Semantic Web

Ontology Engineering

Gerd Gröner, Matthias Thimm

{groener,thimm}@uni-koblenz.de

Institute for Web Science and Technologies (WeST)
University of Koblenz-Landau

July 17, 2013

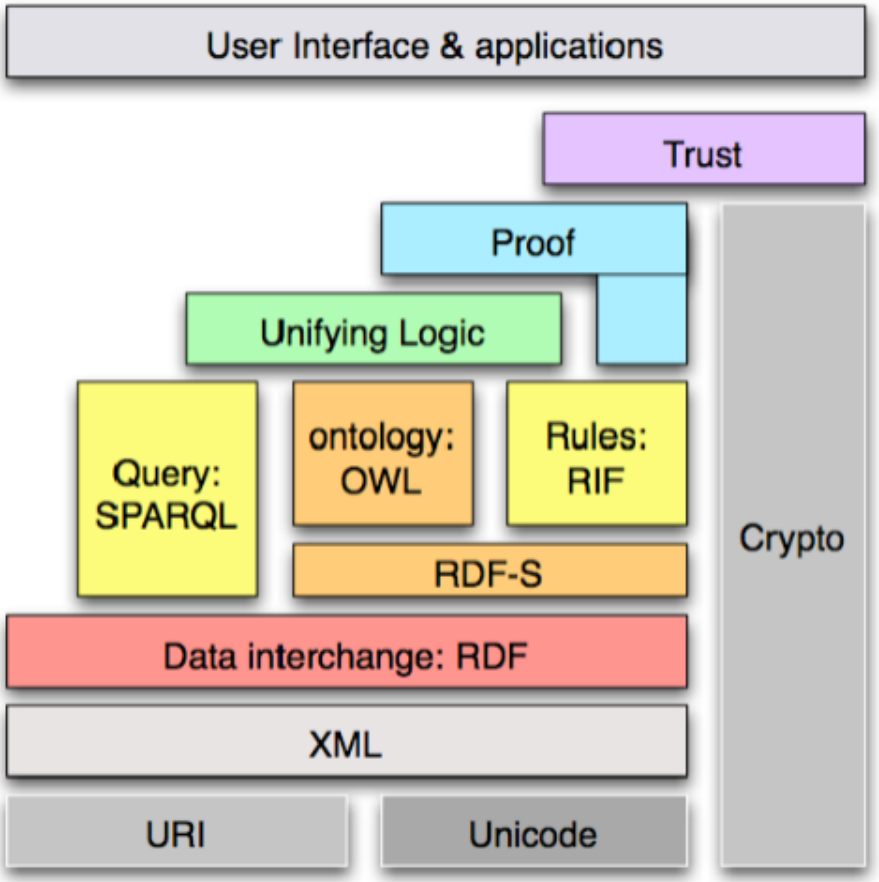
Outline

Outline

Ontology engineering

- An ontology (or any other form of knowledge base)
  - can be used for reasoning, answering queries
  - should be reusable
- Process of building an ontology is called *ontology engineering*
  1. Define language
     - terms
     - concepts,
     - relations, etc.
  2. Define knowledge
     - What are equivalent concepts?
     - Are there subset relations?
     - Which constraints have to be imposed? etc.
  3. Use ontology for reasoning

Remember: Semantic Web Layer Cake



Learning Goals

- Qualify in the engineering and design of an ontology.
- Reflect on different aspects when creating an ontology.

Ontology design principles

based on paper by Natasha Noy and Deborah McGuiness:

"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.

What is an ontology?

An ontology is a formal specification of a shared conceptualization of a domain of interest. (Gruber, 1993)

- automatically processable
- based on an agreement (common understanding)
- definition of terms
- specified subject area

Why should we use ontologies? Some reasons ...

- Share common understanding of the structure of information among people or software agents
  - One common knowledge and data model (computer readable and understandable!)
- Enable reuse of domain knowledge
  - Define once, use in many applications in the same field
- Make domain assumptions explicit
  - Define classes, relationships and instances
- Separate domain knowledge from the operational knowledge
  - What is schema (meta information) and instances (real world)
- Analyze domain knowledge
  - What is the meaning of associations between objects
  - What are existing objects
  - What is the universe of discourse

Why should we use ontologies? (2)

Some thoughts:

- Why is a common knowledge and data model so important?
  - E.g., assume there is different information (from different sources) about the same topic (for instance, some medical ontology)

  → an agent should be able to *combine* information from all sources

  → agents might identify similarities (or even contradictions)
- Why is reuse of ontologies important?

  → avoid developing effort

  → reuse modeling principles from experts (quite often, very detailed models and modeling patterns)
- What are benefits of schema and instance separation?

  → data types of software systems and applications can be built in a first step and then the data (instances) can be "populated".

Building ontologies — the basics

In practical terms, developing an ontology includes:

- defining classes (concepts) in the ontology,
- arranging the classes in a taxonomic (subclass-superclass) hierarchy,
- defining slots (relationships) and describing allowed values for these slots,
- filling in the values for slots (the instances).

Fundamental thumb rules

There is no one correct way to model a domain — there are always viable alternatives.

The best solution almost always depends on the application that you have in mind and the extensions that you anticipate.

Complexity of your ontology should reflect your particular interest in specific area — model what you need, not all that you can.

Fundamental thumb rules

There is no one correct way to model a domain — there are always viable alternatives.

The best solution almost always depends on the application that you have in mind and the extensions that you anticipate.

Complexity of your ontology should reflect your particular interest in specific area — model what you need, not all that you can.

Fundamental thumb rules

There is no one correct way to model a domain — there are always viable alternatives.

The best solution almost always depends on the application that you have in mind and the extensions that you anticipate.

Complexity of your ontology should reflect your particular interest in specific area — model what you need, not all that you can.

A simple knowledge-engineering methodology

Ontology development is **necessarily an iterative process**.

