Windows®95 and Windows®98 is provided in, respectively, "Microsoft® Windows®NT Workstation Operating System Deployment Guide," "Microsoft® Windows®95 Resource Kit," and "Microsoft® Windows®98 Resource Kit," each of which has been incorporated by reference.

As detailed above, the present invention automatically generates a GUI whereby a CM-user can quickly and easily select a group of personalization parameters with which an OS can be configured. OS installation module **54** then edits generic answer file **64** with personalization parameters from desktop profiles file **34** to create an updated answer file **66** that includes the selected personalization parameters. In this way, the present invention advantageously allows a system administrator to deploy personalized operating systems to a plurality of workstations **44** on a network without having to manually edit an answer file for each deployment. With the present invention, the system administrator only needs to create one generic answer file **64** for each type of OS which may be deployed and a desktop profile file for each CM-user who will be deploying an OS. Because there are fewer files to manually edit, this can advantageously simplify and make more reliable the process of OS deployment into networked workstations. Further, it can reduce the time necessary to accomplish such deployment.

As noted above, not all OS personalization parameters can be stored in, and then read from, an answer file. Parameters that cannot be stored in an answer file can include end-user login name and end-user password. As such, the present invention includes a second editing module referred to as post-OS installation module **90**, which is downloaded to workstation **36** after setup program **68** has run to install the operating system. Post-OS installation module **90** can edit a plurality of operating system application programming interfaces ("APIs") to include non-preinstallation configurable personalization parameters. An API is a routine that a configuration manager (or any other program) can use to request an operating system to perform lower-level services.

As shown in FIG. **9**, step **158**, which shows the steps completed by post-OS installation module **90**, post-OS installation module **90** detects from OS setup program **68** when OS setup is complete. Then, in step **160**, post-OS installation module **90** reads continue.ini file **72** for the values of comprofiles environment variable **95** and selectedcomp environment variable **97**. In step **162**, post-OS installation module **90** reads the desktop profile file **34** designated in comprofiles environment variable **95** for keys indicating personalization parameters to be configured at the post-OS install stage. Post OS install module **90** is preprogrammed to recognize such keys. For example, for the end-user login name, the post-OS installation module **90** finds the "LoginID" key and retrieves its value. For the enduser password, post OS-installation module **90** finds the "InitialPassword" key and retrieves its value.

Post-OS installation module **90** then places these retrieved values in arguments of APIs. These APIs are essentially lines of code in post-OS installation module **90**. For example, to configure an end-user login name for Windows®NT, Windows®95, and Windows®98 operating systems, the lines of code appear as follows:

```
USER_INFO_2 UserInfo_2;
UserInfo_2.useri2_name=p_pusUserName;
NetUserAdd(p_pusWorkstationName, 1, & UserInfo_2,
  & Error);
```

Where:

p_pusUserName=LoginID and

p_pusWorkstationName=the computer name of the workstation being configured.

The "p_pusUserName" portion of the API is the argument which is replaced with the value of the "LoginID" key in the selected desktop profile. As such, to edit the end-user login name API, post-OS install module **90** reads desktop profile **39** and retrieves the value for the key "LoginID" and places it in the "p_pusUserName" location in the API. Specifically, if the value for "LoginID" in the selected desktop profile **39** was "JSmith", then post-OS installation module would place "JSmith" in the argument of the API call to appear as "UserInfor_2.usri2_name=JSmith." The computer name of the workstation being personalized (which was already configured using the answer file as described above) is also included in the API to associate the end-user login ID with the correct workstation. The Post-OS installation module **90** then executes the API which automatically configures the installed OS to have an end-user login name of JSmith. As noted in the background section, the use of and syntax for Microsoft® operating system APIs is detailed in "Microsoft® Visual C++6.0 Reference Library", Microsoft® Press, 1998.

By automatically editing APIs to include parameters from a desktop profile, the present invention advantageously allows a CM-user to deploy a OS in a workstation without the need to manually enter values for such parameters for each workstation. This can simplify and save time is the OS deployment process and make the process more reliable.

As described above with reference to FIGS. **8** and **9**, it is possible for the present invention to update and use an answer file in configuration of an operating system. However, it is also within the scope of the present invention to use only the post-OS installation method and apparatus described above for automated OS personalization. That is, the post-OS install module **90** can read all the personalization parameters designated by the comprofile environment variable **95** and selectedcomp environment variable **97**, and place these parameters in the appropriate API calls. In this way, the steps described above involving use of an answer file for operating system configuration could advantageously be eliminated.

Having described the invention in terms of a preferred embodiment, it will be recognized by those skilled in the art that various types of general purpose computer hardware may be substituted for the configuration described above to achieve an equivalent result.

What is claimed is:

1. A method for providing personalization parameters to allow an operating system to install itself with the provided personalization parameters on a computer system, the computer system including a memory, a first central processing unit (CPU), and a second CPU interconnected with the first CPU to allow the operating system to be installed from the first CPU to the second CPU, the method comprising:

providing an operating system configuration file in the memory and used by the operating system to configure itself;

providing a personalization parameters file in the memory;

displaying a graphical user interface (GUI) on a display element of the second CPU, the GUI for displaying a list of personalization parameters available in the personalization parameters file and allowing a user to select a plurality of the personalization parameters from the personalization parameters file; and

automatically editing the operating system configuration file to include at least a first portion of the selected plurality of personalization parameters.

2. The method of claim **1** further including the step of accessing the operating system configuration file in the