

(e.g., T1 line, etc.). In further embodiments, the image server **40** may be connected to the Internet using a cable modem or satellite Internet connectivity (as illustrated by transmitter **54**).

In yet another illustrative embodiment, the image server **40** is coupled to a management information system (MIS) **56** via interface **63**. Management Information Systems may be used to support the infrastructure of businesses and organizations wherein such systems are well known to one skilled in the art.

In referring to FIG. **2** a logic flow for creating a disk image of a desired software configuration is illustrated. Block **200** is the start of the logic flow process, which represents receipt of the customer's order. Block **200** generates the Bill of Materials (BOM).

Receipt of the bill of materials is represented by block **204**. The image builder **20** starts with the top record and calculates a configuration identification (ConFig ID) of all the entries. In block **208**, the image builder **20** groups like orders together. Grouping like orders together allows for increased efficiency due to the commonality between orders.

In block **212**, the image builder **20** compares the configuration IDs to the configuration history. If the configuration ID corresponds to a previously configured image, then the image builder **20** looks at whether the image is in a storage device **30**, as illustrated in FIG. **1**. If the image is found in the storage device **30**, then block **224** flags the configuration as ready for delivery and notifies an operator of the computerized network **10** that a desired image is ready. Otherwise, if the image is not found in the storage device **30**, the image is created by the image builder **20** according to block **216** as a fresh build. As part of the fresh build process, block **230** requires the image builder **20** to process the bill of materials to determine the parameters for building an image according to the desired software configuration and ensure that they are compatible with the customer's hardware, software and special requirements. The final result or output from block **230** is an image or "digital picture" of the desired software configuration according to the bill of materials.

FIG. **3** illustrates an architecture of a disk image **280** as created by the image builder **20**. The image builder **20** builds the image **280** in software according to a desired software configuration and delivers that image to a storage device **30**. Sections of the disk image **280** are discussed in the order in which they are presented in FIG. **3**. One skilled in the art will readily realize other embodiments of an image architecture.

Section **300** contains BIOS flash properties. The next two sections, sections **302** and **304**, contain the main operating system and the main applications' program code or instructions. Hardware characteristics of the computing system receiving the disk image **280** are addressed by section **306** dealing with the CMOS settings, section **308** includes the main BIOS instructions, section **310** supports LOC information and section **312** supports desktop parameters for the main operating system.

Section **314** supports information including, but not limited to the following: operating system registers, initialization information and configurations files. Section **316** includes test information. Similar to section **314**, section **318** includes, but is not limited to the following: application system registers, initialization information and configurations files. The last two sections contain an identification of the specific image itself **320**, and the last section contains an identification of the customer **322**. The identification numbers allow for future reference of the created image, which

is helpful for trouble shooting problems in the software configuration and in also adding delta images to the previously delivered image in order to upgrade existing applications.

FIG. **4** illustrates an exemplary embodiment of an identification scheme **380** for a disk image structure. The image identification is a tree structure with a configuration identification number. Also included in the tree structure are underlying identification numbers corresponding to main files and underlying identification numbers corresponding to edited dynamic files (EDF).

More specifically, the identification scheme **380** includes a configuration number **380** identifying what the desired image is built from. It is the foundation from which the image builder **20** works from in creating the desired image. Once the configuration ID **380** is identified, then the main files corresponding to the operating system **402**, e.g., Windows 95, and the desired application **404** are layered on top of the configuration ID **400** basic files. Edited dynamic files corresponding to registry settings **406**, operating system initialization files **408**, application EDF files **410**.

An image build software delivery process has been described. The process creates an image of a customer's order in software before placing the image on a hard drive or other storage means for the customer. Once an image has been created, changes or deltas to the baseline image can easily be made without having to redefine the baseline. Adding a delta image to the baseline image allows the desired image to be achieved. This method provides levels of granularity wherein incremental changes can be made to a system without having to perform major work by redefining the baseline. This allows for easy upgrades and allows technical support to function efficiently.

Although specific embodiments have been illustrated and described herein, it will be appreciated by those skilled in the art that any arrangement which is calculated to achieve the same purpose may be substituted for the specific embodiment of the present invention. Therefore, it is manifestly intended that this invention be limited only by the following claims and equivalents.

What is claimed is:

1. A method of building a custom software configuration comprising:
 - receiving a desired software configuration;
 - surveying a plurality of images of preexisting software configurations and selecting a baseline software configuration corresponding to the desired software configuration;
 - comparing an image of the baseline software configuration with the desired software configuration; and
 - generating an image of a set of changes based on the comparison, the image of a set of changes corresponding to the difference between the baseline software configuration and the desired software configuration.
2. The method of claim **1** further comprising broadcasting the image of the set of changes.
3. The method of claim **1** further comprising incorporating the image of the set of changes with the image of the baseline configuration to generate an image of a custom software configuration.
4. The method of claim **3** further comprising broadcasting the image of the custom software configuration.
5. The method of claim **4** wherein broadcasting comprises broadcasting by a satellite.
6. The method of claim **4** wherein broadcasting comprises broadcasting over the Internet.