

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4409438号

(P4409438)

(45) 発行日 平成22年2月3日(2010.2.3)

(24) 登録日 平成21年11月20日(2009.11.20)

(51) Int.Cl.

F I

G 0 6 F 12/00 (2006.01)

G 0 6 F 12/00 5 0 1 B

請求項の数 16 (全 36 頁)

|               |                               |           |                       |
|---------------|-------------------------------|-----------|-----------------------|
| (21) 出願番号     | 特願2004-553945 (P2004-553945)  | (73) 特許権者 | 504421497             |
| (86) (22) 出願日 | 平成15年11月18日 (2003.11.18)      |           | イノパス・ソフトウェア・インコーポレ    |
| (65) 公表番号     | 特表2006-506747 (P2006-506747A) |           | ーテッド                  |
| (43) 公表日      | 平成18年2月23日 (2006.2.23)        |           | アメリカ合衆国カリフォルニア州9408   |
| (86) 国際出願番号   | PCT/US2003/037001             |           | 6-1105, サニーベール, イースト・ |
| (87) 国際公開番号   | W02004/046910                 |           | カリビアン・ドライブ 400        |
| (87) 国際公開日    | 平成16年6月3日 (2004.6.3)          | (74) 代理人  | 100089705             |
| 審査請求日         | 平成18年5月23日 (2006.5.23)        |           | 弁理士 社本 一夫             |
| (31) 優先権主張番号  | 10/298, 458                   | (74) 代理人  | 100076691             |
| (32) 優先日      | 平成14年11月18日 (2002.11.18)      |           | 弁理士 増井 忠武             |
| (33) 優先権主張国   | 米国 (US)                       | (74) 代理人  | 100075270             |
|               |                               |           | 弁理士 小林 泰              |
|               |                               | (74) 代理人  | 100080137             |
|               |                               |           | 弁理士 千葉 昭男             |

最終頁に続く

(54) 【発明の名称】 電子ファイル更新中におけるデバイス・メモリ管理

(57) 【特許請求の範囲】

【請求項 1】

電子ファイルの更新システムであって、

デルタ・ファイルを生成する第1デバイスであって、前記デルタ・ファイルは電子ファイルの1つのバージョンを別のバージョンと比較することにより生成され、前記デルタ・ファイルは前記比較の結果を示す電子ファイルを含む、第1デバイスと、

前記第1デバイスから前記デルタ・ファイルを電子的に受け取る第2デバイスと、  
を備え、前記第2デバイスが、

前記デルタ・ファイルから新電子ファイルの少なくとも1つの新コンポーネントを読み出し、前記新電子ファイルが、元の電子ファイルの更新バージョンであり、

前記第2デバイスのメモリにおいて、第1および第2メモリ・エリアの順次探索を行うことにより、前記新コンポーネントを収容するのに十分大きいメモリ・ブロックを識別し予約し、前記新コンポーネントのサイズが前記第1および第2メモリ・エリアの利用可能なメモリ・ブロックのサイズを超過するとき、未使用メモリ・ブロックを使用するために前記第1メモリ・エリアの再書き込みを行い、前記予約したメモリ・ブロックを提供するために前記第1および第2メモリ・エリアを再分配し、

前記新コンポーネントを前記予約したメモリ・ブロックに書き込み、

前記新コンポーネントを、前記元の電子ファイルの対応するコンポーネントを格納した元のメモリ・ブロック以外のメモリ・ブロックに書き込んだときに、ベクトル・テーブルを更新する、

10

20

ことによって前記第2デバイスの電子ファイルを更新するように構成されており、前記ベクトル・テーブルは、前記第2デバイスの前記メモリに対するアクセス動作を実行するのに使用するものであり、

更に、

前記デバイスの主プログラムからの関数コールを受け、前記関数コールは対応する電子ファイルの識別情報を含み、

前記ベクトル・テーブルから前記対応する電子ファイルの開始アドレスを読み取り、

前記開始アドレスおよび前記識別情報を用いて、前記対応する電子ファイルに対するコールを生成することによって、

前記デバイスの前記電子ファイルにアクセスするように構成されている、更新システム。

10

【請求項2】

請求項1記載の更新システムにおいて、前記第2デバイスは、更に、電子ファイルの更新において用いるための第1および第2アプリケーション・プログラミング・インターフェース（API）を含む、更新システム。

【請求項3】

請求項1記載の更新システムにおいて、前記第2デバイスは、更に、電子ファイルへのアクセスにおいて用いるための第3アプリケーション・プログラミング・インターフェース（API）を含む、更新システム。

【請求項4】

請求項1記載の更新システムにおいて、前記第1デバイスは、前記第2デバイス上にホストされたソフトウェアの少なくとも1つのプロバイダによってアクセス可能な、プロセッサ・ベースのデバイスを含む、更新システム。

20

【請求項5】

請求項1記載の更新システムにおいて、前記第2デバイスは、パーソナル・コンピュータ、携帯用計算機、セルラ電話機、携帯用通信デバイス、およびパーソナル・デジタル・アシスタントの中から選択した、少なくとも1つのプロセッサ・ベースのデバイスを含む、更新システム。

