(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4072293号
(P4072293)

(45)発行日　平成20年4月9日(2008.4.9)　(24)登録日　平成20年1月25日(2008.1.25)

(51)Int.Cl.		F I		
G06F 12/00	(2006.01)	G06F	12/00	520E
G06F 17/30	(2006.01)	G06F	17/30	230Z
H04N 5/92	(2006.01)	H04N	5/92	H

請求項の数 28　(全 15 頁)

(21)出願番号　特願平11-189636
(22)出願日　平成11年7月2日(1999.7.2)
(65)公開番号　特開2001-22615(P2001-22615A)
(43)公開日　平成13年1月26日(2001.1.26)
審査請求日　平成18年6月15日(2006.6.15)

(73)特許権者　000001007
キヤノン株式会社
東京都大田区下丸子３丁目３０番２号
(74)代理人　100076428
弁理士　大塚　康徳
(74)代理人　100112508
弁理士　高柳　司郎
(74)代理人　100115071
弁理士　大塚　康弘
(74)代理人　100116894
弁理士　木村　秀二
(74)代理人　100093908
弁理士　松本　研一
(74)代理人　100101306
弁理士　丸山　幸雄

最終頁に続く

(54)【発明の名称】データ処理方法及び装置及び記憶媒体

(57)【特許請求の範囲】

【請求項１】

バイナリデータにメタデータを登録するデータ処理方法であって、

メタデータの付与対象のバイナリデータを読み込む第１読込工程と、

前記バイナリデータに付与すべきメタデータを読み込む第２読込工程と、

前記第１読込工程で読み込まれたバイナリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する第１接続工程と、

前記第１接続工程で接続されたメタデータの後に、前記バイナリデータに対応する終端コードを接続する第２接続工程と、

前記第１及び第２接続工程によって得られたデータの全体を一つのファイルとして出力する出力工程とを備えることを特徴とするデータ処理方法。

【請求項２】

前記バイナリデータが終端コードを有するか否かを判定する判定工程をさらに備え、

前記第２接続工程は、前記判定工程で終端コードを有すると判定された場合に、前記第１接続工程で接続されたメタデータの後に、前記バイナリデータに対応する終端コードを接続することを特徴とする請求項１に記載のデータ処理方法。

【請求項３】

前記第２接続工程は、予め決められた終端コードを前記第１接続工程で接続されたメタデータの後に接続することを特徴とする請求項１に記載のデータ処理方法。

【請求項４】

前記第２接続工程は、予め決められた複数種類の終端コードを前記第１接続工程で接続されたメタデータの後に接続することを特徴とする請求項１に記載のデータ処理方法。
【請求項５】
前記第２読込工程で読み込まれたメタデータが、予め定められたデータ記述言語における適正な形式で記述されているか否かを判定する判定工程を更に備え、
前記第１接続工程は、前記判定工程で適正な形式で記述されていると判定された場合に、前記メタデータを前記バイナリデータの後に接続することを特徴とする請求項１に記載のデータ処理方法。
【請求項６】
前記判定工程は、前記メタデータが前記予め定められたデータ記述言語としての正当性を満足するか否かを含めて判定することを特徴とする請求項５に記載のデータ処理方法。
【請求項７】
メタデータが登録されたバイナリデータにおいてメタデータを判別する方法であって、
データを読み込む読込工程と、
前記読込工程で読み込まれたデータを末尾より検査し、終端コードに続いて予め定められたデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程とを備えることを特徴とするデータ処理方法。
【請求項８】
前記判別工程においてメタデータが判別された場合、判別されたメタデータを抽出して出力する出力工程を更に備えることを特徴とする請求項７に記載のデータ処理方法。
【請求項９】
前記出力工程は、前記抽出されたメタデータに基づく表示を行うことを特徴とする請求項８に記載のデータ処理方法。
【請求項１０】
前記出力工程は、前記抽出されたメタデータを、前記予め定められたデータ記述言語を処理するためのツールに提供することを特徴とする請求項８に記載のデータ処理方法。
【請求項１１】
前記バイナリデータは画像データであることを特徴とする請求項１乃至１０のいずれかに記載のデータ処理方法。
【請求項１２】
前記バイナリデータは音声データであることを特徴とする請求項１乃至１０のいずれかに記載のデータ処理方法。
【請求項１３】
前記バイナリデータは動画像データであることを特徴とする請求項１乃至１０のいずれかに記載のデータ処理方法。
【請求項１４】
前記予め定められたデータ記述言語がＸＭＬであることを特徴とする請求項１乃至１０のいずれかに記載のデータ処理方法。
【請求項１５】
前記予め定められたデータ記述言語がＳＧＭＬであることを特徴とする請求項１乃至１０のいずれかに記載のデータ処理方法。
【請求項１６】
前記予め定められたデータ記述言語がＨＴＭＬであることを特徴とする請求項１乃至１０のいずれかに記載のデータ処理方法。
【請求項１７】
バイナリデータにメタデータを登録するデータ処理装置であって、
メタデータの付与対象のバイナリデータを読み込む第１読込手段と、
前記バイナリデータに付与すべきメタデータを読み込む第２読込手段と、
前記第１読込手段で読み込まれたバイナリデータの後に、前記第２読込手段で読み込ま

