Technische Universität Wien
Vienna University of Technology

AK aus Information Engineering 2

5.5.2004, 15:00-16:30

Monika Lanzenberger

Überblick

Basiert auf:

"Ontology Development 101: A Guide to Creating Your First Ontology" by Natalya F. Noy and Deborah L. McGuinness

und

"Ontology Engineering for the Semantic Web and Beyond"

by Natalya F. Noy

http://protege.stanford.edu/publications/ontology_development/ontology101.html

- Einführung in Ontologie Engineering
- Schritte im Entwicklungsprozess
- Klassen und Klassenhierarchie
- Attribute (Properties / Slots)
- Praktisches Beispiel mit Protége

Ontology Engineering

- Begriffe definieren und deren Relationen beschreiben
  - Definieren von Konzepten (Klassen) in der Wissensdomäne
  - Konzepte in einer Hierarchie ordnen (subclass-superclass)
  - Definieren der Attribute und Merkmale, deren Einschränkungen und möglichen Werte
  - Definieren der Instanzen (Individuals) und erfassen der jeweiligen Werte

Warum eine Ontologie?

- Gemeinsames Verständnis von der Struktur der Information
  - zwischen Menschen
  - zwischen Software Agenten
- Um Wiederverwendung von Wissen zu erleichtern
  - "Rad nicht ständig neu erfinden"
  - Etablieren von Standards um Wissenaustausch zu ermöglichen

Warum eine Ontologie? 2

- Voraussetzungen in einer Wissensdomäne explizit beschreiben
  - Vereinfacht Änderung der Annahmen (Voraussetzungen) in einer Wissensdomäne
  - Vereinfacht das Verstehen von übernommenen Daten
- Faktenwissen (Was) und operationales Wissen (Wie) unterscheiden
  - Wiederverwenden von Faktenwissen und operationalen Wissen getrennt möglich

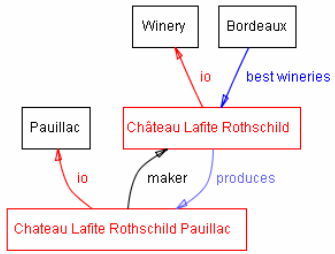
Ontology Engineering: Aktuelle Fragen

- Content erzeugen
- Analyse und Evaluation
- Maintenance
- Ontologie Sprachen
- Tools Entwicklung

Evaluation

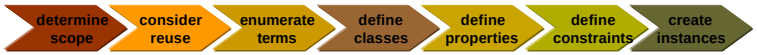
- Eines der kompliziertesten Probleme im Ontologie-Design
- Ontologie-Design ist subjektiv
- Wann ist eine Ontologie korrekt (objektiv)?
- Testen mit einer Applikation, für die die Ontologie erzeugt wurde

Anwendungsgebiet

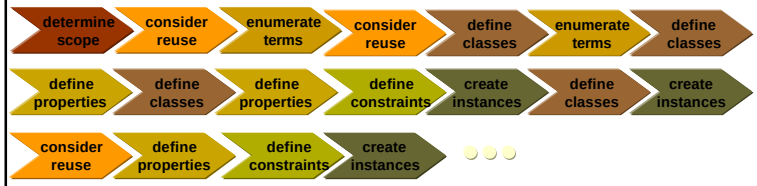


Ontologie Entwicklungsprozess

In der Theorie:



In der Praxis:



Ontology Engineering - Objekt Orientierung

Ontologie
- beschreibt die Gegebenheiten der Welt
- handelt von der Ordnung der Konzepte
- physische Repräsentation ist nicht zentral

Obj.Orient. Klassenstruktur
- beschreibt die Struktur von Daten und Code
- handelt von Methoden
- beschreibt die konkrete Repräsentation der Daten (long int, char, etc.)

