object, but we can drop the list at this stage. The same applies to *record of customers* and *customer*.

- *Associations.* In the class diagram in Figure 5.1 *hires* is modelled as an association between Customer and Bike. Whether hires is an object or an association raises some interesting issues and we will postpone that decision until later. The general rule in this situation is that if there is data associated with the relationship (and in this case there is), then we probably want to model it as an object.
- *Outside the scope of the system.* In this category would be anything we have decided to be beyond the system boundary. For example, according to the Problem Definition in Chapter 2 the system will not cover payroll, personnel or general accounting.
- *Really an operation or event.* This criterion is quite confusing and should be treated with care. If a candidate object seems to have no data associated with it, then it might be better modelled as an operation on another class. Quite often, however, operations and events are modelled as objects. For example, hiring a bike might be described as an event but as there is associated data, start date and number of days, we are more likely to want to model it as an object (see below).
- *Represent the whole system.* It is not normally a good idea to have an object that represents the whole system; we want to divide the system into separate objects. *Wheels system* should be rejected for this reason.

That leaves us with the candidate objects:

- bike
- customer
- hire transaction
- specialist bikes.

We can use these objects to form the basis of our class diagram. Remember that class names are always singular – see Figure 5.3.

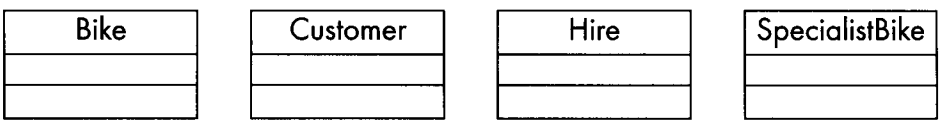

Figure 5.3 Four candidate classes derived from noun analysis