COMP318

Ontologies and Semantic Web

Ontology Engineering - Part 3

Dr Valentina Tamma

V.Tamma@liverpool.ac.uk

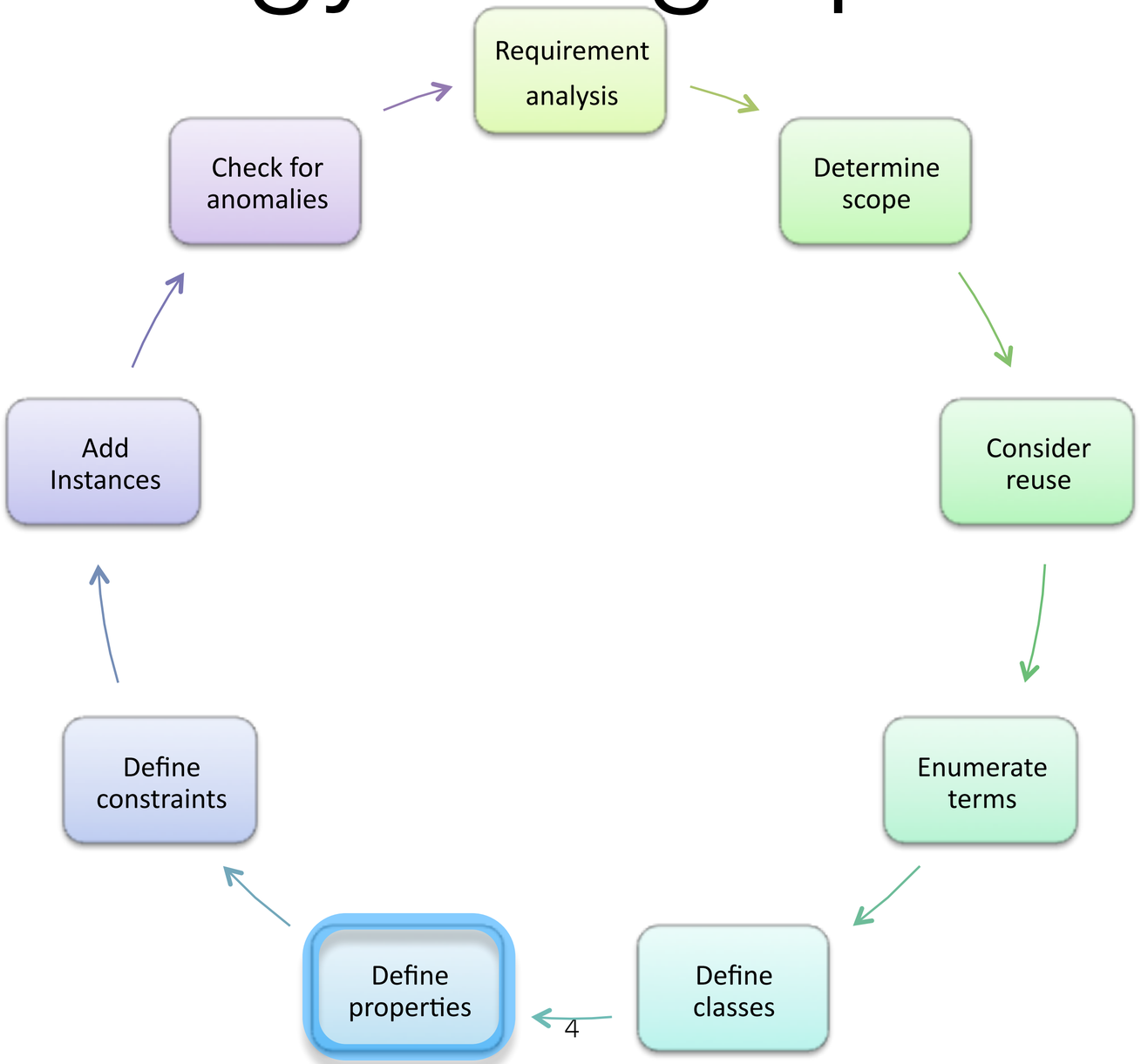
Where were we

- Ontology engineering principles
- Ontology engineering methodologies: Ontology 101