Tools

- Alle hier verwendeten Screenshots stammen von Protégé-2000:
  - ein Ontolgie-Entwicklungs Tool mit GUI
  - Open-Source (http://protege.stanford.edu)
- Andere Tools:
  - Ontolingua (ontolingua.stanford.edu)
  - Chimaera (www.ksl.stanford.edu/sofware/chimaera)
  - OntoEdit (www.ontoprise.de/products/ontoedit)
  - OilEd (oiled.man.ac.uk/download.shtml)

Überblick

- Einführung in Ontologie Engineering
- Schritte im Entwicklungsprozess
- Klassen und Klassenhierarchie
- Attribute (Properties / Slots)
- Praktisches Beispiel mit Protége

Definition von Anwendungsgebiet und Fokus

- Welches Anwendungsgebiet soll die Ontologie abdecken bzw. beschreiben?
- Welche Ziele sollen durch die Anwendung dieser Ontologie erreicht werden?
- Für welche Typen von Fragen soll die Ontologie Antworten bereitstellen (competency questions)?
- *Die Antworten auf diese Fragen können sich ändern*

Zentrale Fragestellungen

- Wein-Ontolgie:
  - Welche Charakteristika eines Weines müssen miteinbezogen werden bei der Auswahl eines passenden Weines?
  - Ist Bordeaux ein Rotwein oder ein Weißwein?
  - Passt Cabernet Sauvignon zu Fisch?
  - Welcher Wein passt am besten zu gegrillten Speisen?
  - Ändert sich die Farbe und der Geruch eines bestimmten Weines abhängig vom Jahrgang?
  - Welche Jahrgänge sind bei Napa Zinfandel gut?

Wiederverwendbarkeit

- Warum bestehende Ontologien wiederverwenden?
  - Um Aufwand zu sparen
  - Um ev. bestehende Tools anzuwenden
  - Bereits mit konkreten Applikationen getestet

Was kann wiederverwendet werden? 1

- Ontology libraries
  - DAML ontology library (www.daml.org/ontologies)
  - Ontolingua ontology library (www.ksl.stanford.edu/software/ontolingua/)
  - Protégé ontology library (protege.stanford.edu/plugins.html)
- “Abstrakte” Ontologien
  - IEEE Standard Upper Ontology (suo.ieee.org)
  - Cyc (www.cyc.com)

Was kann wiederverwendet werden? 2

- Allgemeine Ontologien
  - DMOZ (www.dmoz.org)
  - WordNet (www.cogsci.princeton.edu/~wn/)
- Domänspezifische Ontologien
  - UMLS Semantic Net
  - GO (Gene Ontology) (www.geneontology.org)

Relevante Begriffe sammeln

- Welche Begriffe/Dinge müssen unbedingt beschrieben werden?
- Welche Attribute/Merkmale haben diese Dinge?
- Was wollen wir über diese Begriffe/Dinge aussagen?

Relevante Begriffe: Eine Wein Ontologie

- *Wein, Rebsorte, Winzerei, Anbaugebiet,*
- *Farbe, Körper, Abgang, Bukett, Restzucker,*
- *Weißwein, Rotwein, Bordeaux, Veltliner,*
- *Essen, Meeresfrüchte, Fisch, Fleisch, Gemüse, Käse*

Klassen und Klassenhierarchie definieren

- Klasse ist ein Konzept in einer Wissensdomäne
  - Klasse der Weine
  - Klasse der Winzereien
  - Klasse der Rotweine
- Klasse ist eine Sammlung von Elementen mit ähnlichen Attributen
- Instanz einer Klasse
  - Z.B. das Glas kalifornischen Merlot, das gestern serviert wurde

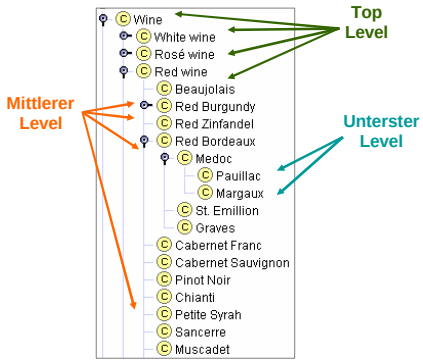
Vererbung zwischen Klassen

- Klassen konstitutieren eine Taxonomie (subclass-superclass hierarchy)
- Eine Klassenhierarchie ist eine “IS-A hierarchy”:
  - *eine Instanz einer Subklasse ist jedenfalls auch eine Instanz der Superklasse*
- Klasse ist ein Set von Elemente -> Subklasse ist ein Subset von Elementen

