Table 9.1: Categories of design patterns

Category	*Description*	*Example*
Creational	Deals with creating objects	Singleton – creates a class with only one instance
Structural	Deals with how classes and objects are combined	Façade – provides a higher-level interface for a set of interfaces in a subsystem
Behavioural	Deals with how classes and objects interact	Iterator – accesses all the elements of a collection one by one without revealing the underlying structure

Patterns are not hard-and-fast rules, but descriptions of how particular types of problem may be solved. It is still down to the expertise of the developer to see that a specific pattern will provide the right answer to a specific problem. Patterns are documented using standard templates; there is no general agreement as to the format of the templates, but most of them provide some or all of the following information:

- Pattern name
- Brief description of what the pattern does
- The problem that the pattern addresses
- Constraints on the use of the pattern
- The design of the solution in UML class or sequence diagrams
- The participants in the pattern (the classes or objects involved)
- An example of where the pattern has been used successfully
- Frequency of use of the pattern
- Sample code and details of implementation
- Rationale for the pattern, why it has developed in this way.

We show in Figure 9.20 a simple example of the Façade pattern, including the name of the pattern, what it does, the problem addressed, the participants, an example and a UML diagram.

We do not have room in this book to cover the topic of design patterns in detail. If you want to read more about this subject, the classic book is by Gamma *et al.*, (1995).