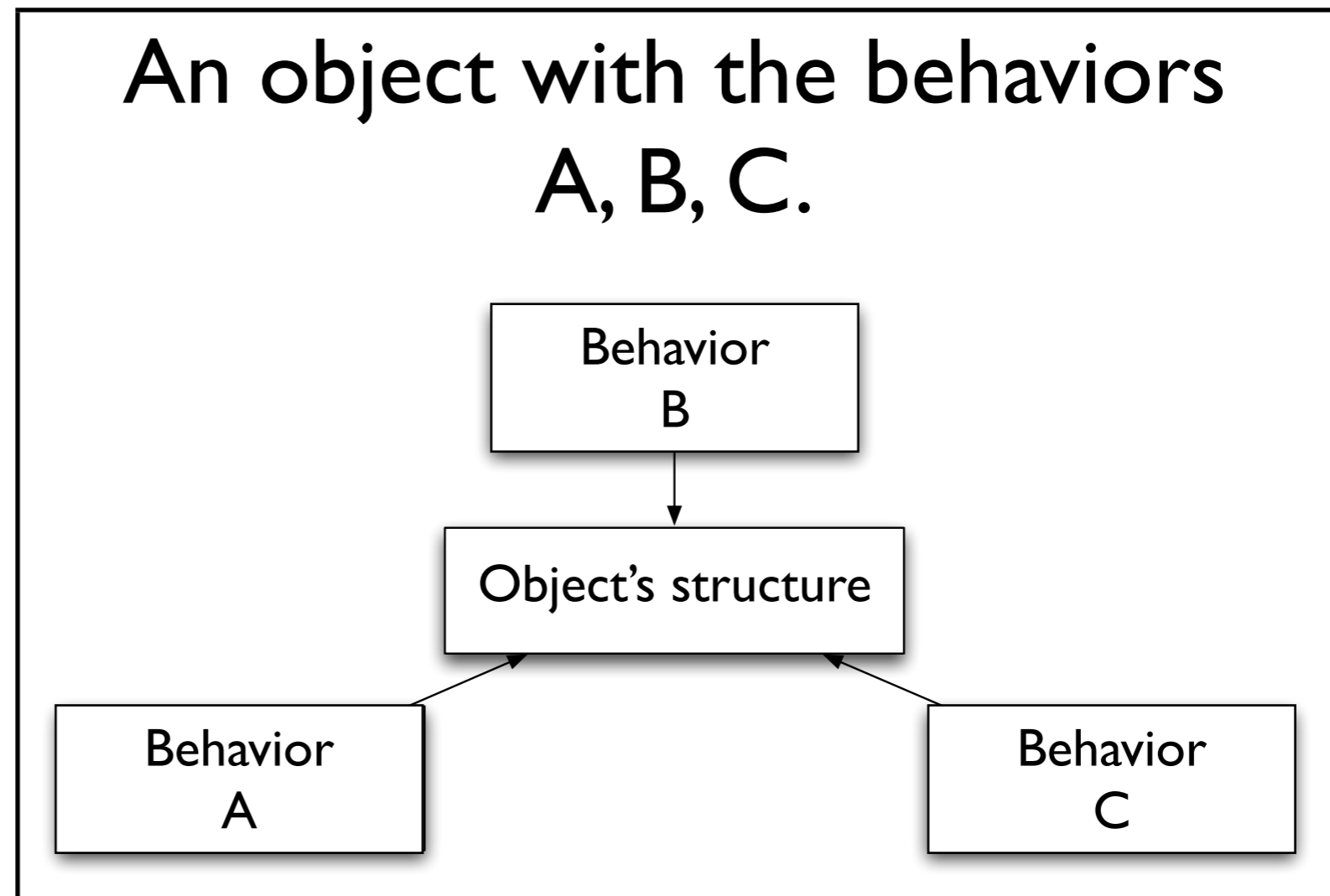
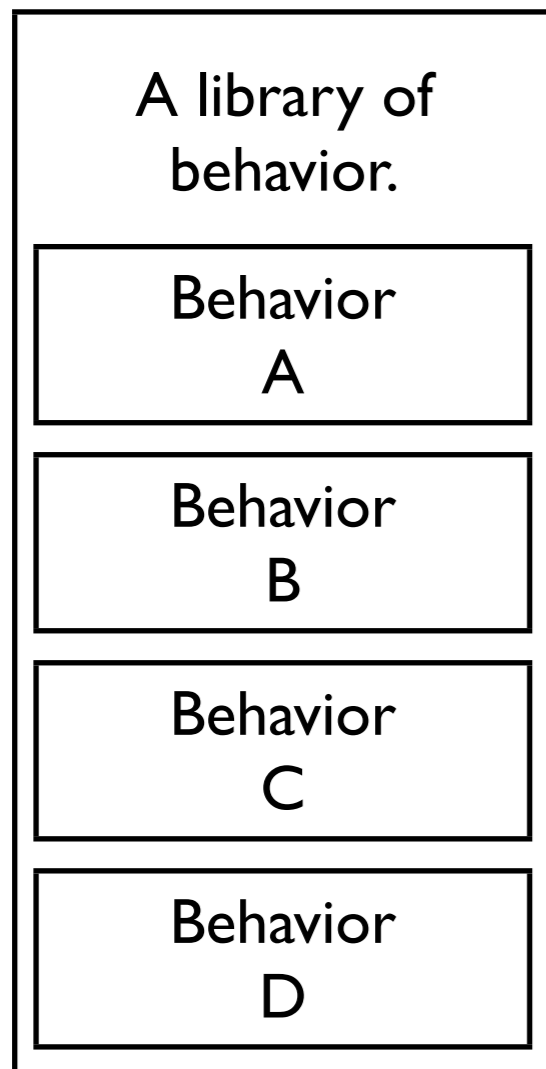