Vererbung zwischen Klassen: Beispiele

- Apfel ist ein Subklasse von Obst
  - *Jeder Apfel ist Obst*
- Rotwein ist eine Subklasse von Wein
  - *Jeder Rotwein ist ein Wein*
- Chianti ist eine Subklasse von Rotwein
  - *Jeder Chianti ist ein Rotwein*
- *Ein Laptop ist eine Subklasse von Computer*
  - *Jeder Laptop ist ein Computer*

Klassenhierarchie - Level



Strategien zur Bildung einer Hierarchie

- Top-Down – zuerst die allgemeinsten Konzepte definieren und dann weiter spezifizieren
- Bottom-Up – zuerst die konkretesten Begriffe deinfieren und dann in allgemeinere Kategorien einordnen
- Kombination – zuerst die wichtigsten Konzepte definieren und dann konkretisieren und verallgemeinern

Dokumentation

- Klassen (und Attribute) sollen dokumentiert werden:
  - Beschreibung der Klasse in natürlicher Sprache
  - Explizite Auflistung von Grundvoraussetzungen / Annahme, die für die Klassendefinition relevant sind
  - Synonyme auflisten
- Dokumentation von Klassen und Attribute genauso wichtig wie Software-Dokumentation!

Definieren der Attribute 1/2

- Properties/Slots beschreiben die Attribute der Instanzen der jeweiligen Klasse und Relationen zu anderen Klassen:
  - *Jeder Wein hat eine bestimmte Farbe, Restzucker und stammt aus einem bestimmten Anbaugebiet, etc.*

Definieren der Attribute 2/2

- Attribut-Typen
  - "Intrinsische" Attribute: Geschmack und Farbe des Weines
  - "Extrinsic" Attribute: Name und Preis des Weines
  - Bestandteile: Zutaten eines Gerichs
  - Relationen zu anderen Konzepten: WinzerIn
- Einfache und komplexe Attribute
  - Einfache Attribute (Datatype Property): bestehen aus einfachen Werten (Strings, Zahlen)
  - Komplexe Attribute (Object Property): beinhalten oder verweisen auf andere Konzepte (z.B., eine Instanz der Klasse Winzerei)

Wein-Ontologie: Attribute

Template Slots

Name	Type	Cardinality	Other Facets
S body	Symbol	single	allowed-values={FULL,MEDIUM,LIGHT}
S color	Symbol	single	allowed-values={RED,ROSÉ,WHITE}
S flavor	Symbol	single	allowed-values={DELICATE,MODERATE,STRONG}
S grape	Instance	multiple	classes={Wine grape}
S maker	Instance	single	classes={Winery}
S name	String	single	
S sugar	Symbol	single	allowed-values={DRY,SWEET,OFF-DRY}

Attribute und Klassenvererbung

- Eine Subklasse erbt alle Attribute von der Superklasse
  - *Wenn "Wein" einen "Namen" und ein "Bukett" hat, dann hat auch "Rotwein" einen "Namen" und ein "Bukett"*
- Wenn eine Klasse mehrere Superklassen hat, dann erbt sie die Attribute von allen Superklassen
  - *"Port" ist sowohl ein "Dessertwein", als auch ein "Rotwein" und erbt "Restzucker: hoch" vom "Dessertwein" und "Farbe:Rot" vom "Rotwein"*

Attribute definieren - Constraints

- Property constraints (facets) beschreiben bzw. grenzen das Set an möglichen Werten eines Attriutes ein
  - *Der Name eines Weines ist ein String*
  - *ErzeugerIn ist eine Instanz von der Klasse Winzerei*
  - *Eine Winzerei hat eine bestimmteAdresse*

Wein-Ontologie: Property Constraints

Template Slots

Name	Type	Cardinality	Other Facets
S body	Symbol	single	allowed-values={FULL,MEDIUM,LIGHT}
S color	Symbol	single	allowed-values={RED,ROSÉ,WHITE}
S flavor	Symbol	single	allowed-values={DELICATE,MODERATE,STRONG}
S grape	Instance	multiple	classes={Wine grape}
S maker I	Instance	single	classes={Winery}
S name	String	single	
S sugar	Symbol	single	allowed-values={DRY,SWEET,OFF-DRY}

