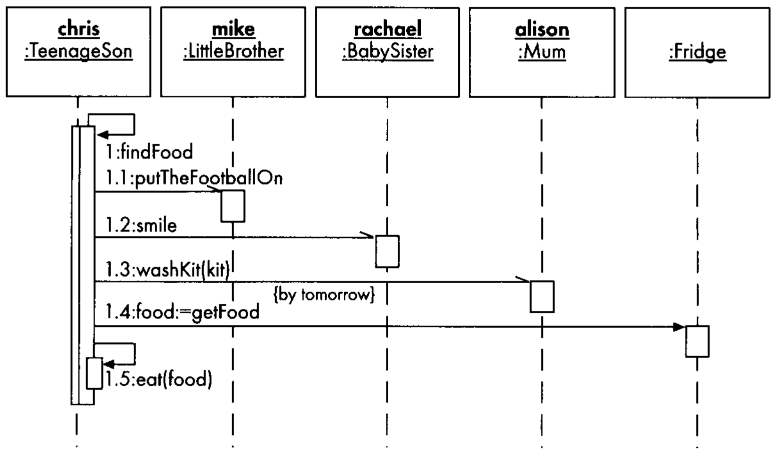

*Figure 10.10* *Asynchronous messages*

describes; it can be added to most modelling elements. Constraints are often attached to classes, e.g. we might add a constraint {hire limit three weeks} to the Hire class if we wanted to impose a time limit on the length of a hire.

The ability to distinguish synchronous from asynchronous messages means that sequence diagrams can be used to model concurrent processing.

We might, more realistically, use asynchronous messages in an automated greenhouse which uses multiprocessing. The greenhouse could have three machines (a fan, heater and sprinkler) controlled by a computer system. We could have a separate process to control each machine. One controlling object could monitor thermostats and send asynchronous messages to objects running the three processes. Because the messages are asynchronous, the three processes could run concurrently.

*Suppressing detail.* Sometimes it is useful to be able to look at sequence diagrams at different levels of detail. The UML has no specific notation to indicate that some detail is hidden in a sequence diagram. However, as we saw in Chapter 6, when an interaction diagram gets too complicated, we can use a package to group cohesive sets of objects. The package is then treated as though it were a single object. A message sent from an object outside the package to any object inside the package is simply sent to the package. The details of inter-object messaging inside the package are suppressed. Another acceptable way of suppressing detail is simply to add a note to a diagram indicating that detail suppressed in this diagram can be found in another diagram.