BLOC : A Trait-Based Collections Library

Introduction



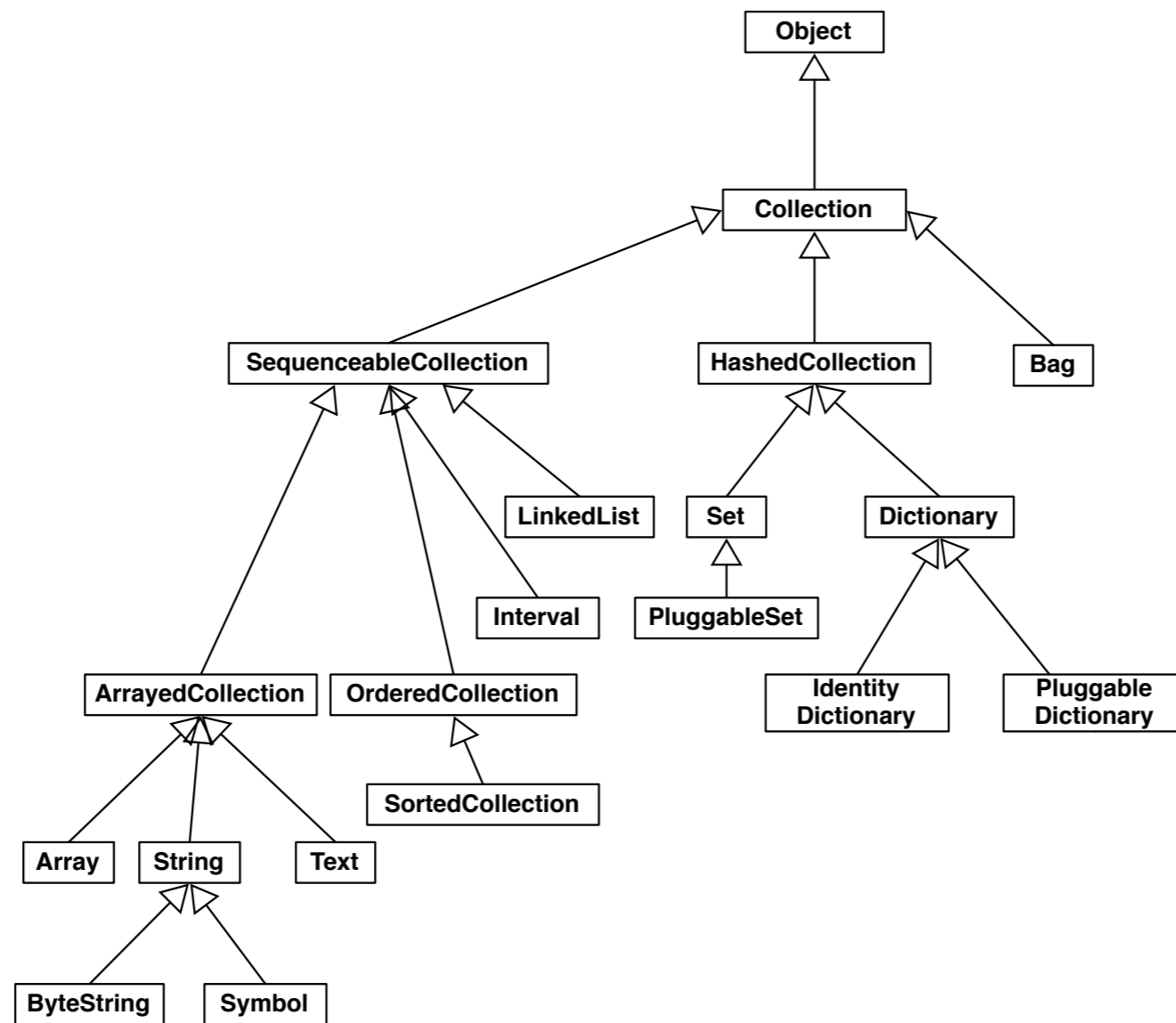
Problematic

- Questions:
 - What is the good granularity for a trait enabling the reuse as well as an easy way to plug it ?
 - How choose between the use of trait and inheritance ?
 - Can traits be used as modular blocks ?