れたメタデータを接続する第１接続手段と、
前記第１接続手段で接続されたメタデータの後に、前記バイナリデータに対応する終端コードを接続する第２接続手段と、
前記第１及び第２接続手段によって得られたデータの全体を一つのファイルとして出力する出力手段とを備えることを特徴とするデータ処理装置。
【請求項１８】
前記バイナリデータが終端コードを有するか否かを判定する判定手段をさらに備え、
前記第２接続手段は、前記判定手段で終端コードを有すると判定された場合に、前記第１接続手段で接続されたメタデータの後に、前記バイナリデータに対応する終端コードを接続することを特徴とする請求項１７に記載のデータ処理装置。
【請求項１９】
前記第２接続手段は、予め決められた終端コードを前記第１接続手段で接続されたメタデータの後に接続することを特徴とする請求項１７に記載のデータ処理装置。
【請求項２０】
前記第２接続手段は、予め決められた複数種類の終端コードを前記第１接続手段で接続されたメタデータの後に接続することを特徴とする請求項１７に記載のデータ処理装置。
【請求項２１】
前記第２読込手段で読み込まれたメタデータが、予め定められたデータ記述言語における適正な形式で記述されているか否かを判定する判定手段を更に備え、
前記第１接続手段は、前記判定手段で適正な形式で記述されていると判定された場合に、前記メタデータを前記バイナリデータの後に接続することを特徴とする請求項１７に記載のデータ処理装置。
【請求項２２】
前記判定手段は、前記メタデータが前記予め定められたデータ記述言語としての正当性を満足するか否かを含めて判定することを特徴とする請求項２１に記載のデータ処理装置。
【請求項２３】
メタデータが登録されたバイナリデータにおいてメタデータを判別する方法であって、
データを読み込む読込手段と、
前記読込手段で読み込まれたデータを末尾より検査し、終端コードに続いて予め定められたデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別手段とを備えることを特徴とするデータ処理装置
【請求項２４】
前記判別手段においてメタデータが判別された場合、判別されたメタデータを抽出して出力する出力手段を更に備えることを特徴とする請求項２３に記載のデータ処理装置。
【請求項２５】
前記出力手段は、前記抽出されたメタデータに基づく表示を行うことを特徴とする請求項２４に記載のデータ処理装置。
【請求項２６】
前記出力手段は、前記抽出されたメタデータを、前記予め定められたデータ記述言語を処理するためのツールに提供することを特徴とする請求項２３に記載のデータ処理装置。
【請求項２７】
バイナリデータにメタデータを登録するデータ処理をコンピュータに実行させるための制御プログラムを格納する記憶媒体であって、前記データ処理が、
メタデータの付与対象のバイナリデータを読み込む第１読込工程と、
前記バイナリデータに付与すべきメタデータを読み込む第２読込工程と、
前記第１読込工程で読み込まれたバイナリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する第１接続工程と、
前記第１接続工程で接続されたメタデータの後に、前記バイナリデータに対応する終端

コードを接続する第２接続工程と、
前記第１及び第２接続工程によって得られたデータの全体を一つのファイルとして出力する出力工程とを備えることを特徴とする記憶媒体。
【請求項２８】
メタデータが登録されたバイナリデータからメタデータを判別する処理をコンピュータに実行させるための制御プログラムを格納する記憶媒体であって、前記データ処理が、
データを読み込む読込工程と、
前記読込工程で読み込まれたデータを末尾より検査し、終端コードに続いて予め定められたデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程とを備えることを特徴とする記憶媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明はバイナリデータとメタデータを扱うデータ処理方法及び装置及び記憶媒体に関する。
【０００２】
【従来技術の説明】
メタデータ（meta-data）とは、「データに関する付属データ」であり、画像データや音声データ等のバイナリデータを説明するデータとして用いられている。しかし、バイナリデータとこれに対応するメタデータが別々のファイルで存在した場合、ファイルの移動やコピーの際に、ユーザはバイナリデータとメタデータとを同時に管理しなければならず、非常にわずらわしいことになる。
【０００３】
そこで一般に、バイナリデータとメタデータの管理を容易にするために、バイナリデータとメタデータを記述する様々な方法が提案されてきた。この種の従来技術は、新しいバイナリフォーマットを規定する方法と、データベースで管理する方法の２つに分けることができる。
【０００４】
まず、新しいバイナリフォーマットを規定する方法の一例をあげると、画像フォーマットではTiff、Exif、Flashpixなどがある。図９は、バイナリデータにメタデータを埋め込んだフォーマットの概観を示す図である。バイナリデータとしては、例えば画像データが挙げられる。図９に示されるように、画像のヘッダ部分にメタデータを記述する枠組みを設け、そこにユーザがメタデータを記述するというのが一般的な方法である。このようにメタデータを記述することにより、データの検索・分類が容易になる。また、バイナリデータ内にメタデータを含むようになるので、１つのファイルで管理でき、ファイルの管理は比較的容易になる。
【０００５】
次に、バイナリデータとメタデータをデータベースで管理する方法を説明する。図１０はバイナリデータとメタデータをデータベースで管理する方法を概念的に示した図である。図１０に示されるような、別々のファイルで存在するバイナリデータとメタデータをデータベース等を用いて管理するという方法も広く行われているものである。この場合は既存のバイナリデータが、既存のアプリケーションでそのまま使えるという利点がある。
【０００６】
【発明が解決しようとする課題】
しかしながら、上述したようなメタデータを記述する新フォーマットを規定する方法とデータベースを用いてメタデータを管理する方法のそれぞれに問題がある。
【０００７】
まず、メタデータを記述する新フォーマットを規定した場合には、既存のバイナリデータを当該新フォーマットに変換し、なおかつその新フォーマット内にメタデータを記述しな