Typische Einschränkungen / Constraints

- Slot cardinality – die erlaubte Anzahl an Werten eines Attributes
- Slot value type – der Werte-Typ eines Attributes
- Minimaler und maximaler Wert – ein Wertebereich für numerische Attribute
- Default value

Typische Einschränkungen - Kardinalität

- Kardinalität
  - Kardinalität N -> Attribut benötigt genau N Werte
- Minimale Kardinalität
  - Minimale Kardinalität 1 -> Mindestens ein Wert notwendig
  - Minmale Kardinalität 0 -> Wert des Attributs ist optional
- Maximale Kardinalität
  - Maximale Kardinalität 1 (single-valued slot)
  - Maximale Kardinalität größer 1 (multiple-valued slot)

Typische Einschränkungen - Typ

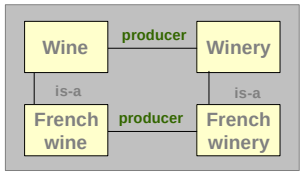
- String: Zeichenkette ("Grüner Veltliner")
- Zahl: Integer oder Float
- Boolean: True/False
- Aufzählungstyp (Symbol) z.B. high, medium, low
- Komplexe Constraints: z.B. Instanzen einer Klasse
  - Klasse(n) der erlaubten Instanzen angeben
  - *Z.B. für "produziert" als Attribut der Klasse "Winzerei" sind Instanzen aus der Klasse "Wein" erlaubt*

Domäne und Wertebereich

- Domain of a slot (Domäne) – Klasse(n) die durch dieses Attribut beschrieben weden
  - Genauer: Klasse(n) deren Instanzen durch dieses Attribut beschrieben werden
- Range of a slot (Wertebereich) – Klasse(n) aus der (denen) der Wert des Attributes kommt

Vererbung der Constraints

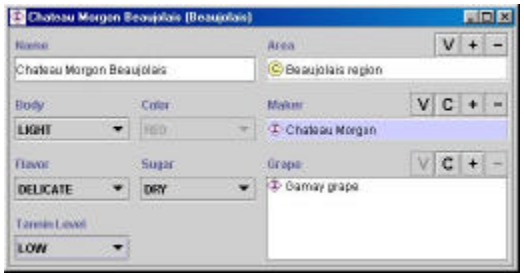
- Subklasse erbt alle Attribute der Superklasse
- Subkklasse kann die Constraints überschreiben -> "Einengnung" der erlaubten Werte:
  - Z.B. Kardinalität kleiner setzen
  - Wertebereich auf eine Subklasse einschränken



Instanzen anlegen

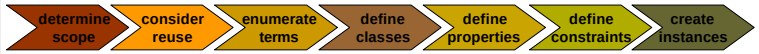
- Instanz einer Klasse erzeugen
  - Klasse wird zum “direct type” dieser Instanz
  - Jede Superklasse von diesem “direct type” ist ein “type” dieser Instanz
- Attribut-Werte für die Instanz eintragen
  - Werte müssen den Constraints entsprechen
  - Knowledge-acquisition Tools überprüfen Wertebereich

Instanzen anlegen – ein Beispiel

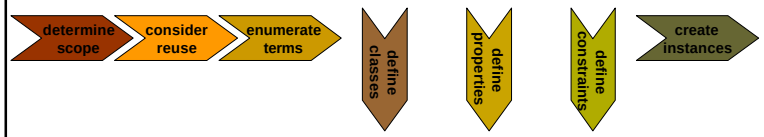


Entwicklungsprozess: Strategien

- Zuerst in die Breite



- Zuerst in die Tiefe



Überblick

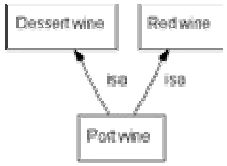
- Einführung in Ontologie Engineering
- Schritte im Entwicklungsprozess
- Klassen und Klassenhierarchie
- Attribute (Properties / Slots)
- Praktisches Beispiel mit Protége

Klassen und Klassenhierarchien

- Zusammenfassend:
  - Wann eine Subklasse, wann ein Attribut?
  - Wann eine Instanz, wann eine Klasse?
  - Nicht nur eine korrekte Klassenhierarchie
  - Aber es gibt Empfehlungen für eine gute Klassenhierarchie
- Zu Überprüfen ist:
  - "Ist jede Instanz einer Subklasse auch eine Instanz der Superklasse?"