Bloc

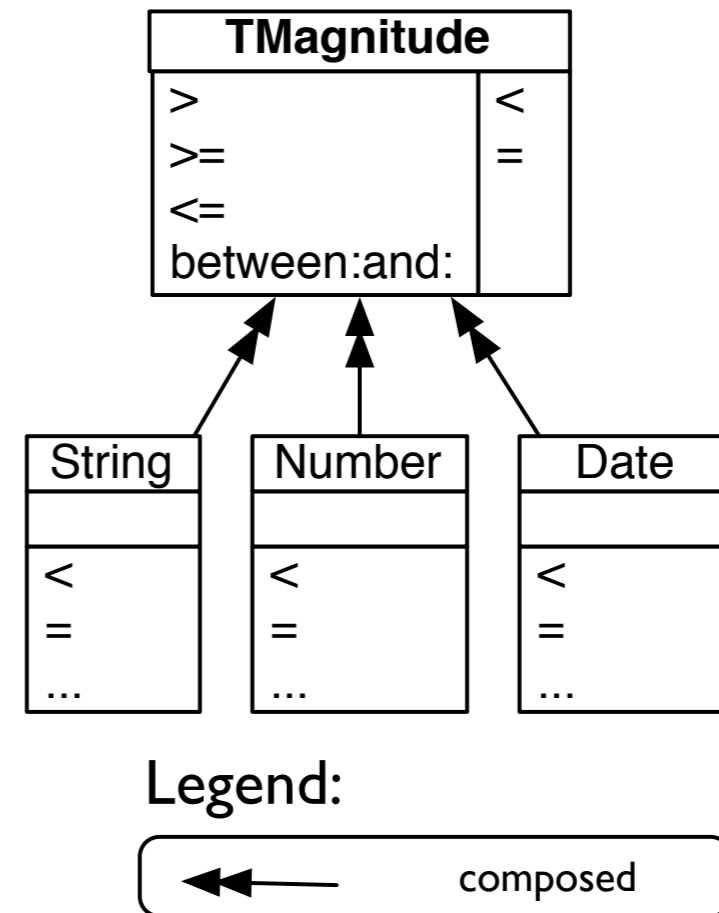
1) Context

Pharo Collections



Traits