けれなばならない。更に、その新フォーマット内のメタデータを用いて検索するためには、当該新フォーマット対応のアプリケーションが必要となる。すなわち、メタデータを記述したり利用したりするために、非常に多くのステップと専用の環境が必要になるという問題がある。また、このような新フォーマットのバイナリデータを処理する（例えば画像データであれば画像の出力）ためには、当該フォーマットに対応したアプリケーションが必要であり、既存のアプリケーションでは対応できなくなる。

【０００８】

そのうえ、メタデータの記述方法も新フォーマットにおいて独自に決められたものであり、新フォーマット内のメタデータを利用するアプリケーションを作成するためには、新規にメタデータの検索ルーチンをつくらなければならないという問題もある。さらに、新しい枠組みのメタデータを記述するにはフォーマットの規定を変更しなければならないという問題点もあった。

【０００９】

一方、データベースを用いてバイナリデータとメタデータを同時に管理する場合、データベースソフトが無ければメタデータの登録も利用もできないという問題があった。また、登録したメタデータを表示するためにも専用のソフトウェアが必要である。更に、バイナリデータをデータベース外に持っていくと、メタデータは付加されず、メタデータのないバイナリデータとなってしまうという問題点もあった。

【００１０】

さらに、メタデータを管理するための情報の中に、バイナリデータのフォーマット上での予約語あるいは、特別な意味をなすコードが入っていた場合、既存のアプリケーションが誤動作する可能性があった。

【００１１】

本発明はメタデータの記述・検索に関する上記の問題点に鑑みてなされたものであり、既存のアプリケーションに影響を与えずに、バイナリデータにメタデータを登録可能とすることを目的とする。

【００１２】

また、本発明の他の目的は、メタデータが登録されたバイナリデータを、既存のアプリケーションで処理することが可能な形態で提供可能とすることにある。

【００１３】

また、本発明の他の目的は、メタデータの記述に一般的なデータ記述言語を用いることにより、データ記述言語用の既存のツールを利用することを可能とし、対応アプリケーションの開発を容易にすることにある。

【００１４】

さらに、本発明の他の目的は、メタデータが記述されたバイナリデータからメタデータを抽出し、例えば検索、参照、変更等の処理に供することを可能とすることにある。

【００１５】

さらに、本発明の他の目的は、メタデータが記述されたバイナリデータを容易に判別する方法を提供することも目的とする。

【００１６】

【課題を解決するための手段】

上記の目的を達成するために、本発明の一態様によるデータ処理方法はたとえば以下の工程を備える。すなわち、

バイナリデータにメタデータを登録するデータ処理方法であって、

メタデータの付与対象のバイナリデータを読み込む第１読込工程と、

前記バイナリデータに付与すべきメタデータを読み込む第２読込工程と、

前記第１読込工程で読み込まれたバイナリデータの後に、前記第２読込工程で読み込まれたメタデータを接続する第１接続工程と、

前記第１接続工程で接続されたメタデータの後に、前記バイナリデータに対応する終端コードを接続する第２接続工程と、

前記第１及び第２接続工程によって得られたデータの全体を一つのファイルとして出力する出力工程とを備える。

【００１７】

また、上記の目的を達成するための本発明の他の態様によるデータ処理方法はたとえば以下の工程を備える。すなわち、

メタデータが登録されたバイナリデータにおいてメタデータを判別する方法であって、

データを読み込む読込工程と、

前記読込工程で読み込まれたデータを末尾より検査し、終端コードに続いて予め定められたデータ記述言語における適正な形式で記述されたデータが存在するか否かを判定することにより、該データに含まれるメタデータを判別する判別工程とを備える。

【００１８】

また、本発明の他の態様によれば、上記のデータ処理方法を実現するデータ処理装置が提供される。さらに本発明の他の態様によれば、上記のデータ処理方法をコンピュータに実現させるための制御プログラムを格納した記憶媒体が提供される。

【００１９】

【発明の実施の形態】

以下、添付の図面を用いて本発明の実施形態を説明する。

【００２０】

＜第１の実施形態＞

本実施形態では、バイナリデータの後ろに、所定のデータ記述言語（本実施形態ではＸＭＬを用いる）で記述されたメタデータを接続することによりメタデータの付加されたバイナリデータを生成する。このとき、接続されたメタデータの中に、バイナリデータのフォーマット上での予約語、あるいは、特別な意味をなすコードが入っていた場合、既存のアプリケーションが誤動作してしまう場合がある。以下では、まずこの点を説明し、その後で本実施形態のデータ処理装置について説明することにする。

【００２１】

たとえば、バイナリデータがＪＰＥＧ方式で符号化されている場合、ファイルの最後にＥＯＩ（End Of Image）コードが存在するか否かを判断する場合がある。図２は、ＥＯＩマーカーコード検索の手順を示すフローチャートである。

【００２２】

図２において、まずステップＳ２００で、変数ＰＯＳにＪＰＥＧ符号化データのファイルのバイト数をセットする。ステップＳ２０１で変数ＰＯＳから２を引く。ステップＳ２０２で、変数ＰＯＳの位置にファイル読み出し位置をシークする。次にステップＳ２０３で２バイトリードする。