Mehrfachvererbung

- Eine Klasse kann mehrere Superklassen haben
- Eine Subklasse erbt alle Attribute und Constraints von den Superklassen
- Erbt die Farbe "Rot" von der Klasse "Rotwein" und den hohen Restzucker der Klasse "Dessertwein"

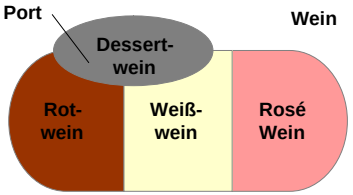


Disjunkte Klassen

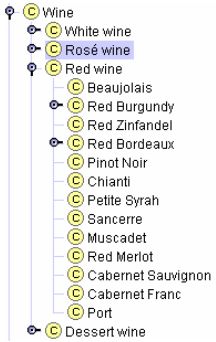
- Klassen sind disjunkt, wenn sie keine gemeinsamen Instanzen haben können
- Disjunkte Klassen können keine gemeinsamen Subklassen haben

Rotwein, Weißwein, Rosé Wein sind disjunkt

Dessertwein und Rotwein sind nicht disjunkt

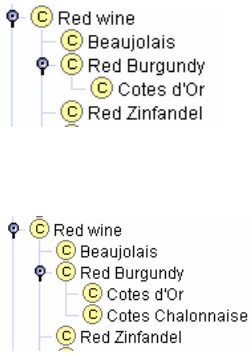


Geschwister-Klassen



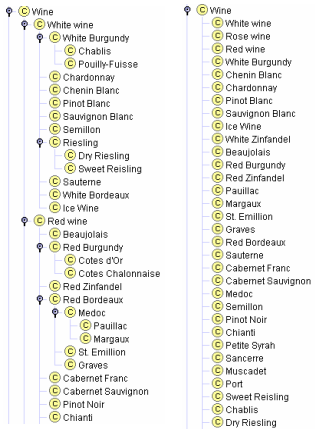
- Alle Geschwister-Klassen müssen auf der selben Hierarchiestufe sein
- Vergl. Kapitel und Unterkapitel in einem Buch

Richtlinien zur Klassengröße 1/2



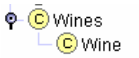
- Klasse mit einer Subklasse deutet auf Modellierungsfehler hin
- Wenn nur eine Weinkategorie zu “Red Burgundy” existiert (“Côtes d’Or”), warum eine Subklasse dafür anlegen?
- Vergl. Punkteliste

Richtlinien zur Klassengröße 2/2

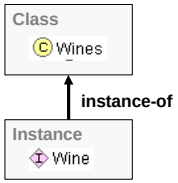


- Wenn eine Klasse sehr viele Subklassen hat, sind meist weitere Klassenhierarchien sinnvoll
- Wenn sich das aus der natürlichen Kategorisierung ergibt, ansonsten lange Liste besser

Klassennamen



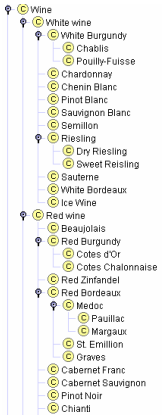
- Ein bestimmter “Wein” ist nicht eine Subklasse der Klasse “Weine”
- Ein bestimmer Wein ist eine Instanz der Klasse “Weine”
- Klassennamen sollen entweder
  - alle singular sein
  - oder alle plural sein



Klassennamen

- Klassen repräsentieren Konzepte einer Wissensdomäne, nicht deren Namen
- Der Klassennamen kann geändert werden, verweist aber auf das selbe Konzept
- Synonyme Namen für ein bestimmtes Konzept sind nicht verschiedene Klassen
  - Aufzählen von Synonymen ist der Teil der Klassendefinition