More criteria

- All the siblings must denote concepts at the ***same level*** of generality
  - *similar to sections and subsections in a book*
- If a class has more than a dozen direct superclasses, then it an additional level of generality is needed
  - *compare to bullets in a bullet list*
    - But in some cases, if no natural classification exist, a long list might reflect the reality better.
- Class names should be either singular or plural, don't mix!
  - *Animal is not a kind-of Animals*
- Classes represent concepts in the domain, but names do not
  - a class name can change but the concept represented will still be the same
    - Synonym names for the same concepts refer to different labels, not to different classes

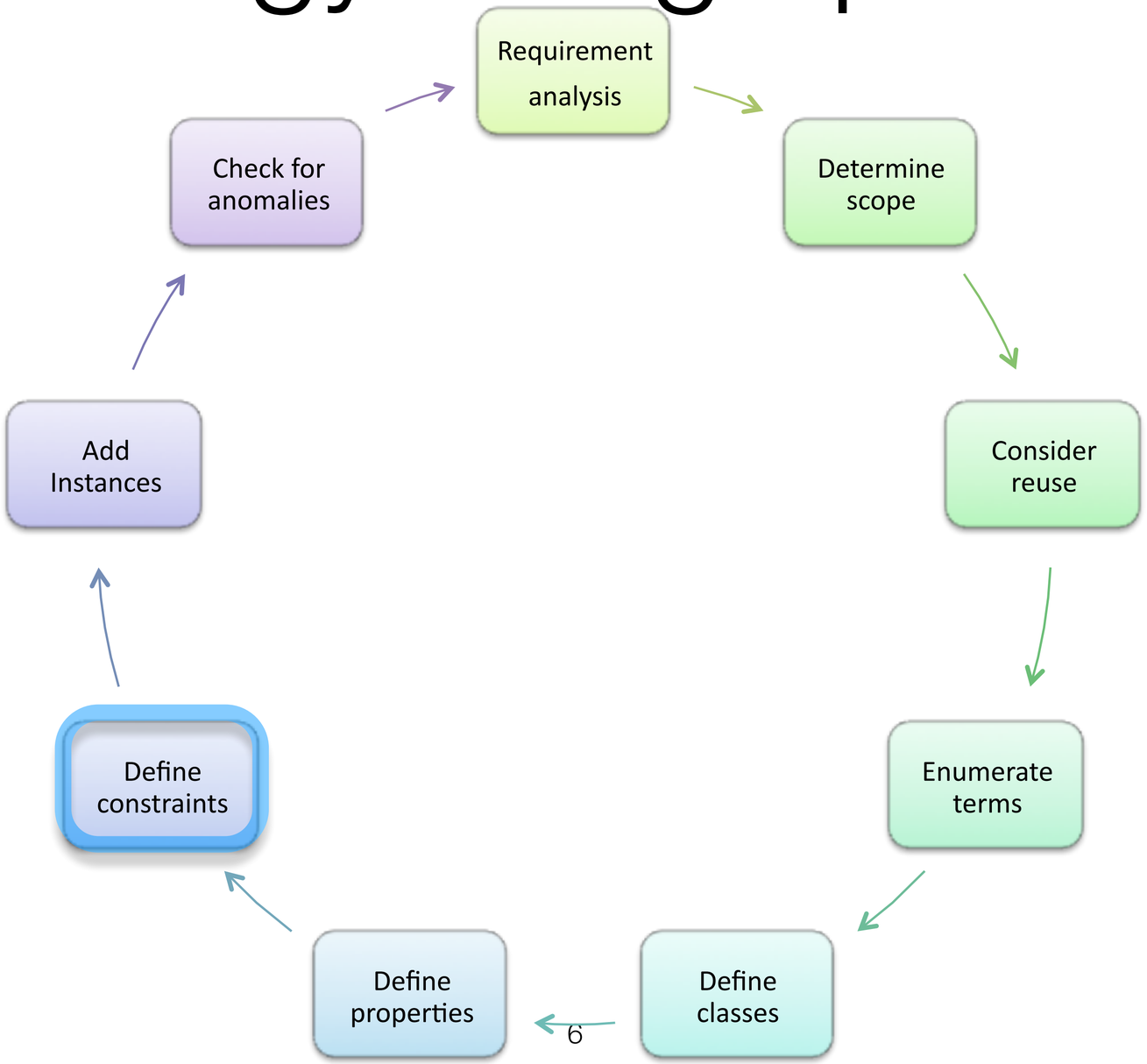
Ontology design process



Define properties

- Often interleaved with the previous step
- Properties (or roles in DL) describe the attributes of the members of a class
  - Defined in terms of domain and range constraints
    - if anything is used in a special way, then add comments
      - *Animal eat LivingThing*, domain: Animal - range: LivingThing
      - *Person owns LivingThing except Person*, domain: Person - range: LivingThing and not Person
      - *Animal parentOf Animal*, domain: Animal - range: Animal
  - Defined in terms of property restrictions
    - What can we say about all instances of a class?
      - *all Cows eat some Plants*
      - *all Cats eat some Animals*
      - *all Pigs eat some Animals and eat some Plants*
- For the semantics of subClassOf whenever A is a subclass of B, every property statement that holds for instances of B must also apply to instances of A
  - It makes sense to attach properties to the highest class in the hierarchy to which they apply

Ontology design process



State constraints: definable things

- Definitions need to be ***paraphrased*** and ***formalised*** in terms of primitive classes, relations and other definable entities
  - Add comments when providing definitions
    - Note any assumptions that need to be represented somewhere else.
  - Paraphrasing needs to achieve consensus on what we meant to represent and how we represent it.