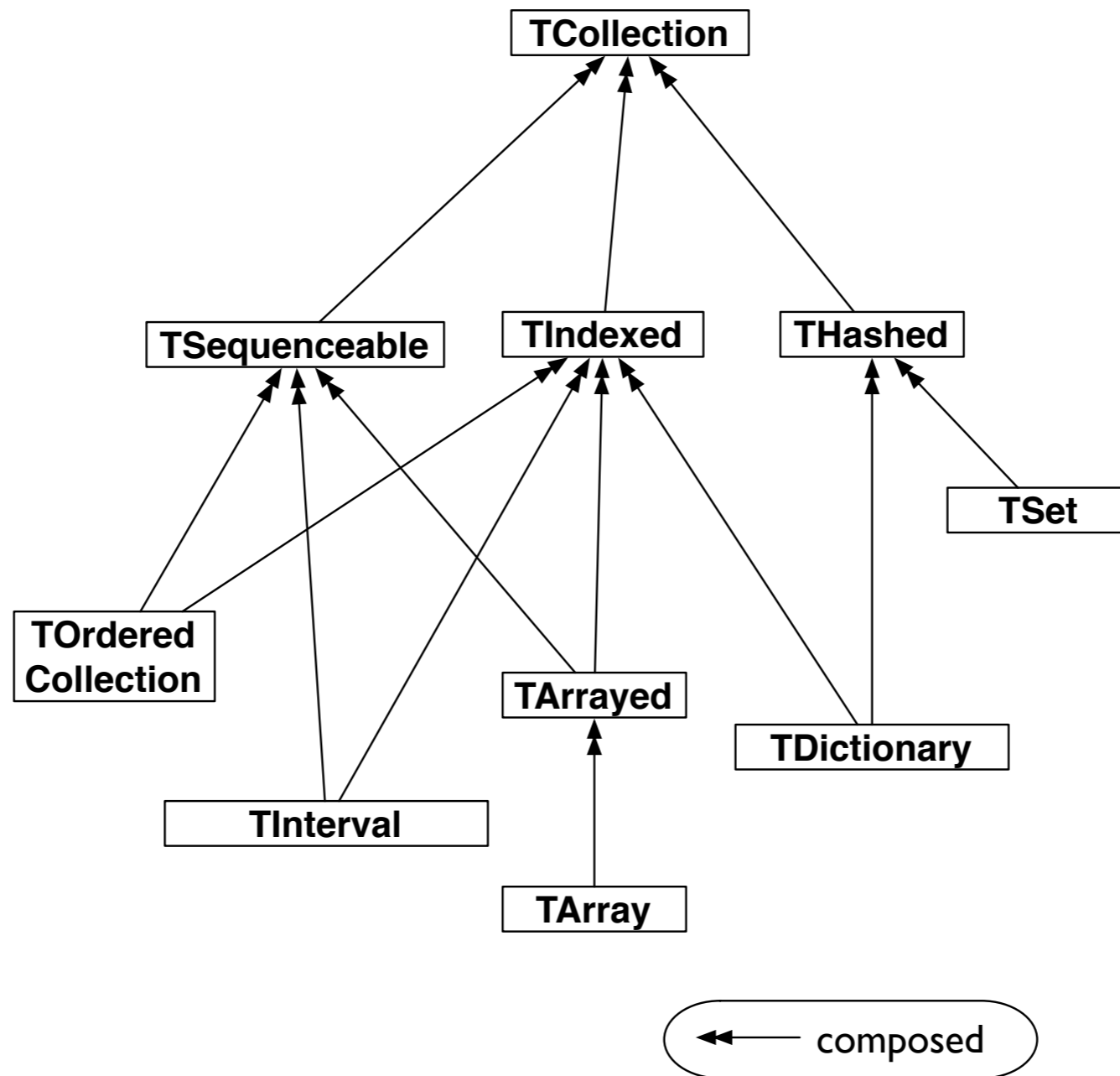
- Define a behavior
- Block reusable methods
- Traits required methods