Concepts in the ontology should be close to **objects** (physical or logical) and **relationships** in your domain of **interest**.

These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe your domain.

A simple knowledge-engineering methodology (2)

1. Determine the domain and scope of the ontology
2. Consider reusing existing ontologies (or parts of them)
3. Enumerate important terms in the ontology
4. Define the classes and the class hierarchy
5. Define the properties of the classes (slots)
6. Define the facets of the slots
7. Create instances (of classes)

Step 1: Determine the domain and scope

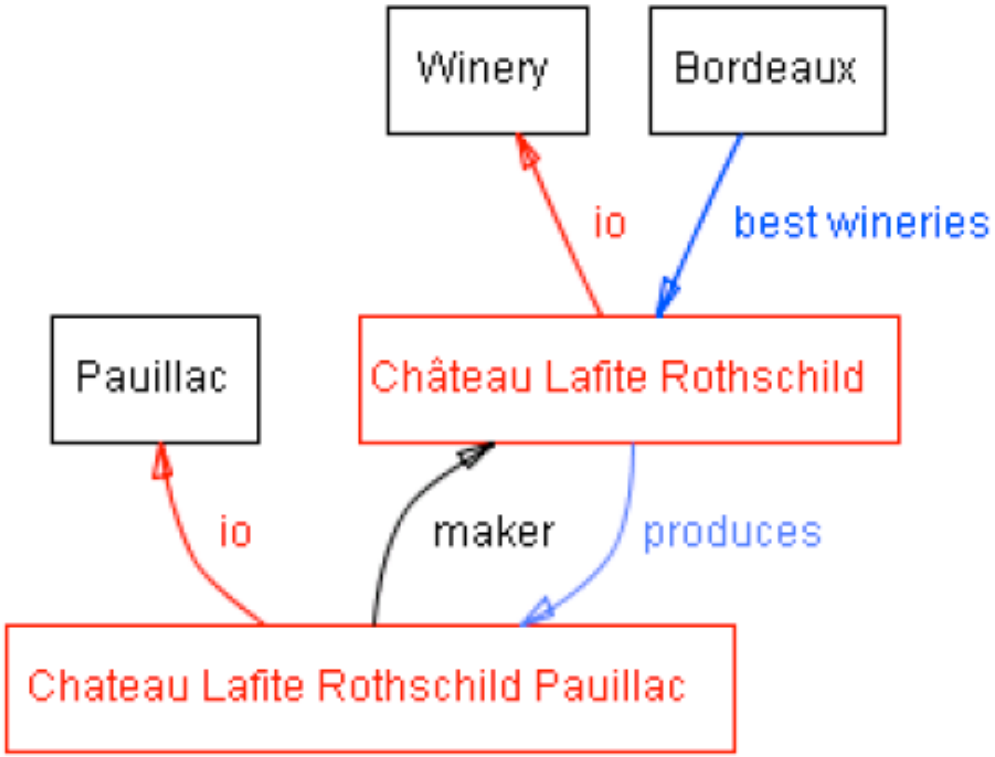
- What is the **domain** that the ontology will cover?
- For what we are going to **use** the ontology?
- For what **types of questions** the information in the ontology should provide answers?
  - $\rightarrow$ competency questions
- Who will **use** and **maintain** the ontology?
  - $\rightarrow$ comprehensible modeling / design decisions

The answers to these questions may change during the ontology design process.

In general, these answers help to limit the scope of the model.

Step 1: Example

The Wine Ontology (very simplified ...)



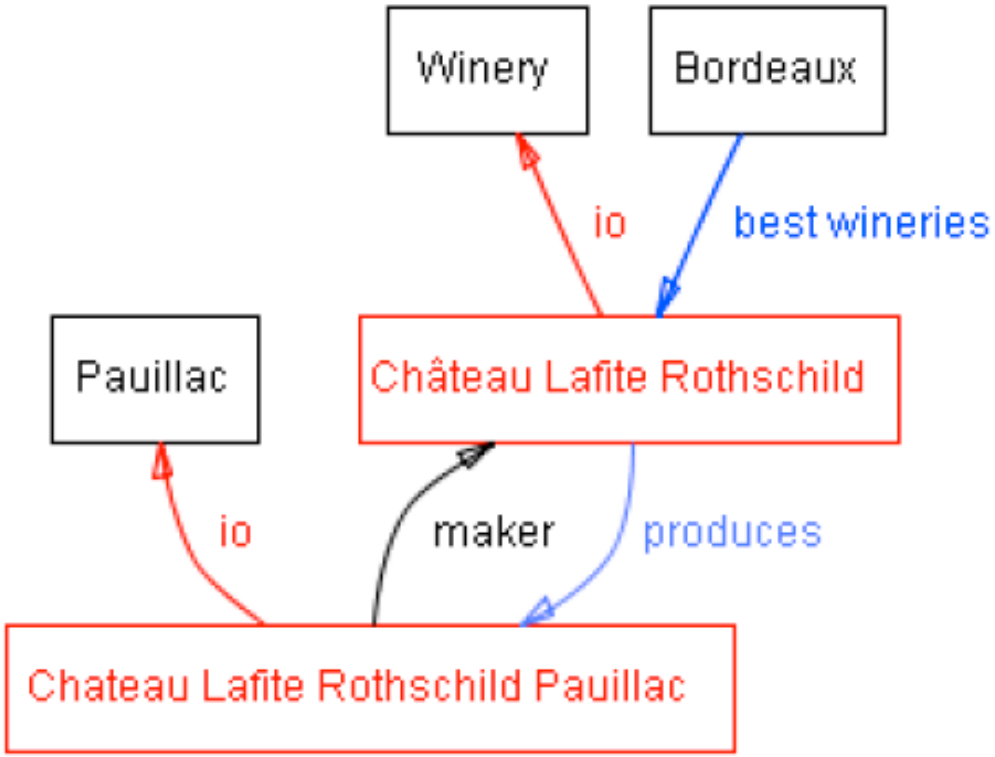
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