Eine „komplette“ Wein-Ontologie



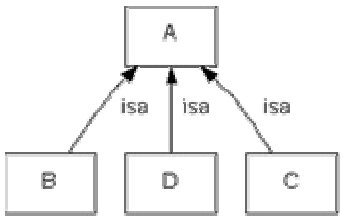
Überblick

- Einführung in Ontologie Engineering
- Schritte im Entwicklungsprozess
- Klassen und Klassenhierarchie
- Attribute (Properties / Slots)
- Praktisches Beispiel mit Protége

Attribute: Domäne und Wertebereich

- Domäne und Wertebereich eines Attributs: die allgemeinste Klasse
- Z.B. das Attribut “Farbe”
  - Domäne: Rotwein, Weißwein, Rosé Wein
  - Domäne: Wein
- Z.B. das Attribut “produziert” der Klasse “Winzerei” :
  - Wertebereich: Rotwein, Weißwein, Rosé Wein
  - Wertebereich: Wein

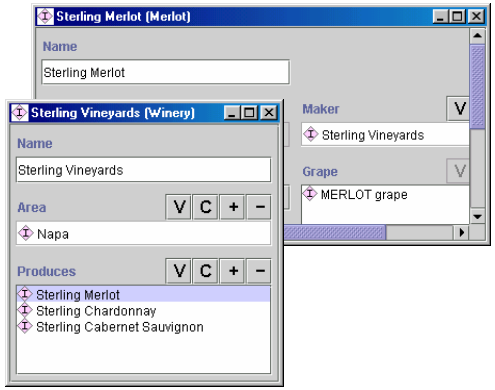
Definieren von Domäne und Wertebereich



- Eine Klasse und eine Superklasse –> Superklasse besser
- Alle Subklassen einer Klasse –> Superklasse besser
- Die meisten Subklassen einer Klasse –> möglicherweise Superklasse besser

Inverse Attribute 1

"Maker" und "Produces" sind inverse Attribute



Inverse Attribute 2

- Inverse Attribute beinhalten redundante Information, aber
  - Ermöglichen eine Ableitung von Information in beide Richtungen
  - Ermöglichen zusätzliche Verification
  - Ermöglichen eine Präsentation von Information in beide Richtungen
- Konkrete Implementation bei verschiedenen Systemen unterschiedlich
  - Werden beide Werte gespeichert?
  - Wann werden die inversen Werte erfasst?
  - Was passiert wenn die Verbindung zu einem inversen Attribut verändert wird?

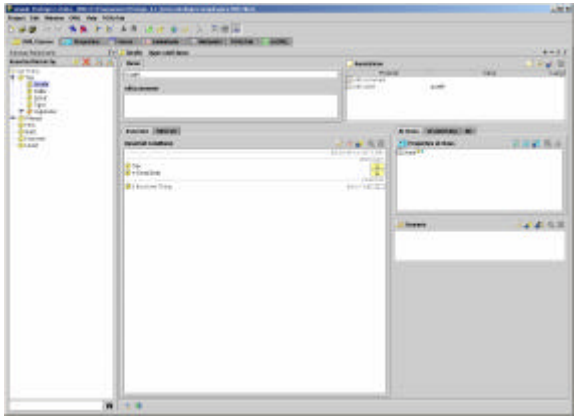
Festlegen der Ziele

- Eine Ontologie soll nicht jede mögliche Information aus einer Wissensdomäne abbilden
- Konkretisieren oder Abstrahieren nicht über die Anforderungen der Applikationen hinausgehend
- Nicht alle möglichen Attribute einer Klasse anlegen:
  - Nur die wichtigen Attribute
  - Nur die Attribute, die für die Applikation notwendig sind

Wein-Ontologie: Festlegen der Ziele

- Ontologie für Wein, Essen, und deren Kombination wird vielleicht nicht beinhalten:
  - Flaschengröße
  - Farbe des Etiquettes
  - Persönliche Wein- oder Essens-Präferenzen
- Eine Ontologie für biologische Experimente wird z.B. beinhalten:
  - Biologische Organismen
  - ForscherIn
- Ist die Klasse "ForscherIn" eine Subklasse der "Biologischen Organismen"?

Überblick

- Einführung in Ontologie Engineering
- Schritte im Entwicklungsprozess
- Klassen und Klassenhierarchie
- Attribute (Properties / Slots)
- Praktisches Beispiel mit Protége

Praktisches Bespiel