

registering, with the second class, the rule information and a reference to the attribute of the first class such that an accessor method used to interrogate the attribute of the second class modifies, by way of the registered reference, the attribute of the first class by applying the registered rule information in association with the attribute of the second class.

**5.** A method according to any previous claim, wherein the incorporated functionality within the first class is provided through one or more external classes having functionality for interpreting the position and rule information.

**6.** A method according to claim 1, wherein the element of the second class is an attribute, and wherein the rule information defines the creation of an object from a choice of classes based on the value of the attribute,

the method further comprising:

obtaining the value of the attribute of the second class using the position information; and

creating a new object in accordance with the obtained value and the rule information.

**7.** A method according to claim 6, further comprising creating a hidden object intermediate a first object created from the first class and the new object for regulating access to the first object.

**8.** A method according to claim 6 or 7, wherein the step of obtaining the value further comprises:

registering, with the second class, a reference to the hidden object and the rule information such that an accessor method used to interrogate the attribute of the second class may cause the hidden object to create a further new object, and hence to prevent access to the previous created object, in accordance with the rule information.

**9.** A method according to claim 8, wherein, where a further new object is created, further comprising:

copying attribute values from the previous created object to the further new object on a best-effort basis.

**10.** A method according to claim 6, **7**, **8** or **9**, wherein the step of obtaining the value of the attribute of the second class further comprises:

interpreting the position information to extract the relative location of the second class; and

retrieving the reference to the second class by recursively navigating the hierarchical arrangement of classes in accordance with the interpreted position information.

**11.** A method according to any previous claim, wherein the computer program is written in the Perl language.

**12.** A method according to any previous claim, wherein the class-making module is the Class::MethodMaker module.

**13.** A method according to any previous claim, wherein each class declaration additionally includes a privilege level identifier associated with each attribute thereof, the method further comprising:

comparing the privilege level identifier with a predetermined user associated privilege level and only allowing access to that attribute if the user privilege level is at least as high.

**14.** A method, within an object-oriented computer program, of creating a dependency between a first class and an element of a second class in a hierarchical arrangement of classes created by a class-making module based on declarative definitions of each class,

the method comprising:

defining, within the first class definition,

position information defining the relative position within the hierarchy of the element of the second class, and

rule information defining the nature of the dependency; such that an accessor method created for accessing the attribute of the first class is arranged for:

obtaining the value of the attribute of the second class; and

modifying the attribute of the first class in accordance with the value of the attribute of the second class and the information contained in the second information item.

\* \* \* \* \*