- different amounts of time
- return of a bike
- total amount due
- state of each bike
- extra details about specialist bikes.

We now examine the list of candidate objects and reject those that are unsuitable. Objects should be rejected if they are:

- *Attributes.* Sometimes it is clear that a noun is an attribute of an object rather than an object itself. *Bike number, type, size, make, model, daily charge rate and deposit* are clearly attributes of a bike object rather than objects in their own right. Similarly, hire transaction sounds like a possible object, with *start date* and *estimated duration* as its attributes. *Number of days* also sounds like an attribute as do *different amounts of time, total amount due* and *extra details about specialist bikes. Specialist bike,* however, sounds like an object. If in doubt ask yourself whether the noun you are considering would be likely to have attributes of its own. Would it have behaviour?

- *Redundant.* Sometimes the same concept appears in the text in different guises. Here *past hire transactions* and *hire transaction* are probably the same thing. *Different amounts of time, number of days* and *estimated duration* are probably the same thing – all three refer to the length of a hire (in any case they are attributes, not objects).

- *Too vague.* If we don't know exactly what is meant by a term it is unlikely to make a good class. For this reason we reject *return of a bike* as an object. Return of bike is really an event and features in our use case model. Any data we might need to store about bike returns (e.g. date of return) can probably be bundled with the data about hires.

- *Too tied up with physical inputs and outputs.* This refers to something that exists in the real world but is a product of the system or data input to the system and not an object in its own right. For example, a bill exists in the real world, but is something the system outputs from data it already stores and as such would not be modelled as a class. *Receipt* qualifies for rejection under this heading. *Receipt* is an output; it is also redundant as we either store or can calculate the details we would print on it. *List of bikes* also requires some consideration. If we have an object to represent each bike, we can use these objects to produce a list. *Bike,* therefore, is a good candidate