$\rightarrow$ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



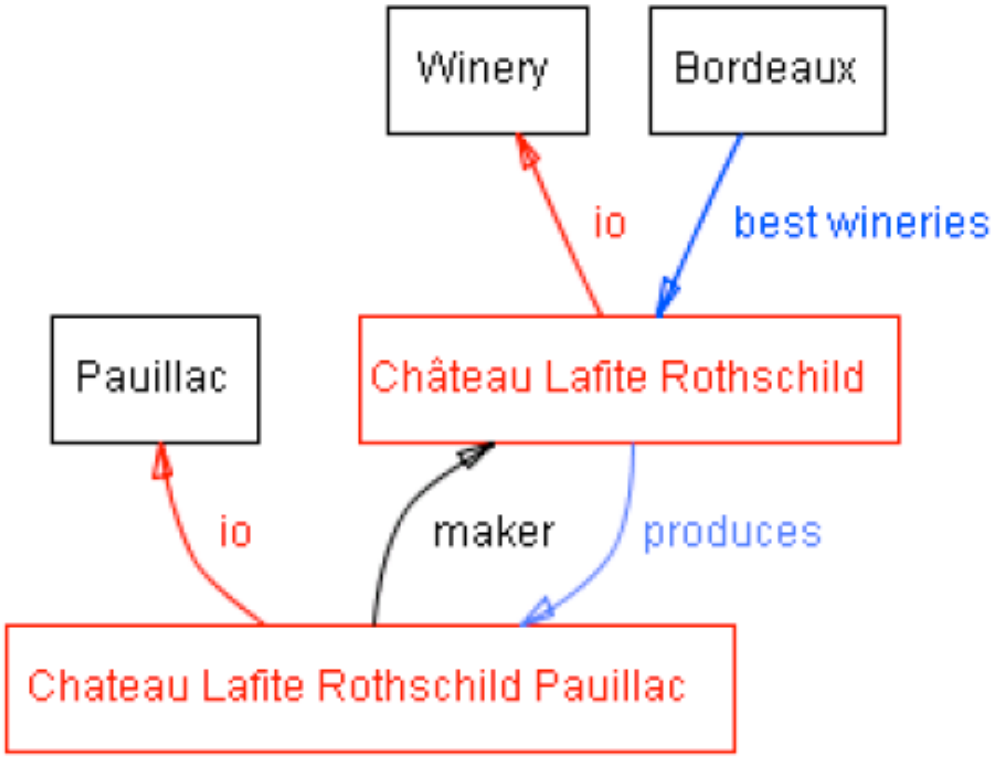
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

$\rightarrow$ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



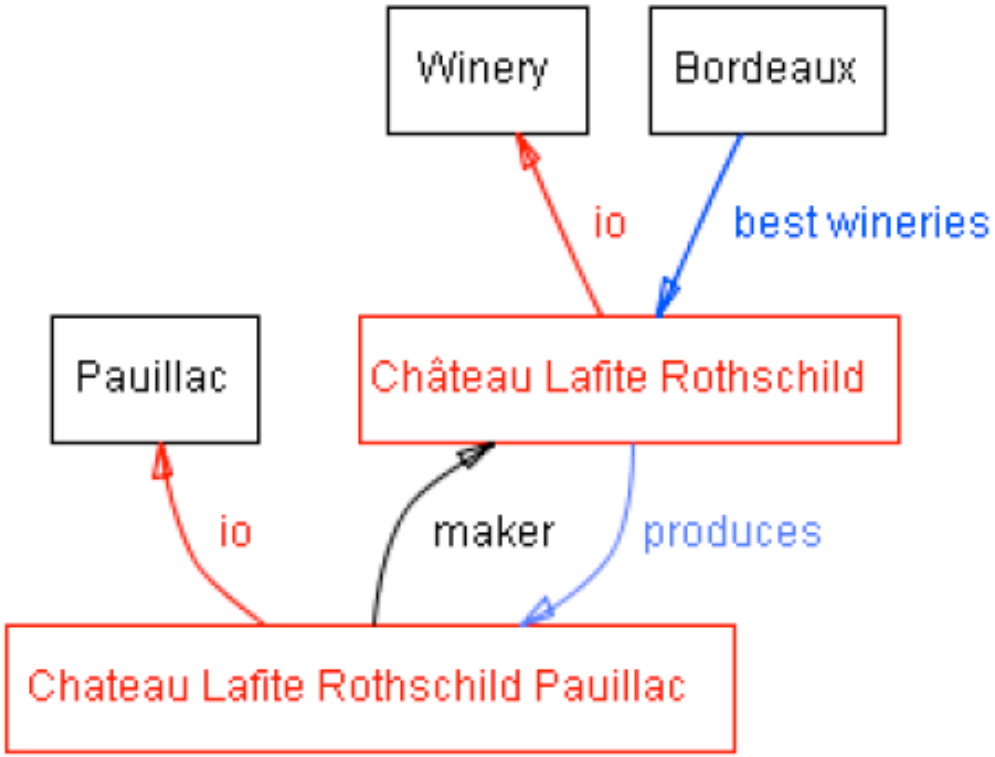
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

→ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



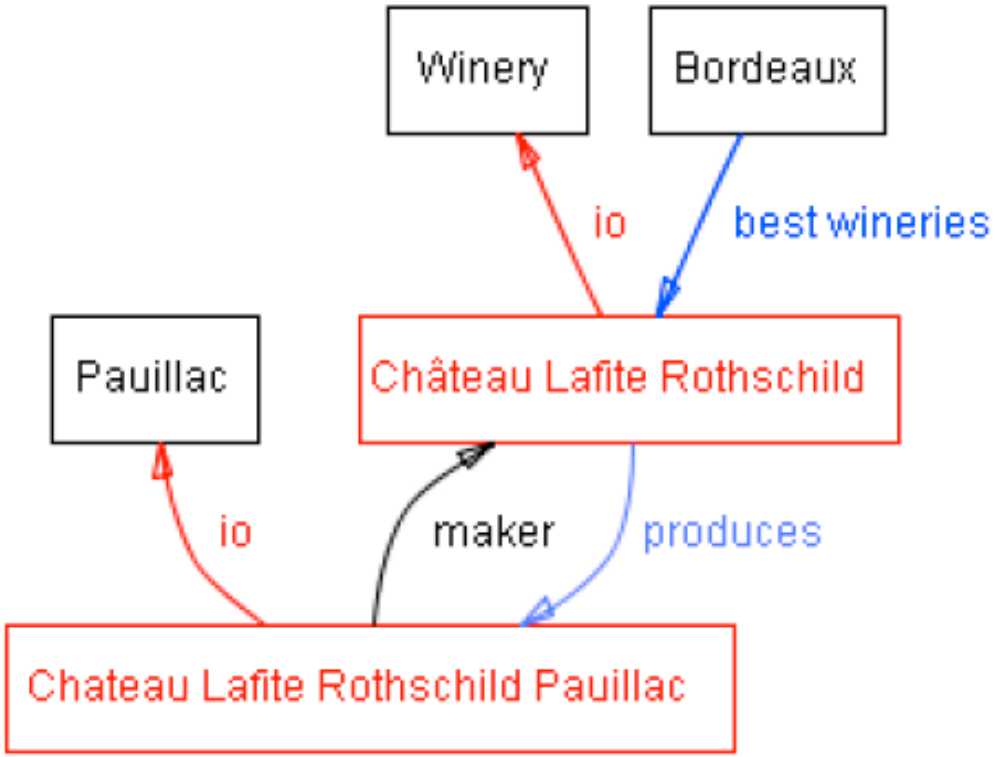
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

→ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



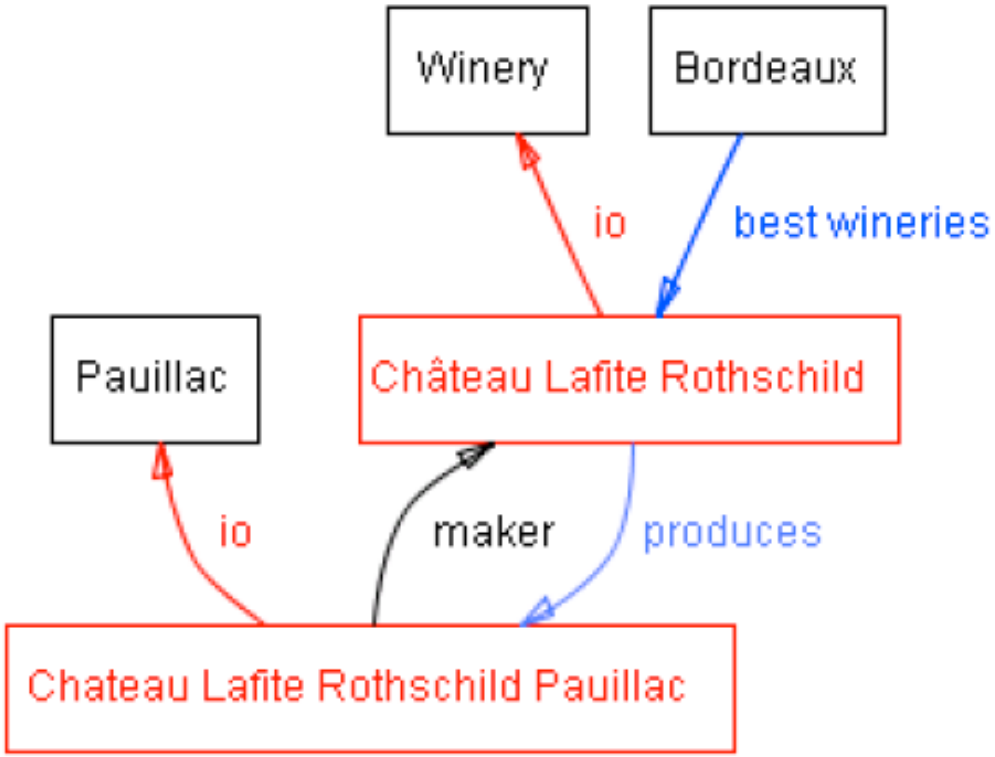
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

→ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



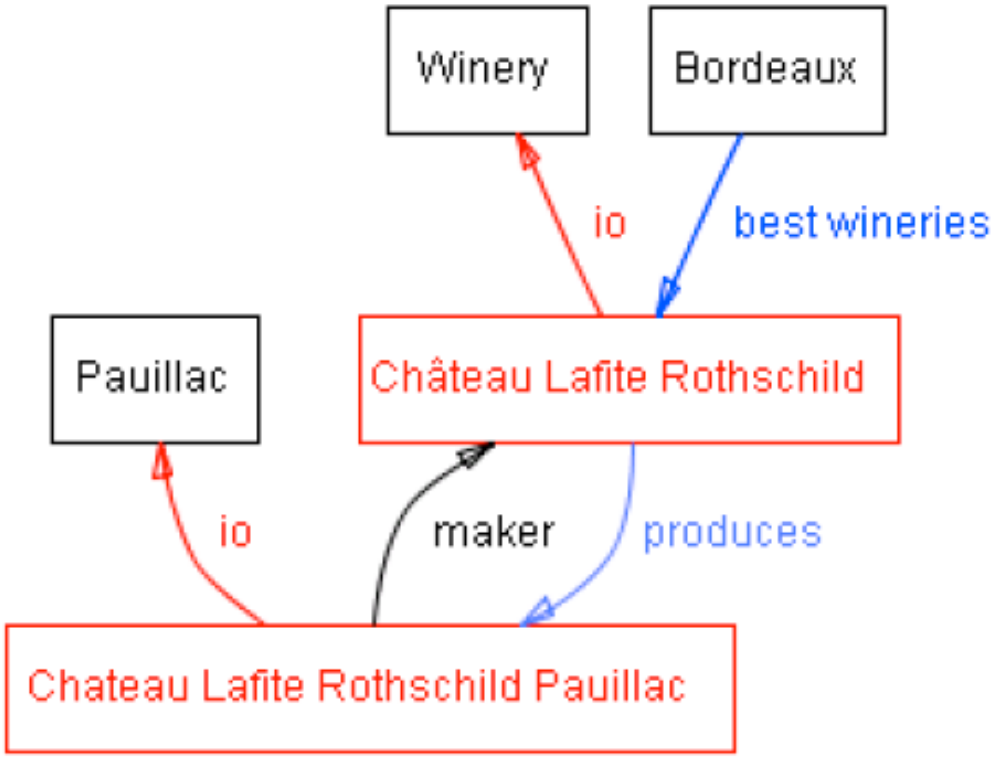
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

→ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



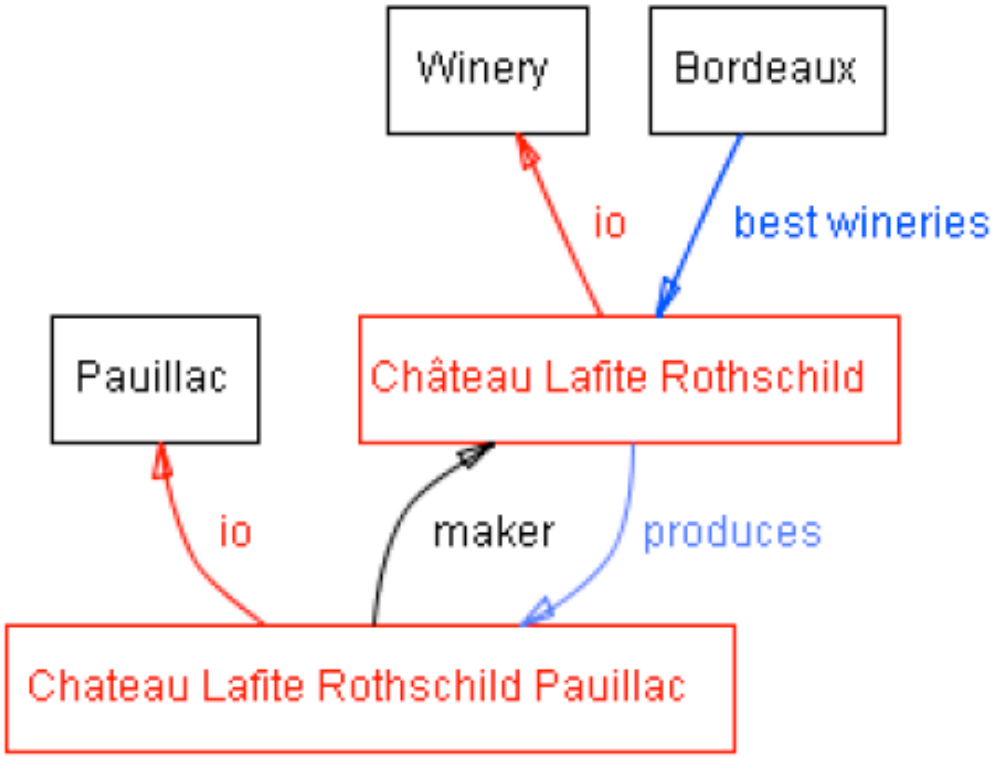
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

→ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



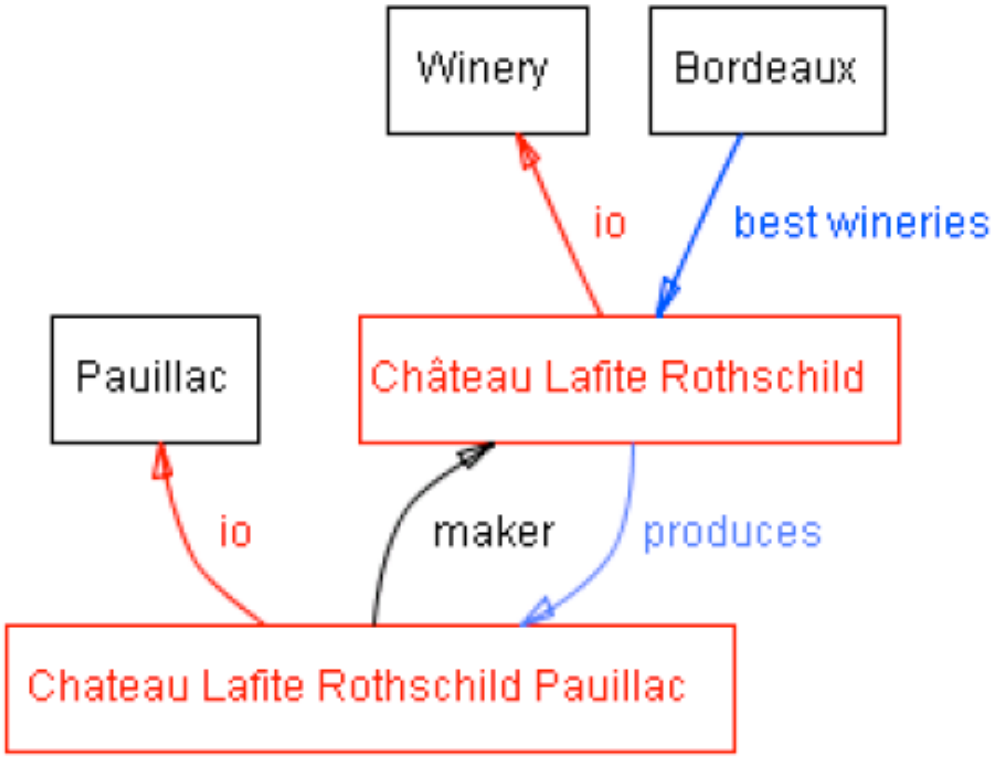
Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

→ An ontology should contain enough information to answer these questions.

