

CEID  
**MSc on DATA DRIVEN COMPUTING AND  
DECISION MAKING (DDCDM)**

# ONTOLOGIES

I. HATZILYGEROUDIS, PROFESSOR EMERITUS

# ONTOLOGY DESIGN

(based on article:

Ontology Development101: A Guide to Creating Your  
First Ontology:

[http://protege.stanford.edu/publications/ontology\\_development/ontology101.pdf](http://protege.stanford.edu/publications/ontology_development/ontology101.pdf) )

# Ontology-Definition

- Explicit specification of a conceptualization.
- **Conceptualization** concerns the conceptual (abstract) model of a knowledge domain (e.g. disease), that is the concepts that make it up.
- **Explicit specification** concerns the relationships between concepts, the definition of the properties of the concepts, the restrictions on the properties.

# Ontology-Definition

- It includes definitions of key concepts of the domain knowledge and the relationships between them.
- It defines a common vocabulary for a community of people with common cognitive interests.
- Ontologies as conceptualizations are products of subjective judgment, so the same field of interest can be described in different ways.
- The descriptions are made in a strict (formal) way, so that they can be represented on the computer.

# Ontologies-Reasons for creation

- Shared understanding of information structure between humans or machines

(E.g. different websites based on a common ontology enable the search for information from the global knowledge through agents).

- Reusing the knowledge of a field

(A well-defined ontology of a field can be reused to define the ontology of a wider field).

- Clear definition of assumptions in the ontology implementation of a domain

(Makes ontology changes easier when assumptions need to change).

# Ontologies-Reasons for creation

- Separating static knowledge of a domain from procedural knowledge

(A procedure can be used as such for different ontologies).

- Analysis of knowledge

(The analysis of knowledge is made possible by the rigorous identification of concepts and their relationships).

# Structural Ontology Elements

- **Classes.** Classes represent concepts of the field. A concept can be anything about which something can be said, and it can be about a task, a function, an action, an idea, etc.
- **Slots or Properties or Roles.** They represent various features/attributes of the classes.
- **Facets or Restrictions.** Restrictions on properties/roles.
- **Instances.** Instances represent specific entities that belong to classes.
- Classes and instances form the **knowledge base** of an application.

# Classes

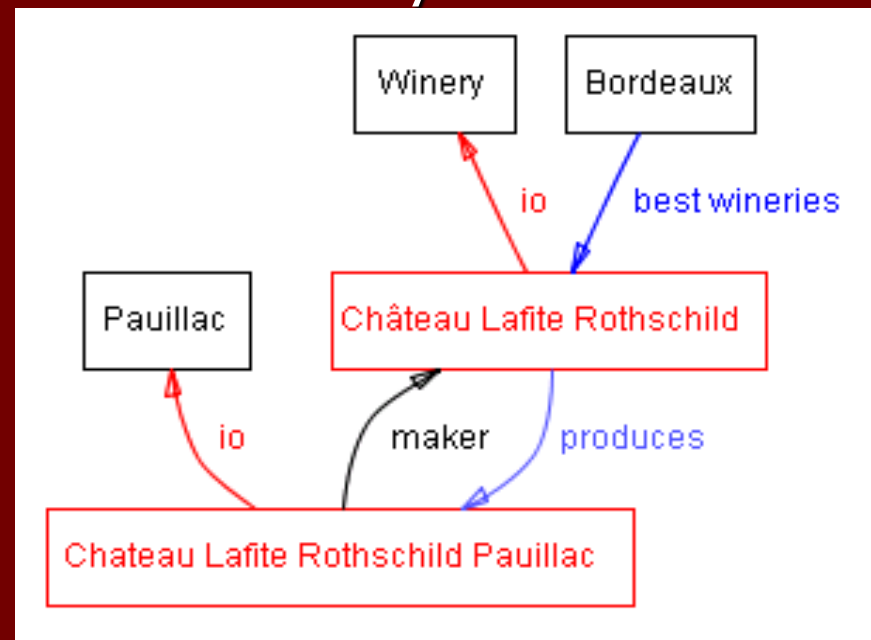
- The most basic elements of an ontology correspond to concepts, which represent sets of entities.  
(E.g. class **Wine** represents all wines)
- They consist of properties, which are described by constraints.  
(E.g. **maker** is a property of **Wine** and **Winery** is a restriction on the value of maker - forces it to take as values entities/instances belonging to the Winery class)
- They are organized in **hierarchies** and related to each other with the **subclass** relationship.
- A subclass is a **specialization** of a class, i.e. it represents a subset of entities of the class.  
(E.g. **RedWine** can be a subclass of **Wine**, representing red wines, a subset of wines).