【００２３】

ステップＳ２０４にて、読み込んだ２バイトのデータがＥＯＩコードか否かを判断し、ＥＯＩコードの場合はステップＳ２０５に移動し、ＥＯＩコードがあったと判断する。一方ステップＳ２０４で、２バイトのデータがＥＯＩコードではない場合は、ステップＳ２０６に移動する。ステップＳ２０６では、ＪＰＥＧ符号化データ内にある可能性があるマーカーコードか否かを判断する。このステップでマーカコードであると判断すると、ステップＳ２０７に移動し、このＪＰＥＧファイにルの中には、ＥＯＩコードがなかったと判断する。一方ステップＳ２０６にて、マーカコードではないと判断した場合は、ステップＳ２０８に移る。ステップＳ２０８では、現在の変数ＰＯＳの値から１を引く。次にステップＳ２０９で、変数ＰＯＳが０か否かを判断する。０の場合は、当該ファイルの先頭まで読み込んだことになるので、処理をステップＳ２０７に移動し、ＥＯＩマーカコードはなかったと判断する。一方ステップＳ２０９で変数ＰＯＳが０ではない場合は、ステップＳ２０２に戻り、上記の処理を繰り返す。

【００２４】

以上の処理により、これから処理を行おうとしているＪＰＥＧファイルが、完全なＪＰＥＧファイル（例えば途中でコードが終わっていないかどうか等）か否かを判断することが

できる。そして、例えば、ＥＯＩコードが存在しないＪＰＥＧファイルに対しては、デコード処理を行わないといった処理を行うことができるようになる。

【００２５】

以上の様な処理が実行された場合、ＪＰＥＧ符号化データの後ろにメタデータが接合されていると、図２で説明したＥＯＩの存否判断において誤動作する可能性がある。また、ＥＯＩコードをファイルの最後から探す場合でも、接合したメタデータのデータ量分コードの判断を繰り返す必要が生じてしまい、処理時間がかかるという問題点もあった。

【００２６】

本実施形態ではこれらの不具合を解決するデータ処理装置を説明する。

【００２７】

図１は第１の実施形態によるデータ処理装置の構成を示すブロック図である。図１において、１００は読込部であり、スキャナ装置などを用いて画像を読み込む。１０１は入力部であり、ユーザからの指示やデータを入力するもので、キーボードやポインティング装置を含む。１０２は蓄積部であり、バイナリデータやメタデータを蓄積する。蓄積部１０２としては、ハードディスクを用いるのが一般的である。１０３は表示部であり、蓄積部１０２に蓄積されたバイナリデータを表示したり、読込部１００で読み込まれた画像データを表示する。表示部１０３としては、ＣＲＴや液晶表示装置が一般的である。

【００２８】

１０４はＣＰＵであり、上述した各構成の処理のすべてに関わり、ＲＯＭ１０５とＲＡＭ１０６はその処理に必要なプログラムやデータを格納する領域、或いは作業領域をＣＰＵ１０４に提供する。なお、図３のフローチャートを参照して後述する本実施形態の処理手順を実現するための制御プログラムもＲＯＭ１０５に格納されているものとする。もちろん、蓄積部１０２にその制御プログラムを格納しておき、ＣＰＵ１０４による実行に応じてその制御プログラムがＲＡＭ１０６上へロードされるような構成であってもよい。

【００２９】

なお、第１の実施形態のデータ処理装置には上記以外にも、種々の構成要素が設けられているが、本発明の主眼ではないので、その説明については省略する。

【００３０】

つぎに、以上のように構成されたデータ処理装置において、メタデータをバイナリデータに登録する処理について説明する。図３は、第１の実施形態によるメタデータの登録処理を説明するフローチャートである。

【００３１】

図３において、まず、ステップＳ３０１で、ユーザによって指定されたバイナリデータをメモリ（ＲＡＭ１０６）上に読み込む。これは例えば所望のバイナリデータファイル名をキーボードから入力したり、ポインティング装置（例えばマウス）によって当該バイナリデータのアイコンを指示することによりなされる。次にステップＳ３０２において、ユーザによって指定された、メタデータが記述されているＸＭＬファイルをメモリ（ＲＡＭ１０６）上に読み込む。このＸＭＬファイルの指定も、キーボードからファイル名を入力したり、ポインティング装置（例えばマウス）で対応するアイコンを指示する等によって行われる。

【００３２】

次にステップＳ３０３で、メタデータを記述したＸＭＬファイルが適正形式のＸＭＬデータであるかを調べる。この適性形式の判定では、ＸＭＬファイルの記述フォーマットを満足しているか（例えば、タグの左右の括弧が正しく対をなしているか、タグ付けの形式が正しいかどうか等）がチェックされる。なお、適性形式のＸＭＬデータであるか否かの判定は、正当なＸＭＬデータであるか否かを含めたチェックであってもよい。ここで、正当なＸＭＬデータか否かの判定は、例えば、ＸＭＬデータがＤＴＤ（Document Type Definition）等のスキーマに従って記述されているかどうか等のチェックを行うことでなされる。

【００３３】

ステップＳ３０３において適正形式のＸＭＬデータでないと判定された場合にはステップＳ３０５に進む。ステップＳ３０５では、ＸＭＬデータにエラーがある旨を表示部１０３に表示し、本処理を終了する。

【００３４】

一方、ステップＳ３０３においてＸＭＬファイルが適正形式のＸＭＬデータであると判定された場合には、処理はステップＳ３０４に進む。ステップＳ３０４では、ステップＳ３０１でメモリ上に読み込まれたバイナリデータの後ろに当該メタデータを接続することにより、メタデータの登録を行う。

