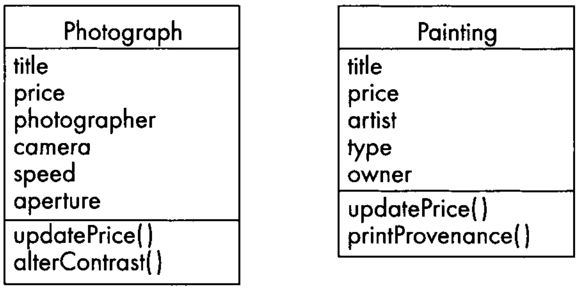

Figure 4.25 *Photograph and Painting have attributes and an operation in common*

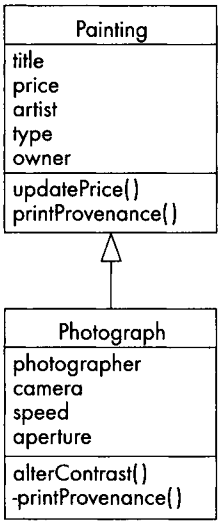

Figure 4.26 *Incorrect use of inheritance*

We could have tried to make Photograph inherit directly from Painting, as in Figure 4.26, but in this case we would have created a subclass whose objects could not be substituted for objects of its superclass.

In Photograph the printProvenance() operation has been effectively disabled by being made private; a client object can send the printProvenance() message to Painting objects but not to Photograph objects. Photograph also inherits the attributes artist, type and owner which it does not use.

A better way to introduce inheritance in this situation is shown in Figure 4.27 (repeated from Figure 4.14). We have introduced a new class, Picture, in which we can place common features. Photograph and Painting retain their distinctive features and share those of Picture.