# Creating an ontology...

... basically, we need

- Definitions of classes.
- Taxonomy of classes (sub-classes)
- Definitions of properties and the values they can take.



The figure shows some classes (in black frame), instances (in red frame) and relationships between them in the wine field. Direct links represent properties and internal links, such as "instance-of" and "subclass-of".

# Basic rules of ontology design

- **Rule 1.** There is not a single "right" way to model a domain of knowledge—there are always alternative "right" ways. The best solution almost always depends on the application and possible extensions.
- **Rule 2.** Developing an ontology is necessarily an iterative process: frequent redesign is required.
- **Rule 3.** Concepts in the ontology must relate to entities (physical or logical) and relationships between them in the cognitive domain of application. These are more likely to be nouns (entities) or verbs (relations) in sentences describing the knowledge domain.

# Steps of building an ontology (1)

## ■ 1: Defining the scope and purpose of the ontology (scope)

- What is the field that the ontology will cover?
- Why are we going to use the ontology?
- What kinds of questions will the ontology provide answers to?
- Who will use and maintain this ontology?

## Ontology of wines

- Suggestions of good wine-food pairings
- Concepts: wines, foods, good pairings (not warehouse or cost management concepts)

# Steps of building an ontology (2)

- 2: Ability to reuse existing ontology (reuse)
  - Research for the existence of an ontology related to the under creation one.
  - Using reusable ontology libraries.
  - Incorporation of existing ontology into the one being created.

## Ontology of wines

To create the wines ontology, if there is a ready/made French wines ontology, we can use its knowledge base as a starting point. Then we not only have a categorization of French wines but also a first idea of the characteristics of the wines which differentiate them from one another.

# Steps of building an ontology (3)

- 3: Enumeration of ontology elements.

-Recording all terms that may be useful for the creation of the ontology under construction, without considerations of propriety. In the next steps will be selected and sorted accordingly.

## Ontology of wines

For this ontology we can record terms such as wine, winery, grape, location, colour, sugar rate, food accompaniment.

# Steps of building an ontology (4)

## ■ 4: Defining the classes and class hierarchy

-We determine which concepts are classes and which are properties. We also define "subclass" relationships between classes and then develop the class hierarchy based on one of the following methods:

- **top-down.** We start from the most general class (which characterizes the ontology) and some of its basic subclasses. Then we specialize the subclasses and so on.
- **bottom-up.** We define the classes of the last level (the leaves of the hierarchy tree), which we organize as subclasses of some classes and those of others above and so on.
- **mixed.** A combination of the two above. One can start top-down and continue bottom-up or vice versa.

# Steps of building an ontology (5)

## Ontology of wines

-E.g. we define **Wine**, **Winery** as classes and **color**, **grape**, **location** as properties.