Step 1: Example

The Wine Ontology (very simplified ...)



Competency questions:

- Which wine characteristics should I consider when choosing a wine?
- Is Bordeaux a red or white wine?
- Does Cabernet Sauvignon go well with seafood?
- What is the best choice of wine for grilled meat?
- Which characteristics of a wine affect its appropriateness for a dish?
- Does a bouquet or body of a specific wine change with vintage year?
- What were good vintages for Napa Zinfandel?

$\rightarrow$ An ontology should contain enough information to answer these questions.

Step 2: Consider reusing existing ontologies

- **Reuse** $\rightarrow$ Sure, if they exist!!!
- Profit from importing existing ontologies — not only cover the scope you need (to some extent), but they also include additional information, classification, axiomatization, etc.
- Check if existing ontologies fit your needs
  - Can you use them directly?
  - Can you use part of them?
  - Maybe only small extension will do?
- There are libraries of reusable ontologies.

If it does not work or fit you — design your own!

Step 2: Consider reusing existing ontologies

- **Reuse** $\rightarrow$ Sure, if they exist!!!
- Profit from importing existing ontologies — not only cover the scope you need (to some extent), but they also include additional information, classification, axiomatization, etc.
- Check if existing ontologies fit your needs
  - Can you use them directly?
  - Can you use part of them?
  - Maybe only small extension will do?
- There are libraries of reusable ontologies.

If it does not work or fit you — design your own!

Step 3: Enumerate important terms

- What are the terms we would like to talk about?
- What are the properties that connect those terms?
- What would we like to say about those terms?
- Example for wine related terms:
  - Wine, grape, winery, location, etc.
  - A wine's color, body, flavor and sugar content
  - Different types of food such as fish and red meat
  - Subtypes of wine such as white wine, etc.

$\rightarrow$ initially, it is important to get a comprehensive list of terms without worrying about overlapping

Step 4: Define the classes and hierarchy

- Methods
  - Top-down:
    Starts with the definition of the most general terms and subsequently specialize concepts
  - Bottom-up:
    Starts with the definition of the most specific terms
  - combination
- Start from defining classes
- Creating hierarchy will then be easier ...

Step 4: Define the classes and hierarchy (2)

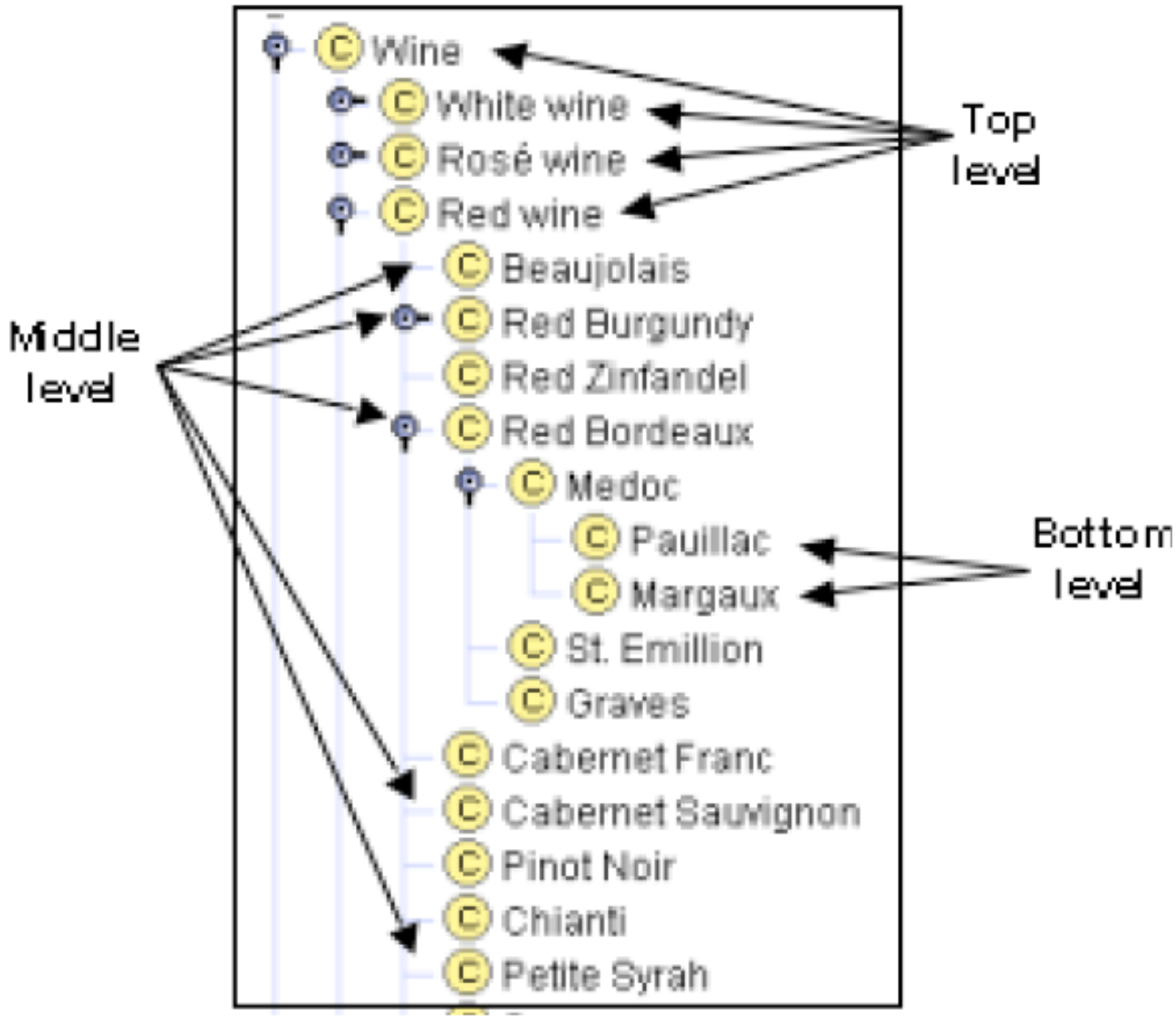
- Result is a
  - hierarchical arrangement of concepts
- If a class A is a superclass of class B, then every instance of B is also an instance of A
- This implies: 'In other words, the class B represents a concept that is a "kind of" A.'