【００３５】

次にステップＳ３０６で、当該バイナリデータ（ステップＳ３０１で読み込んだバイナリデータ）の種類を判断する。この判断は、たとえばバイナリデータが、ＪＰＥＧ符号化データでるとか、ＭＰＥＧ１符号化データあるとかを判断する。そしてその結果を基にステップＳ３０７で、ステップＳ３０４にて追加／登録したメタデータの最後にバイナリデータの終端コードを追加するか否かを判断する。この判断は、例えば画像データファイルフォーマットの仕様から、終端コードが存在するバイナリデータであるか否かを判断することにより行う。すなわち、終端コードを有するバイナリデータである場合は、追加すると判断され、ステップＳ３０８にて終端コードを追加した後にステップＳ３０９に進む。一方、終端コードを追加しないと判断された場合は、ステップＳ３０７から直接ステップＳ３０９に進む。

【００３６】

その後、ステップＳ３０９において、メタデータを追加／登録したバイナリデータを出力し、処理を終了する。なお、ステップＳ３０９におけるデータ出力により、図４に示されるデータ構造を有するデータが１つのファイルとして蓄積部１０２に格納されることになる。

【００３７】

なお、本実施形態の場合の終端コードとは、バイナリデータがＪＰＥＧ符号化方式で符号化されている場合は、ＥＯＩコードを、あるいはバイナリデータがＭＰＥＧ１符号化方式で符号化されている場合にはＳＥＣコードを入れる。また、バイナリデータファイルフォーマットに終端コードが無い場合は、メタデータの最後に終端を示すコードを接合しないように制御される。

【００３８】

以上説明したように、第１の実施形態によれば、メタデータをＸＭＬで記述し、バイナリデータの最後に接合することにより、既存のバイナリデータにメタデータを登録することができる。

【００３９】

図４は本実施形態によるバイナリデータへのメタデータの登録状態を説明する図である。図４に示されるように、メタデータをＸＭＬで記述し、このＸＭＬデータをバイナリデータの最後に接続し、さらにこのメタデータの最後に所定のコードを付加することにより、既存のアプリケーションに影響を及ぼすことなく、既存のバイナリデータにメタデータを登録することが可能となる。すなわち、メタデータが登録されたバイナリデータを、既存のアプリケーションで処理することが可能な形態で提供することができる。例えば、図４の場合、ＥＯＩコードを検索して処理が行われる場合であっても、確実にＥＯＩコードを検出でき、正しく処理が行われることになる。

【００４０】

また、メタデータとして既存のデータ記述言語を用いれば、メタデータの編集、参照等に際して、当該データ記述言語用の既存のツールをそのまま用いることができ、開発に関する手間も省くことができる。

【００４１】

例えば、本実施形態では、メタデータはＸＭＬで記述されているため、このＸＭＬデータ部分を抽出しておくことにより、ＸＭＬデータを理解するツールがあれば、メタデータの

追加・変更・参照が可能であり、非常に汎用性に優れている。なお、ＸＭＬデータ部分の抽出については第２の実施形態で詳しく説明する。

【００４２】

＜第２の実施形態＞

第１の実施形態では、バイナリデータのフォーマットに合わせて、そのバイナリデータの終端コードを、接合したメタデータの後ろに接合することを説明した。第２の実施形態では、予め、サポートする画像データファイルフォーマットを設定しておき、その画像データファイルフォーマットのサポート上必要な終端コード全てをメタデータの最後に接合する場合を説明する。

【００４３】

たとえばＪＰＥＧとＭＰＥＧ１の画像データファイルフォーマットをサポートするときは、ＪＰＥＧのＥＯＩコードと、ＭＰＥＧ１のＳＥＣコードの両方をメタデータの最後に追加する。図８は第２の実施形態によるメタデータが追加／登録されたバイナリデータのデータ構成を示す図である。図８では、ＪＰＥＧの終端コードであるＥＯＩコードと、ＭＰＥＧ１の終端コードであるＳＥＣコードの２つのコードが入っている。このときの挿入順は、図８に示される順序に限らないことは言うまでもない。また、対象となるバイナリデータフォーマットは、ＪＰＥＧやＭＰＥＧだけでなく、その他のフォーマットのバイナリデータであっても良い。

【００４４】

なお、第２の実施形態によるメタデータの登録処理は、図３に示したフローチャートにおいてステップＳ３０６とステップＳ３０７を省略して無条件でステップＳ３０８を実行するようにし、ステップＳ３０８では、上述のようにサポートするフォーマットの終端コードを付加するようにすることで実現できる。このため、第１の実施形態におけるステップＳ３０６に示す、バイナリデータのフォーマットを判断する処理を省略でき、高速に処理することが可能となる。

【００４５】

＜第３の実施形態＞

第１及び第２の実施形態においてバイナリデータにメタデータを登録する方法を説明した。第３の実施形態では、バイナリデータにメタデータが登録されているかどうかを判別し、登録されている場合にはそのメタデータを抽出する処理について説明する。なお、第３の実施形態におけるデータ処理装置の構成は第１の実施形態（図１）と同様であるのでここでは説明を省略する。

【００４６】

以下、指定されたファイルのデータに第１或いは第２の実施形態で説明した如きメタデータが登録されているか否かの判定と、登録されたメタデータを抽出する動作について説明する。図５は第３の実施形態による登録されたメタデータの判別及び抽出手順を示すフローチャートである。なお、本実施形態では、抽出されたメタデータを表示部１０３に表示するが、出力の形態はこれに限らない。例えば、抽出したメタデータを検索処理に提供するように構成してもよいことは当業者には明らかであろう。