【請求項6】

請求項1記載の更新システムにおいて、前記第1デバイスから前記デルタ・ファイルを電子的に受けることは、ワイヤレス結合、ワイヤード結合、混合ワイヤレス／ワイヤード結合、およびローカル・エリア・ネットワーク（LAN）、都市区域ネットワーク（MAN）、ワイド・エリア・ネットワーク（WAN）、企業固有のネットワーク、バックエンド・ネットワーク、インターネットを含む少なくとも1つのネットワークとの結合、ならびにフロッピー・ディスク、ハード・ディスク・ドライブ、およびCD-ROMディスクを含む着脱可能固定媒体、更に電話回線、バス、および電子メール・メッセージを介して受けることを含む、更新システム。

30

【請求項7】

請求項1記載の更新システムにおいて、前記元の電子ファイルおよび新電子ファイルはソフトウェア・ファイルを含み、該ソフトウェア・ファイルは、ダイナミック・リンク・ライブラリ・ファイル、共有オブジェクト・ファイル、埋め込みソフトウェア・コンポーネント（EBSC）、ファームウェア・ファイル、実行可能ファイル、16進データ・ファイルを含むデータ・ファイル、システム・コンフィギュレーション・ファイル、および個人的使用データを含むファイルを含む、更新システム。

40

【請求項8】

新電子ファイルの少なくとも1つの新コンポーネントを受ける手段であって、該新電子ファイルは、元の電子ファイルの更新バージョンである、手段と、

メモリにおいて第1および第2メモリ・エリアの順次探索を行うことによって、前記新コンポーネントを収容することができる十分大きなメモリ・ブロックを識別し予約する手段であって、前記新コンポーネントのサイズが前記第1および第2メモリ・エリアの利用可能メモリ・ブロックのサイズを超過するとき、未使用メモリ・ブロックを使用するため

50

に前記第1メモリ・エリアの再書き込みを行い、前記予約したメモリ・ブロックを提供するため前記第1および第2メモリ・エリアを再分配する、手段と、

前記新コンポーネントを前記予約したメモリ・ブロックに書き込むことによって、前記元の電子ファイルを更新する手段と、

前記新コンポーネントを、前記元の電子ファイルの前記対応するコンポーネントを格納した元のメモリ・ブロック以外のメモリ・ブロックに書き込んだとき、ベクトル・テーブルを更新する手段であって、前記ベクトル・テーブルは、前記メモリに対するアクセス動作を実行するのに使用する、手段と、  
を備えた装置。

【請求項9】

請求項8記載の装置において、該装置は、パーソナル・コンピュータ、携帯用計算機、セルラ電話機、携帯用通信デバイス、およびパーソナル・ディジタル・アシスタントの中から選択した、少なくとも1つのプロセッサ・ベースのデバイスを含む、装置。

【請求項10】

請求項8記載の装置において、前記メモリ・ブロックを識別し予約する手段は、第1アプリケーション・プログラミング・インターフェース（API）である、装置。

【請求項11】

請求項8記載の装置であって、更に、

前記デバイスの主プログラムから関数コールを受け取る手段であって、該関数コールは、対応する電子ファイルの識別情報を含む、手段と、

前記ベクトル・テーブルから前記対応する電子ファイルの開始アドレスを読み取る手段と、

前記開始アドレスおよび前記識別情報を用いて、前記対応する電子ファイルに対するコールを発生する手段と、

を備えた、装置。

【請求項12】

請求項11記載の装置において、関数コールを受け取る前記手段は、第2アプリケーション・プログラミング・インターフェース（API）である、装置。

【請求項13】

電子ファイルの更新システムであって、

デルタ・ファイルを生成する第1デバイスであって、前記デルタ・ファイルは電子ファイルの1つのバージョンを別のバージョンと比較することにより生成され、前記デルタ・ファイルは前記比較の結果を示す電子ファイルを含む、第1デバイスと、

前記第1デバイスから前記デルタ・ファイルを電子的に受ける第2デバイスと、  
を備え、前記第2デバイスが、

前記デルタ・ファイルを通じて新電子ファイルの少なくとも1つのコンポーネントを受け、前記新電子ファイルは、元の電子ファイルの更新バージョンであり、

前記受けたコンポーネントのサイズを判定し、

前記サイズが前記元の電子ファイルの対応するコンポーネントのサイズ以下であるとき、前記受けたコンポーネントを、前記元の電子ファイルの前記対応するコンポーネントを含む元のメモリ・エリアに書き込み、

前記サイズが前記元の電子ファイルの前記対応するコンポーネントのサイズを超過するとき、前記受けたコンポーネントを、前記元のメモリ・エリアと関連した予約メモリ・エリアに書き込む、