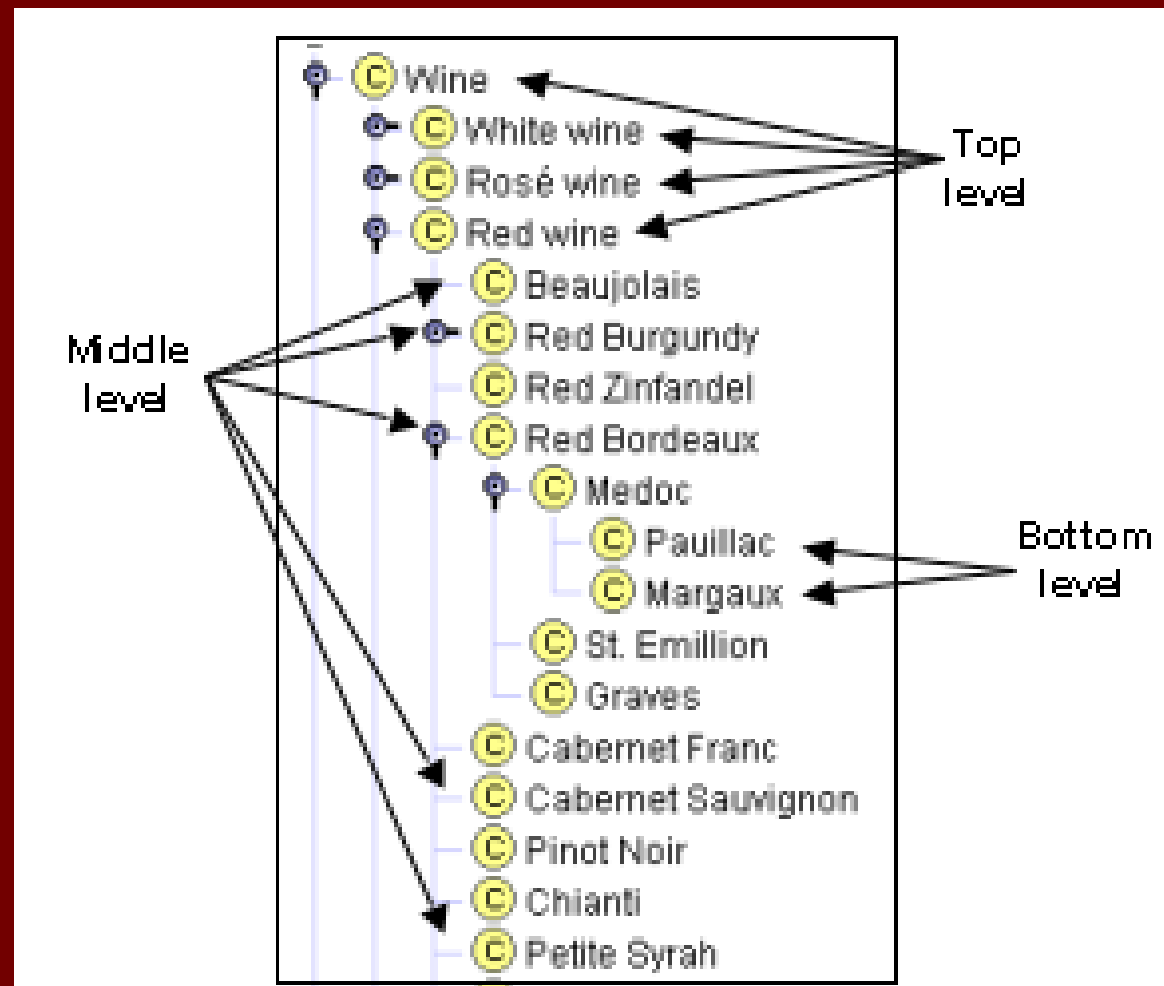
-**top-down**. We start with Wine and consider **Red wine**, **White wine** and **Rose wine** as subclasses. Then we proceed with **Red Burgundy**, **Cabernet Sauvignon** as subclasses of Red wine and so on.

- **bottom-up**. We start e.g. from the classes **Pauillac** and **Margeux**, as terminal classes, and we define them as subclasses of the **Medoc**, which, together with the **St. Emillion** and **Graves**, we define as subclasses of **Red Bordeaux**.

-**mixed**. We start from some intermediate classes, e.g. Red Burgundy, Red Bordeaux, and we proceed bottom-up by defining them as subclasses of Red wine, and continue top-down by creating St. Emillion and Medoc as subclasses of Red Bordeaux and so on.

# Steps of building an ontology (6)

- Part of wine ontology in protege (older version)





# Steps of building an ontology (7)

- The use of the above methods is based on **subjective criteria**. For example, if one is interested in the differentiation of wines into types (white, red, rosé), then it would be preferable to use top-down development. But if he is interested in which specific elements (wine title), then it is better to use bottom-up development.
- Whichever development is chosen, it is necessary to define those **classes which are independent of others**, thus trying to make a hierarchical classification by checking whether one class is a subclass of another. For example, Pinot Noir wine is a red wine. So it will necessarily be a subclass of the Red Wine class.

# Steps of building an ontology (8)

- 5: Defining the properties of the classes (slots)
  - Properties are determined and distributed to the classes of the hierarchy.
  - Classes alone cannot provide enough information. This is done by defining their properties.
  - In an ontology there are different **kinds of properties**:
    - \* "intrinsic" or "natural" properties (eg flavor)
    - \* "extrinsic" properties (e.g. name, area),
    - \* "parts", if entities of a class are structured (eg stages of a meal)
    - \* "relationships" between independent members of the class and others entities (e.g. Wine maker).

# Steps of building an ontology (9)

- 6: Defining facets of the properties (slots) of the classes

The facets, i.e. the restrictions on the values of the class properties, are described. Facets concern restrictions such as number of values (cardinality), type of values (type), etc. Each aspect has a filler or value.