Bloc

II) Granularity

Specific collection behavior



Global behaviors collections

- TOrdered:

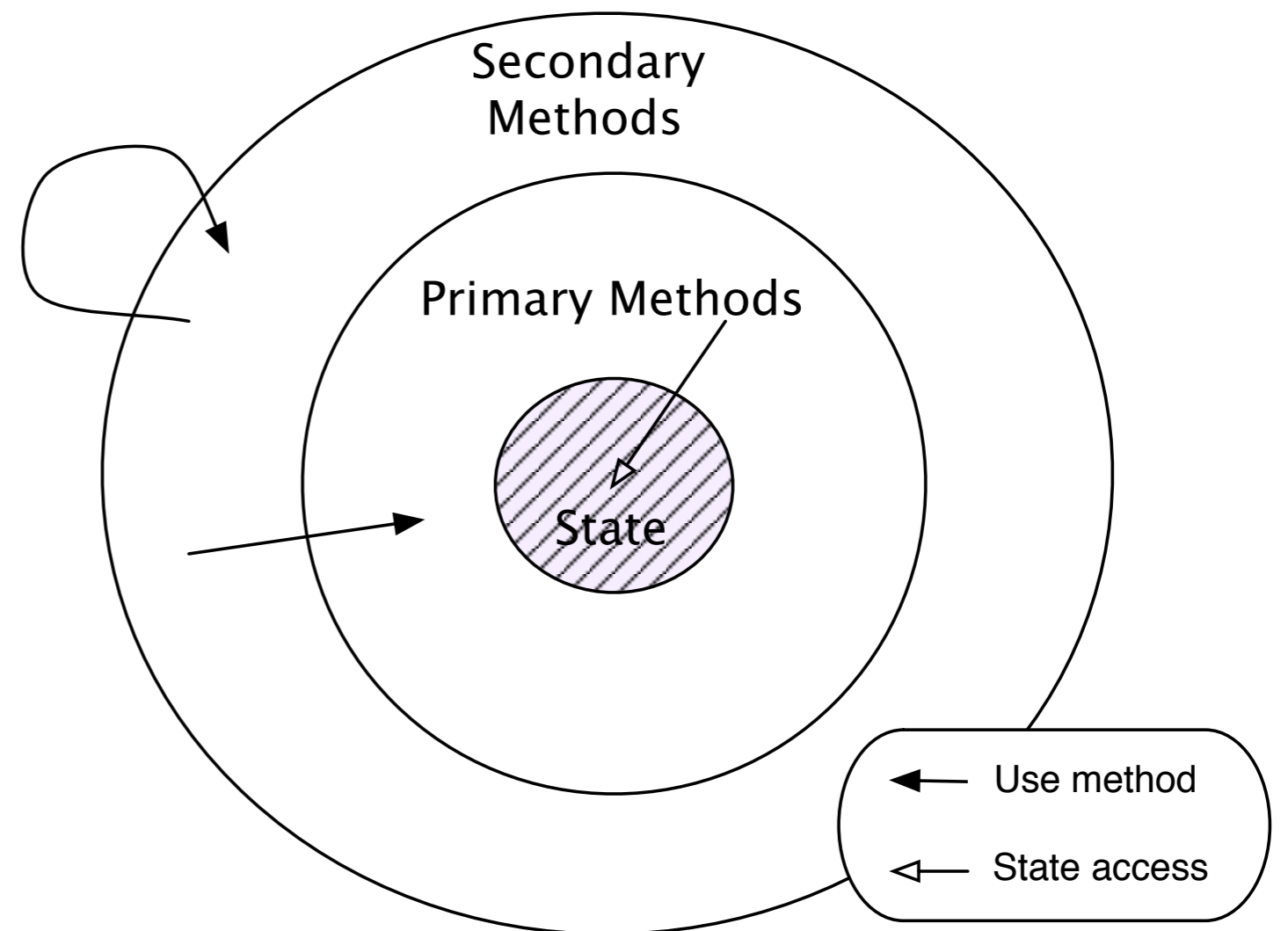
- TOrderedAccessing
- TOrderedAdding
- TOrderedCollection
- TOrderedCopying
- TOrderedCreation
- TOrderedEnumerating
- TOrderedRemoving
- TOrderedTesting
- TOrderedUpdatable