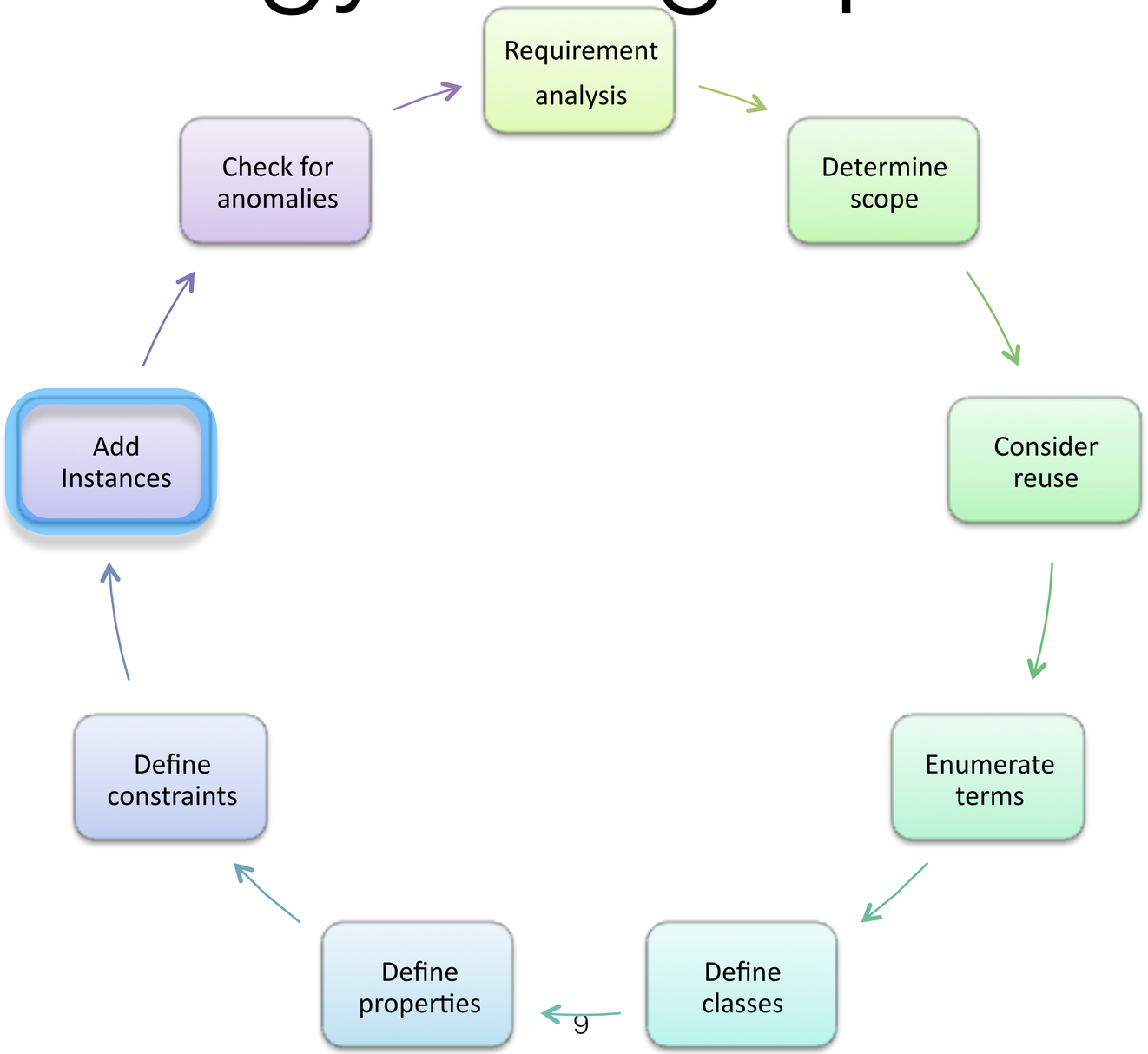
```
:Parent owl:equivalentClass [
  rdf:type         owl:Class;
  owl:intersectionOf (:Animal [
    rdf:type         owl:Restriction ;
    owl:onProperty     :hasChild ;
    owl:someValuesFrom  :Animal .])
 ] .
```

```
:Herbivore owl:equivalentClass [
  rdf:type         owl:Class;
  owl:intersectionOf (:Animal [
  rdf:type         owl:Restriction ;
  owl:onProperty     :eats ;
  /* eats range LivingThing */
  owl:allValuesFrom  :Plant .])
] .
```

State constraints: definable things

- *A **Parent** is an **Animal** that is a parent of some **Animal***
  - `Parent = Animal and parentOf some Animal`
- *A **Herbivore** is an **Animal** that eats only **Plants***
  - assume that Animals eat some LivingThing
  - `Herbivore ≡ Animal and eats only Plant`
- *An **Omnivore** is an **Animal** that eats both **Plants** and **Animals***
  - `Omnivore ≡ Animal and eats some Plant and eats some Animal`