**-Common Facets:**

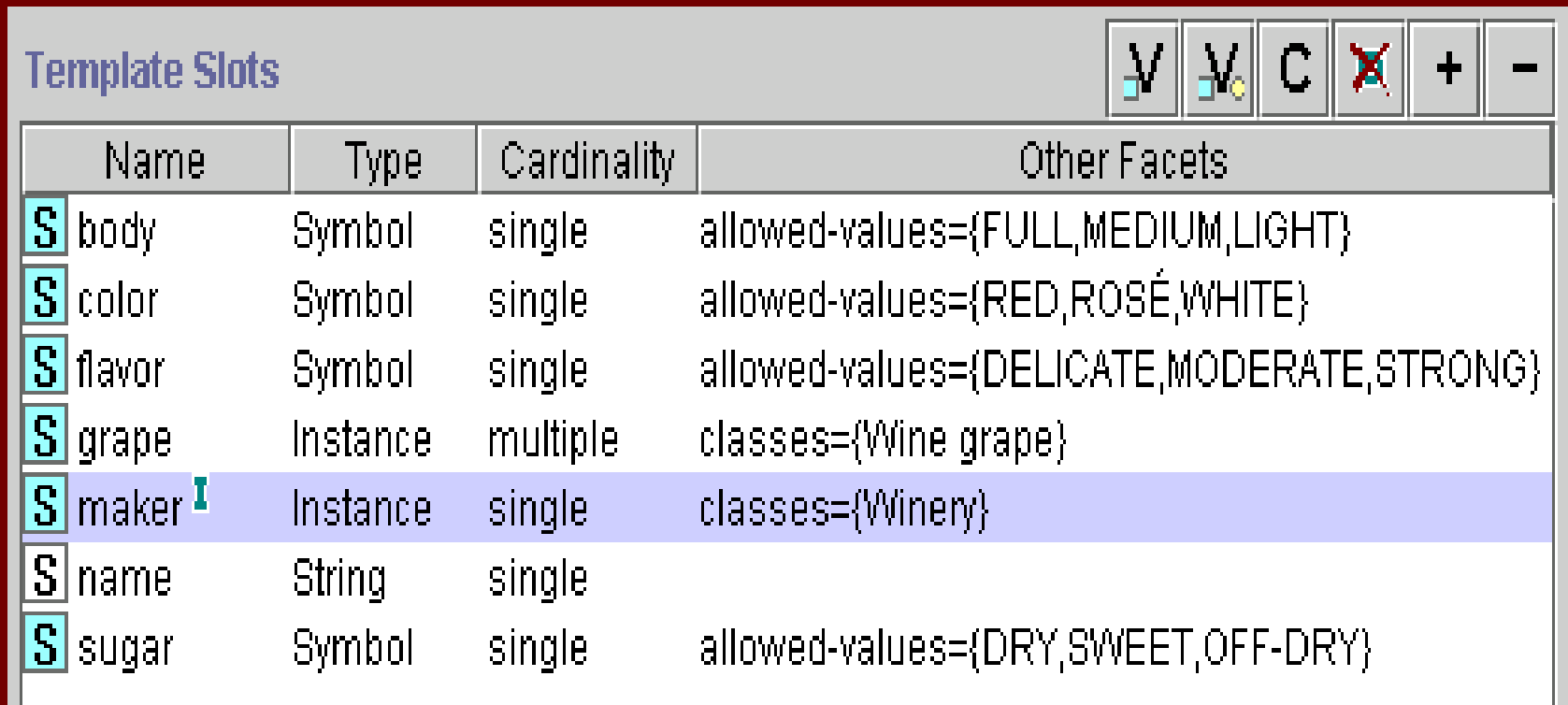
**Slot cardinality.** Defines how many values a property can have. For example, the body property of the Wine class can take only one value (single cardinality slot) since a wine has a type in terms of how strong it is.

**Slot-value type.** Defines the type of values of a property, e.g. String, Number, Boolean, Enumerated (here we list the values: e.g. for flavor it can be strong, moderate, delicate)

**Instance-type slot:** E.g. the produces property, which is of type instance, of the Winery class can have as values instances of the Wine class.

# Steps of building an ontollogy (10)

- Properties (slots) of class **Wine** and their facets, as displayed in protégé (older version)



The screenshot shows the 'Template Slots' window in Protégé. The window title is 'Template Slots'. At the top right, there are several icons: a blue 'V', a blue 'V' with a yellow dot, a black 'C', a red 'X', a plus sign, and a minus sign. Below the icons is a table with the following columns: Name, Type, Cardinality, and Other Facets. The table contains seven rows of slots. The 'maker' slot is highlighted in blue.