【００４７】

図５によれば、まず、ステップＳ５０１で、ユーザの指示により、メタデータが登録されているかを判別したいファイル、即ち処理対象データを指定する。ステップＳ５０１における、処理対象データの指定は、キーボードから当該バイナリデータのファイル名を入力したり、対応するアイコンをポインティング装置（マウス）で指示することにより行われる。

【００４８】

次にステップＳ５０２において、指定されたファイルのデータにＸＭＬで記述されたメタデータが登録されているかどうかを判別する。以下、ステップＳ５０２における判別処理の詳細について図６のフローチャートと、図７の概略図にしたがって説明する。図６は第２の実施形態によるメタデータの判別処理の詳細を説明するフローチャートである。また

、図７はメタデータとしてＸＭＬデータが登録されたバイナリデータのデータ構成例を示す図である。

【００４９】

第１の実施形態で説明したように、メタデータとしてのＸＭＬデータが登録されている処理対象データのデータ構成は図７のようになっている。したがって、メタデータの有無の判別は以下のように行われる。

【００５０】

図６に示されるように、まず、ステップＳ６０１で、ステップＳ５０１で指定されたファイルのデータ全体（処理対象データの全体）をメモリ（ＲＡＭ１０６）上に読み込む。なお、第１の実施形態のステップＳ３０６によって出力されたデータは一つのファイルとして管理されるので、一般的なファイル管理システムによってこのデータの全体を読出すことが可能である。

【００５１】

次にステップＳ６０２において、ステップＳ６０１で読み込んだデータの最後に、終了コードに続いて“＜/PhotoXML＞”という文字列があるか調べる。存在しなかった場合はステップＳ６０５に進む。

【００５２】

一方、読み込んだ処理対象データの最後に、終了コードに続いて“＜/PhotoXML＞”という文字列が存在した場合はステップＳ６０３に進む。ステップＳ６０３では“＜/PhotoXML＞”という文字列の前方に“＜PhotoXML＞”という文字列が存在するかどうかを調べ、さらにそれらの文字列で囲まれたデータが、ＸＭＬの適正形式で記述されているかを確認する。なお、このとき、ＸＭＬの正当なデータであるか否かの判定を含めて行うようにしてもよい。適性形式か否かの判定、正当なデータか否かの判定は、第１の実施形態（ステップＳ３０３）で説明したとおりである。

【００５３】

ステップＳ６０３において適正形式であることが確認された場合は、ステップＳ６０４にすすむ。ステップＳ６０４においては、メタデータが登録されているものと結論づけ、本処理を終了する。一方、ステップＳ６０３において適正形式であることが確認されなかった場合には、処理はステップＳ６０５に進む。ステップＳ６０５においては、メタデータは登録されていないものと結論づける。すなわち、ステップＳ６０２で、当該バイナリデータの最後に文字列“＜/PhotoXML＞”が存在しない場合、ステップＳ６０３で文字列“＜PhotoXML＞”が存在しない場合、或いはステップＳ６０３で内部の記述が適正でないと判定された場合には、処理はステップＳ６０５にすすみ、当該処理対象データにメタデータは登録されていないものと結論づける。

以上で、メタデータの判別を終了する。

【００５４】

次に、図５のフローチャートにもどる。図６のフローチャートで示される処理によってメタデータが登録されていると結論づけられた場合には、処理はステップＳ５０３に進む。ステップＳ５０３では、文字列“＜PhotoXML＞”と“＜/PhotoXML＞”で囲まれた部分のＸＭＬデータに基づいて登録されているメタデータの内容を表示し、処理を終了する。一方、ステップＳ５０２でメタデータが登録されていないと判定された場合にはそのまま処理を終了する。

【００５５】

以上説明したように、第３の実施形態によれば、メタデータ付きのバイナリデータと通常のバイナリデータとの判別を、データの末尾にＸＭＬデータが適正形式で記述されているか否かによって判別することが可能となる。また、メタデータが判別された場合には、そのメタデータを表示することが可能となる。

【００５６】

すなわち、第３の実施形態によれば、メタデータが登録されたバイナリデータとメタデータが登録されていないバイナリデータとを判別するとともに、登録されたメタデータを抽

出することが可能となる。従って、メタデータとして既存のデータ記述言語を用いれば、メタデータを用いた検索に際して、当該データ記述言語用の既存のツールをそのまま用いることができ、開発に関する手間も省くことができる。

【００５７】

なお、上記各実施形態では、メタデータとしてＸＭＬデータを用いたがこれに限られるものではない。例えば、ＳＧＭＬやＨＴＭＬ等のデータ記述言語であってもよい。また、バイナリデータとしては静止画像データ、動画像データ、音声データ等が挙げられる。

なお、第３の実施形態では、第１の実施形態の手順で提供されるデータ（図７）に対する処理を説明したが、第２の実施形態の手順で提供されたデータ（図８）を処理対象とすることももちろん可能である。この場合、ステップＳ６０２で認識される終了コードが複数個（第２の実施形態では２つ）となる。

【００５８】

なお、本発明は、複数の機器（例えばホストコンピュータ，インタフェイス機器，リーダ，プリンタなど）から構成されるシステムに適用しても、一つの機器からなる装置（例えば、複写機，ファクシミリ装置など）に適用してもよい。

【００５９】

また、本発明の目的は、前述した実施形態の機能を実現するソフトウェアのプログラムコードを記録した記憶媒体を、システムあるいは装置に供給し、そのシステムあるいは装置のコンピュータ（またはＣＰＵやＭＰＵ）が記憶媒体に格納されたプログラムコードを読出し実行することによっても、達成されることは言うまでもない。