Ontology design process

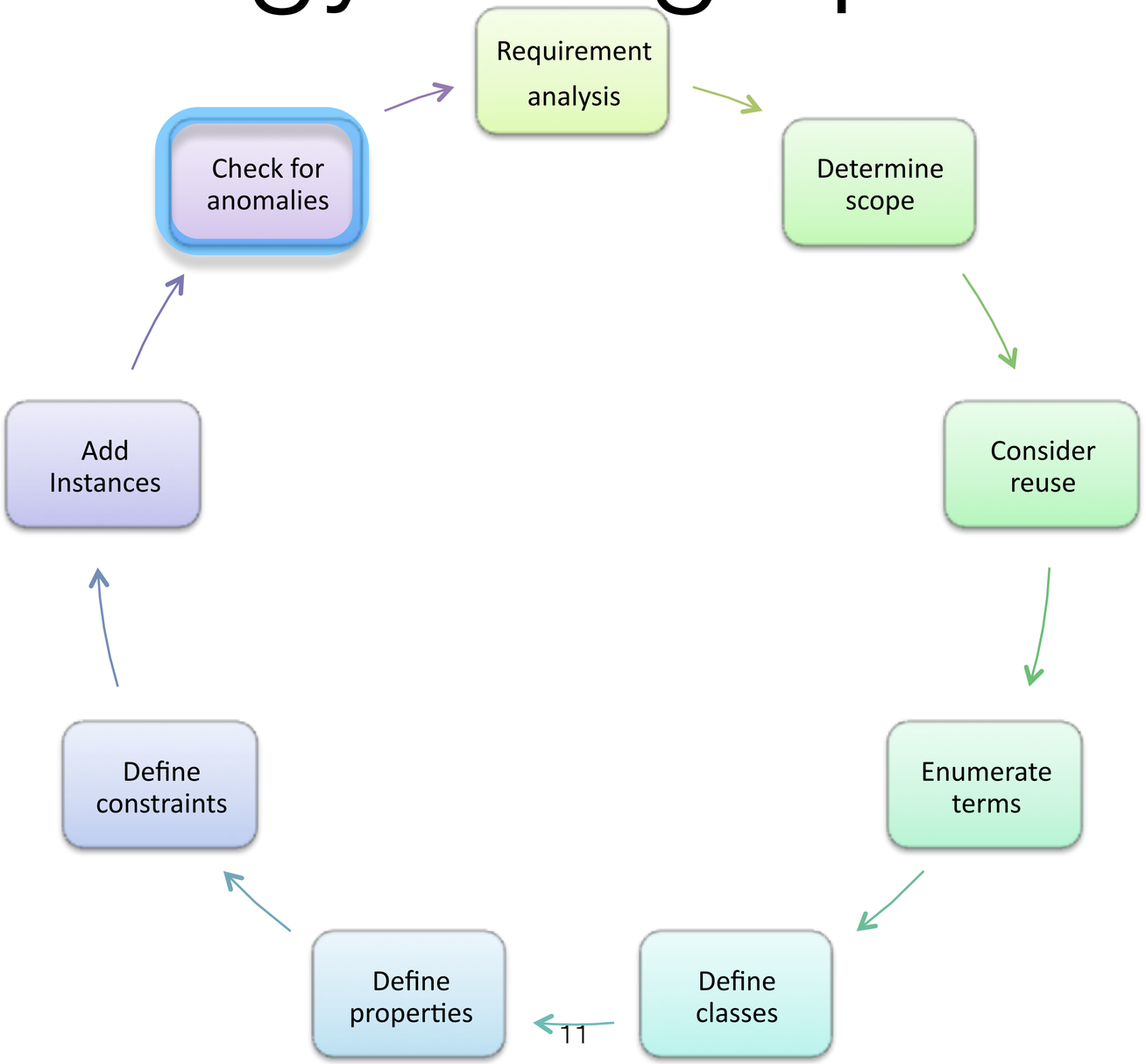


Creating instances

- Create an instance of a class
  - The class becomes a direct type of the instance
  - Any superclass of the direct type is a type of the instance

- Assign property values for the instance description
  - property values should conform to the constraints asserted for the property
  - Knowledge-acquisition tools often check that constraints are satisfied

Ontology design process



Check for anomalies

- An important advantage of the use of OWL over RDF Schema is the possibility to detect inconsistencies
  - In ontology
    - incoherent ontology: at least an unsatisfiable class, class that cannot have any instance
  - In ontology+instances
    - inconsistent ontology: every class is interpreted as the empty set
- Examples of common inconsistencies
  - incompatible domain and range definitions for transitive, symmetric, or inverse properties
  - cardinality properties
  - requirements on property values can conflict with domain and range restriction
- Examples from the Pizza tutorial for Protege
  - http://owl.cs.manchester.ac.uk/tutorials/protegeowltutorial/

COMP318

Ontologies and Semantic Web

End of Ontology Engineering - Part 3

Dr Valentina Tamma

V.Tamma@liverpool.ac.uk