Name	Type	Cardinality	Other Facets
<b>S</b> body	Symbol	single	allowed-values={FULL,MEDIUM,LIGHT}
<b>S</b> color	Symbol	single	allowed-values={RED,ROSÉ,WHITE}
<b>S</b> flavor	Symbol	single	allowed-values={DELICATE,MODERATE,STRONG}
<b>S</b> grape	Instance	multiple	classes={Wine grape}
<b>S</b> maker <sup>1</sup>	Instance	single	classes={Winery}
<b>S</b> name	String	single	
<b>S</b> sugar	Symbol	single	allowed-values={DRY,SWEET,OFF-DRY}

# Steps of building an ontology (11)

- The figure below shows the relationship of the **produces** property to the **Winery** class.

The screenshot shows a window titled "produces" with the following configuration:

- Name:** produces
- Documentation:** This slot contains the wines produced by a particular winery
- Value Type:** Instance
- Allowed Classes:** Wine
- Cardinality:**
  - required at least
  - multiple at most

# Steps of building an ontology (12)

## Domain and Range of a property

The classes as values (fillers) in properties of type instance form the **range** of values of the property.

The classes to which a property is attached (or belongs to) constitute the property's **domain**.

General rule: To define domain and range of a property, we find the most general classes that can constitute them.

## Ontology of wines

**produces** is an instance property of the **Winery** class with as value **Wine**. **Wine** (ie the instances of Wine) is the **range** of produces, while **Winery** is the **domain** of produces.

Winery – produces – Wine  
(domain) (range)

# Steps of building an ontology (13)

## ■ 7: Creation of instances

To create an instance of a class: (1) select the class, (2) create an (empty) instance of the class, (3) fill in the values of its properties.

The screenshot shows a software window titled "Chateau Morgon Beaujolais (Beaujolais)". The window contains several input fields and dropdown menus for defining the properties of a wine instance. The properties and their values are as follows:

Property	Value
Name	Chateau Morgon Beaujolais
Area	Beaujolais region
Body	LIGHT
Color	RED
Maker	Chateau Morgon
Flavor	DELICATE
Sugar	DRY
Grape	Gamay grape
Tannin Level	LOW

# Definition and Hierarchy of Classes (1)

## ■ Class Hierarchy Consistency Checkpoints

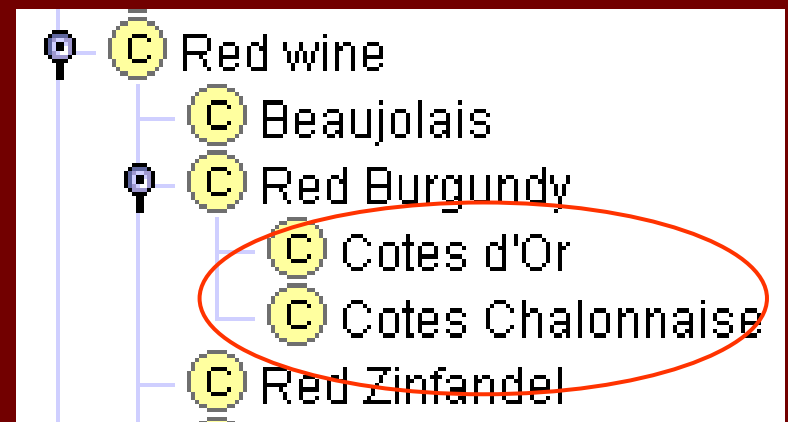
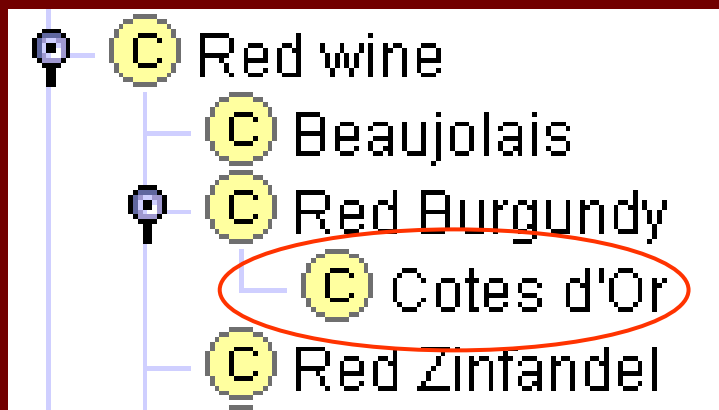
- An ontology is based on the "is-a" relationship: A class A is a subclass of B if any instance of A is also an instance of B. That is, it is a set-subset relationship.
- The "part-of" relationship is not captured as a hierarchy relationship of the ontology, but as a property.
- The "subclass" relationship is transitive: If class B is a subclass of A and C is a subclass of B, then C is also a subclass of A.
- It is recommended to use class names in the singular.
- Avoid using different names for the same class (eg Shrimp-Prawn).
- Avoiding cycles in classes: This means that class A has a subclass B and at the same time B has a subclass of A.