【００６０】

この場合、記憶媒体から読出されたプログラムコード自体が前述した実施形態の機能を実現することになり、そのプログラムコードを記憶した記憶媒体は本発明を構成することになる。

【００６１】

プログラムコードを供給するための記憶媒体としては、例えば、フロッピディスク，ハードディスク，光ディスク，光磁気ディスク，ＣＤ－ＲＯＭ，ＣＤ－Ｒ，磁気テープ，不揮発性のメモリカード，ＲＯＭなどを用いることができる。

【００６２】

また、コンピュータが読出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているＯＳ（オペレーティングシステム）などが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００６３】

さらに、記憶媒体から読出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるＣＰＵなどが実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。

【００６４】

【発明の効果】

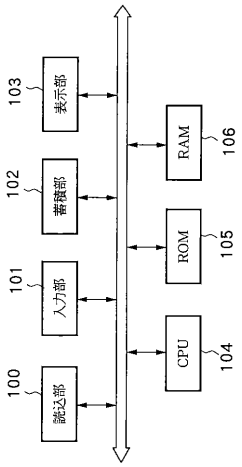
以上説明したように、本発明によれば、メタデータをデータ記述言語で記述し、バイナリデータの最後に接合することにより、既存のバイナリデータにメタデータを登録することができる。

また、メタデータが登録されたバイナリデータと通常のバイナリデータも容易に判別することが可能であり、さらに、メタデータの登録や検索に際しても、既存のデータ記述言語専用のツールをそのまま用いることができ、開発に関する手間も省くことができる。

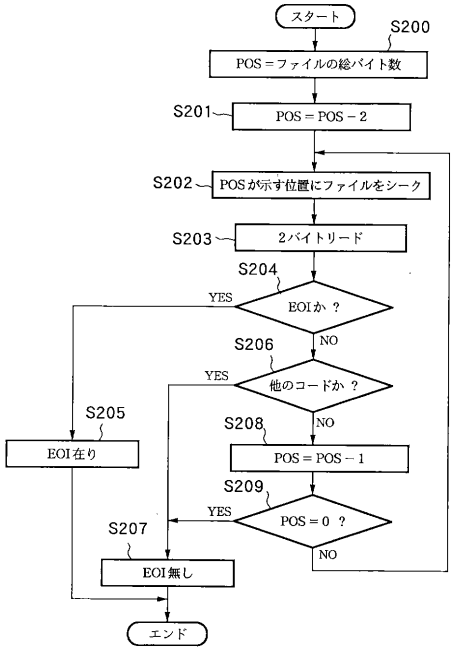
また、メタデータの最後にバイナリデータの終端コードを入れることにより、バイナリデータの完全性を判断する部分も高速に処理することができる。

【図面の簡単な説明】
【図１】第１の実施形態によるデータ処理装置の構成を示すブロック図である。
【図２】ＥＯＩマーカーコード検索の手順を示すフローチャートである。
【図３】第１の実施形態によるメタデータの登録処理を説明するフローチャートである。
【図４】本実施形態によるバイナリデータへのメタデータの登録状態を説明する図である。
【図５】第３の実施形態による登録されたメタデータの判別及び抽出手順を示すフローチャートである。
【図６】第２の実施形態によるメタデータの判別処理の詳細を説明するフローチャートである。
【図７】メタデータとしてＸＭＬデータが登録されたバイナリデータのデータ構成例を示す図である。
【図８】第２の実施形態によるメタデータが追加／登録されたバイナリデータのデータ構成を示す図である。
【図９】バイナリデータにメタデータを埋め込んだフォーマットの概観を示す図である。
【図１０】バイナリデータとメタデータをデータベースで管理する方法を概念的に示した図である。

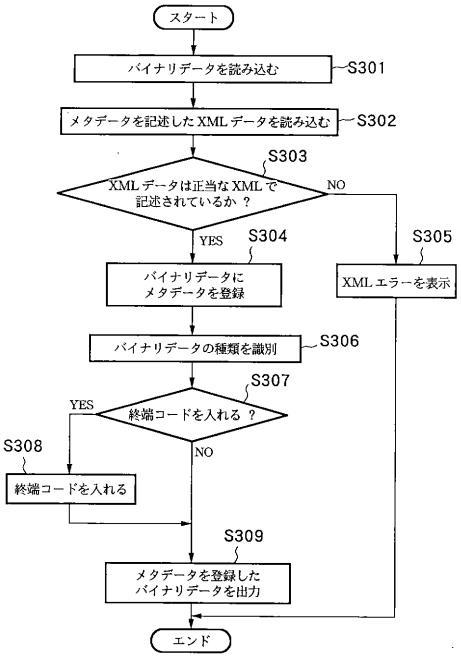
【図１】



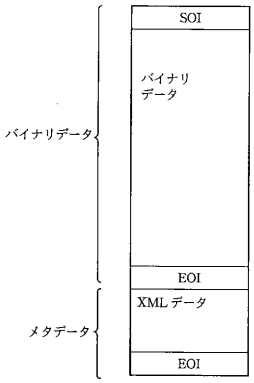
【図２】



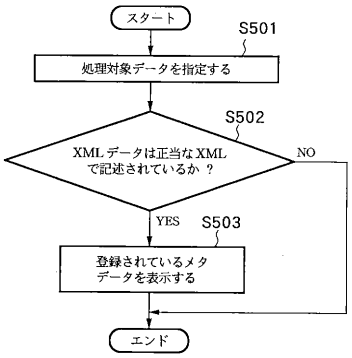
【図３】


スタート
バイナリデータを読み込む
S301
メタデータを記述したXMLデータを読み込む
S302
S303
XMLデータは正当なXMLで記述されているか ？
NO
YES
S304
バイナリデータにメタデータを登録
S305
XMLエラーを表示
バイナリデータの種類を識別
S306
S307
終端コードを入れる ？
YES
NO
S308
終端コードを入れる
S309
メタデータを登録したバイナリデータを出力
エンド

