

COMPUTER PROGRAMMING

[0001] The present invention relates to computer programming and, more particular, to computer programming techniques for use in conjunction with a declarative approach to defining classes.

[0002] Many modern day computer-based systems are both complex in nature and are highly configurable enabling such systems to be specifically tailored for the differing requirements of individual users. In many systems, such is the level of complexity surrounding the way in which such systems may be configured it is becoming increasingly difficult for users to comprehend and fully understand the full range of configuration options available to them. Consequently, many computer-based systems may be less than optimally configured which means that important functions or aspects, which producers of such systems often go to great lengths to develop, may be seldom used or not used in the intended manner.

[0003] Computers, computer operating systems, web servers and telecommunications systems are just a few examples of such systems.

[0004] Many of these computer-based systems use configuration files, often in human readable form, which provide configuration parameters for different system elements, and which are used during initialization of the system to configure the appropriate system elements with the appropriate configuration parameters. For complex systems the configuration files may run into many thousands of lines of data describing each of the individual elements to be configured along with the data required for configuring the elements.

[0005] Systems providers often provide standard or default configuration files which may be used to provide a 'typical' system configuration. Such default configuration files are generally not, however, optimized for any particular user. Thus, generally speaking, the user is required to make modifications to the configuration files to ensure that the systems are suitably configured for their particular requirements. For example, to ensure optimum performance, a computer operating system may need to be configured for the particular type of computer hardware used to run the operating system. Although manual editing of a configuration file with, for example, a text editor is generally possible, it typically requires a thorough understanding of the changes being made and the possible implications that a modification to one system element may have on another system element. Since the configuration files are only generally used during the initialization of a system, any errors or inaccuracies which are introduced will usually only be noticed when the system is initialized. Any such problems may cause the system to fail to initialize or may result in an incorrectly configured system. In such cases, the configuration file may require re-editing and the system re-initializing. Such a process can be time-consuming and particularly frustrating for the system administrator.

[0006] To help overcome these problems, it is known to use computer programs to assist in the creation or modification of configuration files. Such computer programs model the configuration behavior of the system to be configured and may provide a degree of checking and validation which aim to help prevent erroneous or incorrect configuration

parameters from being stored in the configuration file. Once the configuration parameters have been entered using the computer program, a configuration file may be created or modified. Webmin (available through <http://webmin.com>) is an example of one such computer assisted configuration file creation tool which is used for creating/modifying configuration files for Unix systems. Other examples include Linuxconf (<http://www.solucorp.qc.ca/linuxconf/>), used for creating and managing Linux configuration files, and Swat, which is a web administration tool for Samba.

[0007] Many systems requiring configuration may be represented using a hierarchical tree-like structure (hereinafter referred to as a configuration tree), as shown in FIG. 1, wherein system elements are represented by 'nodes' of the tree and attributes of the system elements are represented by 'leaves' of the tree. In such trees a node may typically be connected to other nodes, but a leaf may typically only be connected to a single node.

[0008] FIG. 1 shows part of a configuration tree representation 100 of a simple computer system. A computer 102 has a number of system elements: a video card, node 104; a monitor, node 108; and a processor, node 114. The computer node 102 has an attribute of computer type, 103, video card has an attribute of video memory, leaf 106, and pixel clock, leaf 107, the monitor has attributes of screen resolution, leaf 110, and refresh rate, leaf 112, and the processor has attributes of clock frequency, leaf 118, and cache size, leaf 120.

[0009] A computer program used to assist in the creation/modification of a configuration file can model the configuration behavior of each system element, for example, by modeling any particular requirements such as limits on the values an attribute may take, the type of attribute and so on. Although there exist many different programming techniques by which this may be achieved, it will be generally appreciated that the techniques of so-called object oriented programming are particularly suited.

[0010] Object oriented programming broadly involves modeling system elements, or the nodes of the tree, as software objects, and representing the attributes of the system elements, or leaves of the tree, as attributes of the software objects. A software object provides containers for storing data (i.e. the attributes) and methods for accessing, modifying and storing the data. Software objects are generally created from a class definition which acts as a generic blueprint or template for a software object and defines the nature of the data containers and methods. The numerous benefits and advantages of object-oriented programming will be well appreciated by those skilled in the art.

[0011] In order to implement a computer program for modeling the configuration behavior of the various system elements, a class must typically be written for each system element. A class, as is well known in the art, defines, amongst other things, the name of the class, the attributes of each class and methods which enable the attributes to be accessed and modified, known as accessor methods. A certain degree of 'intelligence' may be built into the accessor methods to provide functionality such as range and type checking of attributes.

[0012] In some situations system elements may have dependencies which need to be modeled by the computer