Step 4: Define the classes and hierarchy (3)

- Top-down
  - Food and Wine — followed by White, Blush and Red
- Bottom-up
  - Define specific wine class first and the work your way up
- combination
  - Start from known classes and fill the gaps

Step 4: Define the classes and hierarchy (4)

Levels of generality:



Step 5: Define properties of classes — slots

Types of properties

- "Intrinsic" properties (essential properties) such as the flavor of a wine
- "Extrinsic" properties such as a wine's name and area it comes from
- Parts, if the object is structured; these can be both physical and abstract "parts" (e.g., the courses of a meal)
- Relationships to other individuals — between individual members of the class and other items (e.g., the maker of a wine, representing a relationship between a wine and a winery, and the grape the wine is made from.)

Step 5: Define properties of classes — slots

Types of properties

- "Intrinsic" properties (essential properties) such as the flavor of a wine
- "Extrinsic" properties such as a wine's name and area it comes from
- Parts, if the object is structured; these can be both physical and abstract "parts" (e.g., the courses of a meal)
- Relationships to other individuals — between individual members of the class and other items (e.g., the maker of a wine, representing a relationship between a wine and a winery, and the grape the wine is made from.)

Step 5: Define properties of classes — slots

Types of properties

- "Intrinsic" properties (essential properties) such as the flavor of a wine
- "Extrinsic" properties such as a wine's name and area it comes from
- Parts, if the object is structured; these can be both physical and abstract "parts" (e.g., the courses of a meal)
- Relationships to other individuals — between individual members of the class and other items (e.g., the maker of a wine, representing a relationship between a wine and a winery, and the grape the wine is made from.)

Step 5: Define properties of classes — slots

Types of properties

- "Intrinsic" properties (essential properties) such as the flavor of a wine
- "Extrinsic" properties such as a wine's name and area it comes from
- Parts, if the object is structured; these can be both physical and abstract "parts" (e.g., the courses of a meal)
- Relationships to other individuals — between individual members of the class and other items (e.g., the maker of a wine, representing a relationship between a wine and a winery, and the grape the wine is made from.)

Step 5: Define properties of classes — slots

Types of properties

- "Intrinsic" properties (essential properties) such as the flavor of a wine
- "Extrinsic" properties such as a wine's name and area it comes from
- Parts, if the object is structured; these can be both physical and abstract "parts" (e.g., the courses of a meal)
- Relationships to other individuals — between individual members of the class and other items (e.g., the maker of a wine, representing a relationship between a wine and a winery, and the grape the wine is made from.)

Step 5: Define properties of classes — slots (2)

- Be aware: different naming for the same thing!
  - Relationships
  - Property
  - Slot
- Examples of properties of a class “Wine”:
  - a wine’s
    - color
    - body
    - flavor
    - sugar content
  - location of a winery

Step 6: Define the facets of the slots

- Facets of relationships
  - means: Role restrictions
- Sample facets
  - Value type (e.g., value type of name is string)
  - Allowed values
  - Number of the values (cardinality – single, multiple ...)
  - ... other features of the values the slot can take

Step 6: Define the facets of the slots (2)

Some rules for domain and range:

- When defining a domain or range ...
  - Find the most general classes or class that can be respectively the domain or the range for the slots
- Do not define a domain and range that is overly general
  - All the classes in the domain of a slot should be described by the slot and
  - Instances of all the classes in the range of a slot should be potential fillers for the slot.
- Do not choose an overly general class for range (i.e., one would not want to make the range 'Thing') but one would want to choose a class that will cover all fillers
- Shortly: aim well – not too general, not too specific!

Step 6: Define the facets of the slots (3)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot includes a class and its subclass, remove the subclass.
  → keep only the most general classes

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (3)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot includes a class and its subclass, remove the subclass.
  $\rightarrow$ keep only the most general classes

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  $\rightarrow$ if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (3)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot includes a class and its subclass, remove the subclass.
  → keep only the most general classes

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (3)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot includes a class and its subclass, remove the subclass.
  → keep only the most general classes

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (4)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot contains not all but some subclasses of a class A, consider if the class A would make a more appropriate range definition.
  - → if you have not all but few subclasses, use superclass
  - → or give a second thought to the created hierarchy

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  - → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (4)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot contains not all but some subclasses of a class A, consider if the class A would make a more appropriate range definition.
  → if you have not all but few subclasses, use superclass
  → or give a second thought to the created hierarchy

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (4)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot contains not all but some subclasses of a class A, consider if the class A would make a more appropriate range definition.
  - → if you have not all but few subclasses, use superclass
  - → or give a second thought to the created hierarchy

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  - → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (4)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot contains not all but some subclasses of a class A, consider if the class A would make a more appropriate range definition.
  - → if you have not all but few subclasses, use superclass
  - → or give a second thought to the created hierarchy

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  - → if you have all subclasses, use only the superclass

Step 6: Define the facets of the slots (4)

Some rules for domain and range (more specific):

- If a list of classes defining a range or a domain of a slot contains not all but some subclasses of a class A, consider if the class A would make a more appropriate range definition.
  - → if you have not all but few subclasses, use superclass
  - → or give a second thought to the created hierarchy

- If a list of classes defining a range or a domain of a slot contains all subclasses of a class A, but not the class A itself, the range should contain only the class A and not the subclasses.
  - → if you have all subclasses, use only the superclass

Step 7: Create Instances

- Body: light-1
- Color: red-1
- Flavor: delicate-1
- Tannin level: low-1
- Grape: gamay-1 (instance of the Wine grape class)
- Maker: chateau-morgon-1 (instance of the Winery class)
- Region: beaujolais-1 (instance of the Wine-Region class)
- Sugar: dry-1

Are we done?