- TArray:

- TArrayAccessing
- TArrayCollection
- TArrayCopying
- TArrayCreation
- TArrayEnumerating
- TArrayRemoving
- TArrayTesting
- TArrayUpdatable

Traits protocols defining: Primary/secondary methods

- Allows separation between traits and the structure.
- Simulated encapsulation.
- Allows to don't waste time because of accessors.

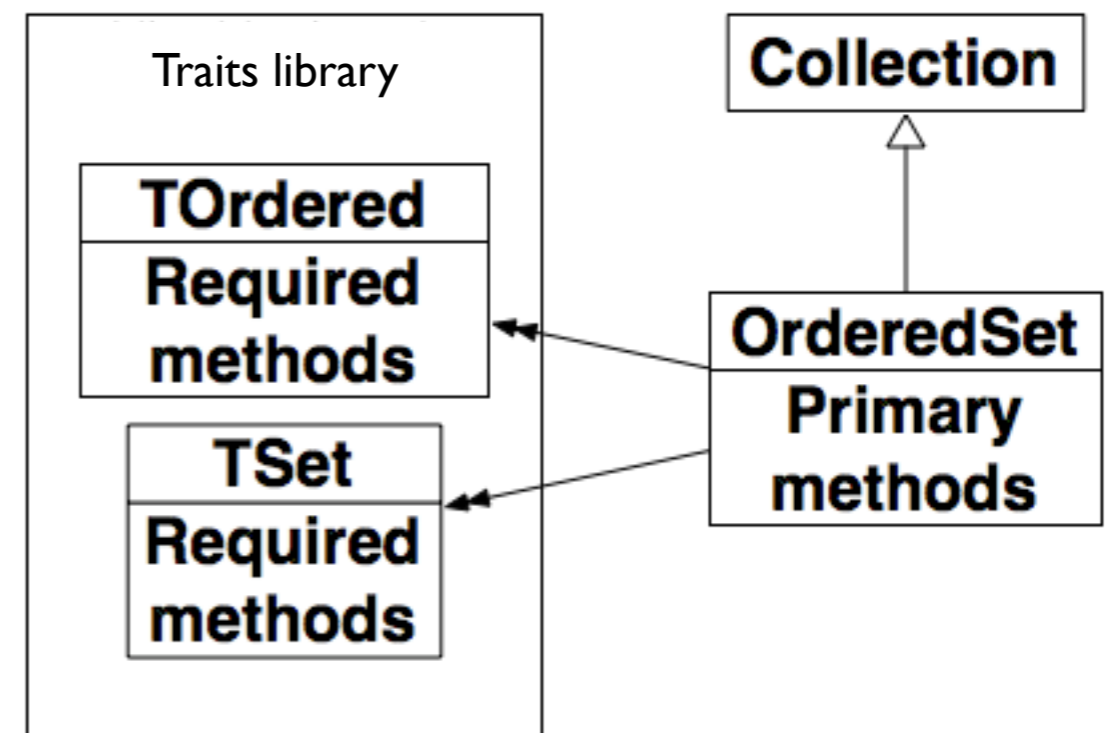


Bloc

III) Case study and discussions

Case study: OrderedSet

- Step 1) Select the behaviors in the traits library.
- Step 2) Define the structure of the new collection.
- Step 3) Implement all the required methods for the new collection.



Discussions

?



- Traits granularity
- Traits remodularity
- Traits vs inheritance

Thanks for your
attention

