen einer Ontologie hinaus gefragt.

Literatur

01 Stock, W. G.; Stock, M.: Wissensrepräsentation: Informationen auswerten und bereitstellen. München, Wien: Oldenbourg, 2008

02 Weller, K.: Folksonomies and Ontologies: Two new players in indexing and knowledge representation. H. Jezzard (Ed.), Applying Web 2.0. Innovation, Impact and Implementation. Proceedings of the Online Information Conference, London, Great Britain, 108-115, 2007. London: Learned Information Europe

03 Smith, M. K.; Welty, C.; McGuinness, D. L. (Eds.): OWL Web Ontology Language Guide: W3C Recommendation 10 February 2004 (http://www.w3.org/TR/2004/REC-owl-guide-20040210/)

04 Miles, A.; Matthews, B.; Wilson, M.; Brickley, D.: SKOS core: Simple knowledge organisation for the Web. Vocabularies in Practice. Proceedings of the International Conference on Dublin Core and Metadata (DC 2005), Madrid, Spain (3-10)

05 Horridge, M.; Knublauch, H.; Rector, A.; Stevens, R.; Wroe, C. (2004). A Practical Guide To Building OWL Ontologies Using The Protégé-OWL Plugin and CO-ODE Tools: Edition 1.0, 2004 (http://www.co-ode.org/resources/tutorials/ProtegeOWLTutorial.pdf)

06 Noy, N. F.; McGuinness, D. L.: Ontology Development 101: A Guide to Creating Your First Ontology. Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880, March 2001. (http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html)

07 Horrocks, I.: Ontologies and the Semantic Web: How ontologies provide the semantics, as explained with the help of Harry Potter and his owl Hedwig. Communications of the ACM 51 (12) 58-67, 2008

08 Smith, B.; Welty, C.: Ontology: Towards a new synthesis. C. Welty; B. Smith (Eds.), Formal Ontology in Information Systems: Collected Papers from the Second International Conference (FOIS 2001) (3-9). New York: ACM

09 Gruber, T. R.: A translation approach to portable ontology specification. Knowledge Acquisition 2 (5), 199-220, 1993

10 Weller, K.: Knowledge Representation in the Social Semantic Web. (Knowledge & Information, Vol. 3). Berlin: De Gruyter Saur, 2010

11 Berners-Lee, T.; Hendler, J.; Lassila, O.: The Semantic Web. Scientific American 284 (5) 34-43, 2001
12 Hitzler, P.; Krötzsch, M.; Rudolph, S.; Sure, Y.: Semantic Web. Berlin, Heidelberg: Springer, 2008
13 Gómez-Pérez, A.; Fernández-López, M.; Corcho, O.: Ontological Engineering: Advanced Information and Knowledge Processing (3rd Print). London: Springer, 2004
14 Motik, B.; Cuenca Grau, B.; Horrocks, I.; Wu, Z.; Fokou, A.; Lutz, C.: OWL 2 Web Ontology Language: Profiles. W3C Working Draft 08 October, 2008 (http://www.w3.org/TR/2008/WD-owl2-profiles-20081008/)
15 Manola, F.; Miller, E.: Resource Description Framework (RDF). Primer: W3C Recommendation, 10 February 2004 (http://www.w3.org/TR/rdf-primer/)
16 Staab, S.; Studer, R. (Eds.): Handbook on Ontologies. Berlin, Heidelberg, New York: Springer, 2004
17 Beckett, D.; Broekstra, J.: SPARQL Query Results XML Format: W3C Working Draft 14 June 2007 (http://www.w3.org/TR/rdf-spargl-XMLres/)
18 Fikes, R.; Hayes, P.; Horrocks, I.: OWL-QL: A Language for Deductive Query Answering on the Semantic Web (KSL Technical Report No. 03-14). Stanford, CA: Knowledge Systems Laboratory, Stanford University, 2003
19 Peters, I.; Weller, K.: Paradigmatic and syntagmatic relations in knowledge organization systems. Information – Wissenschaft & Praxis 59 (2) 100-107, 2008
20 Bodenreider, O.; Stevens, R.: Bio-ontologies: Current trends and future directions. Briefings in Bioinformatics 7 (3) 256-274, 2006
21 Ashburner, M.; Ball, C. A.; Blake, J. A.; Botstein, D.; Butler, H.; Cherry, J. M. et al.: Gene Ontology: Tool for the unification of biology. Nature Genetics 25, 25-29, 2000
22 Hepp, M.: Products and services ontologies: A methodology for deriving OWL ontologies from industrial categorization standards. International Journal on Semantic Web & Information Systems (IJSWIS) 2 (1) 72-99, 2006
23 Bojars, U.; Breslin, J. G.; Finn, A.; Decker, S.: Using the Semantic Web for linking and reusing data across Web 2.0 communities. Journal of Web Semantics 6 (1) 21-28, 2008
24 Breslin, J. G.; Passant, A.; Decker, S.: The Social Semantic Web. Berlin: Springer, 2009

URL-Verzeichnis

URL1 http://protege.stanford.edu/
URL2 http://www.cyc.com
URL3 http://www.heppnetz.de/projects/eclassowl/
URL4 http://purl.org/goodrelations/
URL5 http://www.foaf-project.org/