FIG. **6** illustrates steps of the routine **600** for decoding a current and selected property modifier. Starting at step **601**, the first action taken is to obtain the size of the argument of the property modifier from the size field. From the discussion above, it will be recalled that the size field in the property modifier is three bits and allows a preferred determination of the size of the argument (spra) of the property modifier. At step **605**, the data type code (sgc) is obtained from the data type code field. This is also three bits, as described in connection with FIG. **4** and Table 4. At step **607**, the special handling code (fSpec) is obtained from the special handling field, if appropriate. Finally, at step **610**, the index (ispmd) is obtained from the index field, which is utilized to access into the property information array in the program module of FIG. **2**. Once these parameters are obtained from the current and selected property modifier, the routine **600** exits, returning these parameters for utilization by other processes, as described in connection with FIG. **5**.

FIG. **7** illustrates the steps associated with the routine **700**, for determining whether the program module can process the current and selected property modifier. Starting at decision **701**, an inquiry is made whether the data type code (three bits) of the current property modifier matches a data type associated with the display command. It will be recalled from prior discussion that the program module that issued the display command (step **501** in FIG. **5**) can be that of various types, such as display character, display paragraph, display section, etc.) It will also be recalled that there is a data-type code (sgc) indicative of the particular type of data presently being formatted upon application of a selected property modifier. In the event that the data type code does not match a data type associated with the display command, the NO branch is taken and the routine **700** exits, passing a NO parameter back to the calling routine. If the data type code does match one the data types associated with the display command, the YES branch is taken to decision **705**.

At decision **705**, the inquiry is made whether the index (nine bits) of the currently examined property modifier is within a range recognized by the calling program module. It will be recalled from prior discussion that each particular version of a computer program typically is constructed to provide a predetermined range of indexes into its property information array. If the index is for some reason outside the range of that recognized by the program module, it is an indication that the calling program module cannot process the property modifier being examined. In this case, the NO branch is taken, and a NO parameter is returned as the routine **700** exits. If at decision **705** the index is within the recognized range, the YES branch is taken, and a YES parameter is returned as the routine **700** exits and returns control to the calling process. In particular, control is passed to process **800** as shown in FIG. **5**.

FIG. **8** illustrates the steps of a routine **800** for applying a selected and current property modifier to the selected data items in the data structure. Starting at decision **801**, the inquiry is made whether the selected and current property modifier requires special handling. This inquiry is made based on the state of the special handling code (fSpec). If so (fSpec=TRUE), the YES branch is taken to step **805**. At step **805**, the current and selected property modifier is processed using exception code of the program module. It will be recalled from prior discussion that the special handling code indicates the presence of special processing code associated with the program module that is employed to carry out the formatting in circumstances for special handling. Upon completion of the exception code execution, the process **800** exits.

Returning to decision **801**, if fSpec is FALSE, the NO branch is taken to step **810**. At step **810**, the data type code (sgc) of the property modifier is employed to determine the particular type of data structure which is to be modified (e.g., a character data type structure (CHP), a paragraph properties structure (PAP), a section property structure (SEP), etc.), and control then passes to step **812**. At step **812**, the nine-bit index of the current and selected property modifier is employed to reference in the property information array associated with the program module (**210** in FIG. **2**). The index provides an offset within the property information array so as to obtain the value of the "b" offset value.

It will be recalled that the appropriate structure for storing the character formatting for the particular type code, such as a CHP, is indexed by the value of the "b" offset in the property information array, and that this particular data type structure (e.g., CHP) is stored in a temporary workspace in RAM. Control then passes to step **815**.

In step **815**, the argument size of the property modifier (three bits) is employed to determine the length of the data in the argument portion of the property modifier presently being examined. Then, in step **817**, the argument data of the current and selected property modifier is employed and inserted into the particular data type structure (e.g., CHP, PAP, etc.), starting at the "b" offset value obtained at step **812**. The result is to replace or write over the values in the data type structure, thereby creating a modified data type structure that contains the formatting information that is to be applied to the selected data elements.

The end result is the modification of the data type structure stored in temporary workspace to contain appropriate formatting information so that the selected data elements upon display, printing, etc. have applied thereto appropriate formatting properties to achieve the desired formatting result. Upon completion of step **817**, the process **800** is complete and the particular data type structure associated with the property modifier being examined is complete. Process **800** then returns, and further steps would follow execution of process **800** are carried out (FIG. **5**).

Advantageously, the present invention provides a formatting method and system that allows a particular earlier version of a computer application program such as a word processor to handle a document created with a later version of the word processing computer application program that may contain property modifiers added after creation of the earlier version. The invention allows the earlier version handling such a document to skip to property modifiers that it is not programmed to handle. Furthermore, the earlier version of the program is still able to preserve the property modifiers introduced into the document by the later version by merely skipping them, while still applying any property modifiers that it can handle. Such skipped property modifiers may be preserved upon a file save operation by leaving the unknown property modifiers intact and writing them out to permanent storage.

Further, the present invention can be applied to other uses where representing a change in a data structure is more convenient than representing the full set of values for that structure. For example, as a user edits a document and changes property values, rather than making changes to each and every property structure changed, one can merely record the change as one of more improved property modifier/value groupings, and apply those changes only as needed for displaying the document.

These improved property modifiers can also be used to represent the differences between a "style" and its base style