# Definition and Hierarchy of Classes (2)

## ■ Analysis of sibling classes

- Siblings are classes, which are subclasses of the same class and must be at the same level of generality.
- If a class has only one subclass (ie no sibling classes), then there is a modeling problem or the hierarchy is not complete.
- If a class has more than 12 sibling subclasses, then further (intermediate) categorization is needed.



# Definition and Hierarchy of Classes (3)

## **When a new (sub)class needs to be defined:**

Usually the subclass of a class a) has additional properties that the superclass does not have, or b) has different restrictions than the superclass, or c) participates in different relationships than the superclasses.

## **Create a new (sub)class or add an extra property value?**

If adding a value to a property is restrictive for a property of another class, then a new class needs to be created. For example, if we didn't create a White wine class and just fill the value white in the color property, we wouldn't be able to associate white wine with different foods.

# Definition and Hierarchy of Classes (4)

If a distinction between concepts in the domain is important, and we think of objects with different values for that distinction as objects of a different kind, then a class needs to be created for that distinction. E.g. the creation of the White Merlot and Red Merlot classes is different from the creation of a Merlot wine class, since they have big differences (different grapes, etc.)

A class to which an instance belongs to should not change frequently. E.g. Chilled wine does not need to be a class of the ontology, but an attribute of the Wine class. The Chilled wine designation for wine bottles is not permanent, or in other words the chilled state for a bottle of wine is not permanent, so a wine will jump from the chilled state to normal.

# Definition and Hierarchy of Classes (5)

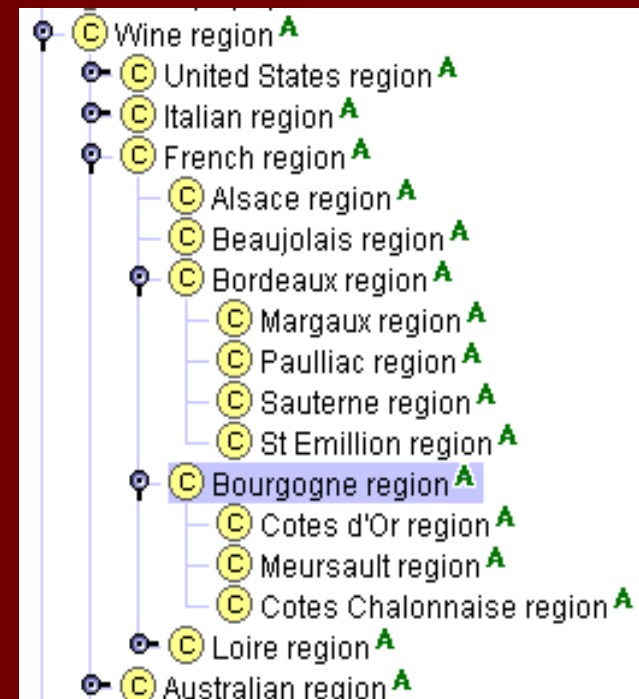
## Instance or class?

A concept can often be modeled as either a class or an instance.

E.g. there is a specific wine Sterling Vineyards Merlot, which can be modeled either as an instance (eg of the Merlot class), or as a class (a subclass of the Merlot class). This depends on the use of the ontology.

If concepts form a natural hierarchy, then they should be represented as classes.

E.g. the hierarchy of wine producing regions (see figure). Usually these classes are defined as **abstract**, i.e. classes without instances.



# Definition and Hierarchy of Classes (6)

When definition of an ontology is sufficient?

Two rules:

- The ontology should not contain all possible information about the domain: it needs not specialize (or generalize) more than is necessary for the application (at most one extra layer).
- The ontology should not contain all possible properties and distinctions between classes in the hierarchy.