

represent an, object, a property, a characteristic of any object in the self-describing file, or any component of the self-describing file (such as a section, a template, etc.) or a characteristic of the self-describing file itself. In embodiments, the second portion of data may be related to the first portion of data. For example, the value of the second portion of data may depend upon the value of the first portion of data. Furthermore, the second portion of data may be natively supported by a first application that creates the self-describing file, but it may not be natively supported by other applications that may subsequently access the self-describing file. As such, a second application that later accesses and modifies the file, for example, by writing a new value for the first portion of data, may not correctly update the second portion. In such circumstances, when the application that created the file again accesses the file, it may encounter an error with respect to the second portion of data. Exemplary errors include errors with respect to displaying the second portion of data to the user, errors accessing the second portion of data, errors writing a value related to the second portion of data, errors in the value of the second portion of data, or any other type of error.

To prevent error situations, flow proceeds to operation **208** where the application provides at least one extension that provides information about the portions of the file that a less featured application may not natively support. For example, the one or more extension may define a relationship between the first portion of data and the second portion of data, may define the second portion of data, or may provide any other type of information regarding the file that a different version of an application (or different application altogether) may not natively support. In embodiments, defining a relationship may entail providing a formula to calculate the second portion of data based on the first, data defining the second portion of data (e.g., the structure of the data, information about the fields of the data, etc.), or any other information about the second portion of data that an application accessing the self-describing file may not natively understand and/or support. In embodiments, the at least one extension may be written to the extension section that may be part of or associated with the self-describing file, such as extension section **102** of FIG. **1**. In alternate embodiments, the at least one extension may be provided in a separate file that is related to the self-describing file, as described with respect to FIG. **1**. The at least one extension provided at operation **208** may be of the type of extension elements and/or the child described. One of skill in the art will appreciate that an extension may take any form so long as it describes a relationship, an object, a property, a formula, or any other component of a self-describing file in a manner such that a second application that subsequently accesses the self-describing file is capable of calculating a value for the relationship, an object, a property, a formula, or any other component even if the second application does not natively support such calculations.

In embodiments, the application that creates the self-describing file may generate the at least one extension that provides information about at least a portion of the content of the self-describing file. For example, the application may generate a function or formula that describes the relationship between the first portion of data and second portion of data. In embodiments, the application creating the self-describing file may also include information in the extension section related to properties, object, or other elements of the file that a less featured application may not natively support. Such information may be provided in the at least one extension. In an alternate embodiment, the application creating the self-describing file may receive information about the extension from another source. For example, the application creating

the self-describing file may receive a formula or function describing the relationship from a user, from another application, or from a file stored locally or on a network. Regardless of whether the application generates the extension itself or receives the extension from another source, the extension may be included in the self-describing file at operation **208**.

After providing the at least one extension at operation **208**, flow continues to operation **210** where, in embodiments, the application creating the self-describing file writes any additional data (e.g., additional object, properties, extensions, etc.) to the file and saves the file. In embodiments, the file may be saved in computer storage media, such as the computer storage media described in FIG. **5**. Storing the self-describing file allows the applications to access the self-describing file in the future.

Although the method **200** is described as providing two portions of data to the self-describing file, one of skill in the art will appreciate that any number of properties may be included in the self-describing file. Additionally, in embodiments, any number of extensions may be written to the self-describing file defining different types of relationships, properties, objects, etc. For example, a third property may be written to the self-describing file. The value of the third property may be dependent upon the value of the second property. In embodiments, an extension may be written to the self-describing file that defines the calculation used to derive a value for the third property based upon the second property. Furthermore, if the second property in the described embodiment is related to the first property, the third property is also related to the first property based upon the defined calculation. In such embodiments, the extensions defining the relationship (e.g., defined calculations) between the first and second properties and the second and third properties will capture such relationships between the properties. In embodiments, the relationships between an unsupported second property, defined in the one or more extensions, and a supported third property may be defined in the data associated with the third property. The definition of the second property in the extension section may allow the third property to recalculate and preserve values associated with the third property.

In further embodiments, complex relationships may be defined by one or more extensions. For example, a fourth property and a fifth property may be added to the self-describing file during the method **200**. The value of the fifth property may be related to both the fourth and first properties. In embodiments, one or more extensions may be provided that capture the multiple dependency of the fifth property. One of skill in the art will appreciate that embodiments of self-describing file disclosed herein may be capable of capturing any type of relationship or dependency, regardless of the complexity, by including relationship information in the self-describing file (e.g., by including one or more extensions).

In another embodiment, rather than defining a relationship between two portions of data the extension may provide information about a single portion of data that a less featured application may not natively support. For example, the extension may contain information defining an object, property, or other component of the self-describing file that may not be natively supported by some versions of an application. In embodiments, the information defining the object, property, or other component of the self-describing file may provide the less featured application with enough detail to calculate a value for the unsupported portion of the file. As such, extension data may be used for providing information about portions of a file other than information related to relationships.

The self-describing file created and saved during the method **200** may be of any type. For example, the self-de-