What about
Consistency
Validity
Sanity
checks???

Step 8: Define classes and class hierarchies

- Ensuring that the class hierarchy is correct
  - "is-a" relation: a subclass of a class represents a concept that is a "kind of" the concept that the superclass represents
  - A single wine is not a subclass of all wines

  → this typically occur if singular and plural names are used, e.g., Wine is a subclass of Wines.

  → To avoid this: use only singular

- Remember about **transitivity** of hierarchical relations.
  - For instance: define a class White wine as a subclass of Wine. Then we define a class Chardonnay as a subclass of White wine. Transitivity of the subclass relationship means that the class Chardonnay is also a subclass of Wine. Chardonnay is a **direct** subclass (i.e., the closest subclass) of White wine and is not a direct subclass of Wine.

Step 8: Define classes and class hierarchies (2)

- **Evolution of a class hierarchy**
  - Distinction between **classes** and their **names**
    - Hence synonyms of concept name do not represent different classes
  - Avoid class hierarchy **cycles**
  - All the siblings in the hierarchy (except for the ones at the root) must be at the same level of generality

Step 8: Define classes and class hierarchies (3)

- **Analyzing siblings in a hierarchy**
- How many are too many and how few is are too few?
  - If a class has only **one direct subclass** there may be **a modeling problem** or the ontology is incomplete
  - If there are **more than a dozen subclasses** for a given class then **additional intermediate categories** may be necessary. (may not always be possible!)

Step 8: Define classes and class hierarchies (3)

- **Multiple inheritance**
- Use it to combine properties of both (or many) classes within one
- When do you introduce a new class?
  - Subclasses of a class usually
    1. have additional properties that the superclass does not have, or
    2. different restrictions from those of the superclass, or
    3. participate in different relationships than the superclasses

Step 8: Define classes and class hierarchies (4)

- **Multiple inheritance – counterexample to the rule for "When do you introduce a new class?"**
  - Classes in terminological hierarchies do not have to introduce new properties
- Example: an ontology underlying an electronic medical-record system may include a classification of various diseases. This classification may be just a hierarchy of terms, without properties (or with the same set of properties). $\rightarrow$ In that case, it is still useful to organize the terms in a hierarchy rather than a flat list.
  Reasons:
  - easier to explore and navigate
  - easier / better selection with respect to concept granularity

Step 8: Define classes and class hierarchies (4)

- **Multiple inheritance – counterexample to the rule for "When do you introduce a new class?"**
  - Classes in terminological hierarchies do not have to introduce new properties
- Example: an ontology underlying an electronic medical-record system may include a classification of various diseases. This classification may be just a hierarchy of terms, without properties (or with the same set of properties). $\rightarrow$ In that case, it is still useful to organize the terms in a hierarchy rather than a flat list.
  Reasons:
  - easier to explore and navigate
  - easier / better selection with respect to concept granularity

Step 8: Define classes and class hierarchies (5)

- **A new class or property value**
  - Class White wine or simply a property like color of class Wine that takes value white-1?
  - Depends on how **important** a concept White wine is in the domain
  - Some principles:
    - numbers, colors location (at least in the wine example) are properties

Step 8: Define classes and class hierarchies (5)

- **A new class or property value**
  - Class White wine or simply a property like color of class Wine that takes value white-1?
  - Depends on how **important** a concept White wine is in the domain
  - Some principles:
    - numbers, colors location (at least in the wine example) are properties

Step 8: Define classes and class hierarchies (5)

- **An instance of a class**
  - Starts with deciding what is the lowest level of granularity in the representation
  - Individual instances are the most specific concepts represented in a knowledge base.
  - Often depends on the point of view, e.g.,
    - Sterling Vineyards Merlot would be an instance if we want to model our favorite wine in a certain restaurant
    - For the restaurant, Sterling Vineyards Merlot would be a class containing several instances

Step 8: Define classes and class hierarchies (6)

- **Limiting the scope**
  - The ontology should not contain all the possible information about the domain:
    - You do not need to specialize (or generalize) more than you need for your application (at most one extra level each way).
    - Tailor ontology for your needs and applications, but ...
    - ... think about possible extensibility (how easy, in which direction)

Step 8: Define classes and class hierarchies (6)

- **Other details of design**
  - Inverse slots
    - Functional or non-functional properties
    - What do you express with inverse relation?
  - Naming conventions
    - Are there available/well-established in general or in your field/area?
    - Stick to one naming convention – be consistent
    - Use existing vocabularies
  - Synonyms
    - Just different name or really different objects?
    - Maybe multiple labels for the same object?
  - Defaults
    - What values are there in case the user do not give any?
  - Capitalization and delimiters
    - Some systems allow spaces in concept names
  - Disjoint subclasses
    - What is the reason for introducing additional restrictions?

Conclusion

- There are steps worth following in designing ontology
- Have rationale for each of your design choices!
- Remember: there is no single correct ontology for modeling any domain
  - "Ontology design is a creative process and no two ontologies designed by different people would be the same."
- Designing ontology is an iterative process
- Ontology can evolve and change while you design it
- There are helpful methodologies and patterns:
  - `http://ontologydesignpatterns.org/`

Learning Goals

- Qualify in the engineering and design of an ontology.
- Reflect on different aspects when creating an ontology.