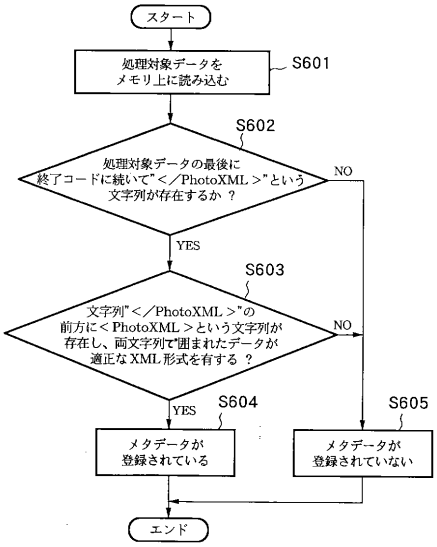
【図４】


SOI
バイナリデータ
EOI
XMLデータ
EOI
バイナリデータ
メタデータ

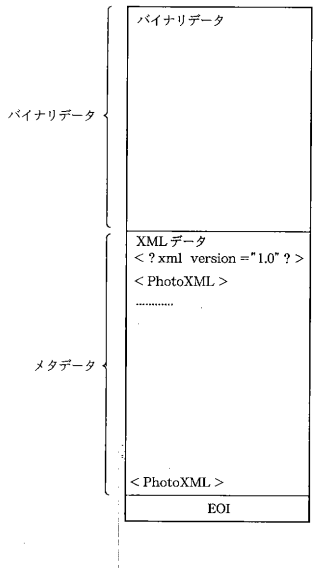
【図５】


スタート
S501
処理対象データを指定する
S502
XMLデータは正当なXMLで記述されているか ？
NO
YES
S503
登録されているメタデータを表示する
エンド

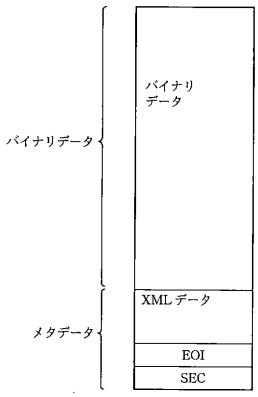
【図６】


スタート
処理対象データをメモリ上に読み込む
S601
S602
処理対象データの最後に終了コードに続いて"＜／PhotoXML＞"という文字列が存在するか ？
NO
YES
S603
文字列"＜／PhotoXML＞"の前方に＜PhotoXML＞という文字列が存在し、両文字列で囲まれたデータが適正なXML形式を有する ？
NO
YES
S604
メタデータが登録されている
S605
メタデータが登録されていない
エンド

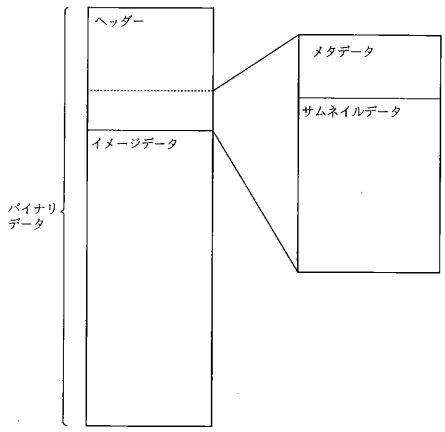
【図７】


バイナリデータ
バイナリデータ
XMLデータ
< ? xml version ="1.0" ? >
< PhotoXML >
.....
メタデータ
< PhotoXML >
EOI

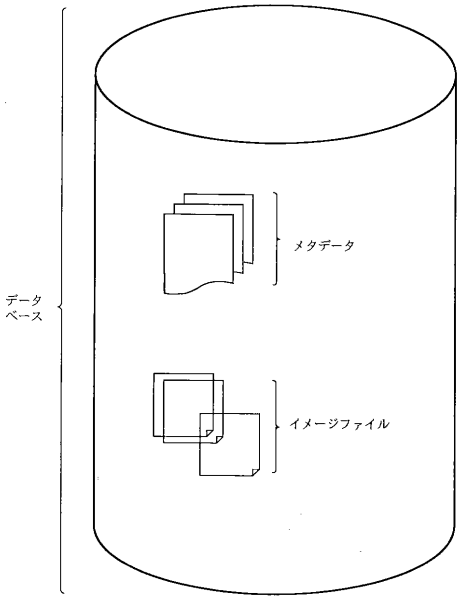
【図８】


バイナリデータ
バイナリ
データ
メタデータ
XMLデータ
EOI
SEC

【図９】


ヘッダー
メタデータ
サムネイルデータ
イメージデータ
バイナリ
データ

【図１０】


データ
ベース
メタデータ
イメージファイル

フロントページの続き

(72)発明者　榎田　幸
東京都大田区下丸子３丁目３０番２号　キヤノン株式会社内

審査官　高瀬　勤

(56)参考文献　特開平７－２７４１０８（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
G06F 12/00
G06F 17/30
H04N 5/92
JSTPlus(JDream2)