ことにより前記第2デバイスの電子ファイルを更新するように構成された、更新システム。

【請求項14】

プロセッサとメモリとを備えた電子デバイスにおけるメモリ管理のための方法であって、

前記電子デバイスの電子ファイルを更新するステップであって、

10

20

30

40

50

前記プロセッサにより実行されるアップグレード・クライアントが、アップグレード・サーバから、新電子ファイルの少なくとも1つの新コンポーネントを受けるステップであって、前記新電子ファイルは、元の電子ファイルの更新バージョンである、ステップと、

前記アップグレード・クライアントが、前記電子デバイスの前記メモリ内の第1および第2のメモリ・エリアの順次探索を行うことによって、前記新コンポーネントを収容することができる十分大きなメモリ・ブロックを特定するステップであって、前記第1メモリ・エリアは、電子ファイルを格納したエリアであり、前記第2メモリ・エリアは前記第1メモリ・エリアに格納された前記電子ファイルの更新に使用するために予約されたメモリ・エリアであり、前記アップグレード・クライアントは、前記新コンポーネントのサイズが前記元の電子ファイルのうちの対応するコンポーネントのサイズを超えるときは、前記新コンポーネントを収容することができる十分大きなメモリ・ブロックを特定するため、前記第2メモリ・エリアを探索し、さらに該第2メモリ・エリアに利用可能なメモリ・ブロックがないときには前記第1メモリ・エリアにおける未使用メモリのブロックを探索し、利用可能な未使用メモリ・ブロックが前記第1メモリ・エリアにないときは前記第1メモリ・エリア内の未使用メモリ・ブロックを除くように前記メモリをパックして前記第2メモリ・エリアを大きくすることにより前記第2メモリ・エリア内でメモリ・ブロックを特定できるようにする、ステップと、

前記アップグレード・クライアントが、前記特定したメモリ・ブロックに前記新コンポーネントを書き込むステップと、

前記元の電子ファイルのうちの前記新コンポーネントに対応するコンポーネントを含む元のメモリ・ブロック以外のメモリ・ブロックに対し、前記新コンポーネントを書き込んだとき、前記アップグレード・クライアントが、ベクトル・テーブルを更新するステップであって、前記ベクトル・テーブルは、前記メモリにおける電子ファイルの各コンポーネントが格納されたメモリ・ブロックの開始アドレスを含む、ステップと、  
により行う、電子ファイルを更新するステップと、

前記電子デバイスの前記プロセッサが前記メモリに格納された前記電子ファイルにアクセスするステップであって、

前記アップグレード・クライアントが、前記電子デバイスの主プログラムからの関数コールを受けるステップであって、前記関数コールは、対応する電子ファイルの識別情報を含む、ステップと、

前記アップグレード・クライアントが、前記対応する電子ファイルの開始アドレスを前記ベクトル・テーブルから読み出すステップと、

前記アップグレード・クライアントが、前記開始アドレスと前記識別情報とを使って前記対応する電子ファイルに対するコールを生成するステップと、  
により行う、アクセスするステップと、

を備えた方法。

#### 【請求項15】

請求項14記載の方法において、前記識別情報は、関数識別とこの関数に対する引数とを含む、方法。

#### 【請求項16】

請求項14記載の方法において、前記特定するステップは、前記新コンポーネントのサイズが前記元の電子ファイルの対応するコンポーネントのサイズ以下のとき、前記第1メモリ・エリアの元のメモリ・ブロックを特定することを含み、前記元のメモリ・ブロックは、前記元の電子ファイルの前記対応するコンポーネントを含む、方法。

#### 【発明の詳細な説明】

#### 【技術分野】

#### 【0001】

開示する実施形態は、電子ファイル更新中におけるメモリ管理に関する。

#### 【背景技術】

## 【0002】

## 関連出願

本願は、以下の米国特許出願に関連がある。2002年5月13日に出願された出願番号第10/146,545号、2002年9月30日に提出された特許出願第10/261,153号、2002年11月12日に提出された特許出願第10/292,245号、2002年11月18日に提出された出願番号10/298,393号、2002年11月18日に提出された出願番号10/298,863号、2002年11月18日に提出された出願番号10/298,862号、および2002年11月18日に提出された出願番号第10/298,896号。これらの全ては、現在係属中である。

## 【0003】

10  
ホスト・デバイスのある機能を実現するために中央演算装置（CPU）上で実行するソフトウェアは、時間と共に変化していくことが多い。この変化は、ソフトウェア・バグを訂正する必要性から生じたり、発展する技術に適合するためであったり、あるいはホスト・デバイスに新たな特徴を追加するためであると考えられる。特に、セルラ電話のような移動ワイヤレス・デバイスにおける埋め込みソフトウェア・コンポーネントは、他の携帯のデバイスまたは処理システムよりも多くのソフトウェア・バグを含み、したがって、新たな特徴をそのデバイスに追加するためにより頻繁な更新を一般に必要とする。

## 【0004】

20  
ソフトウェアは、人が読むことができる情報相互交換用アメリカ標準コード（ASCII）の平文ファイルまたは二進コードの形態の1つ以上のファイルを含む。ソフトウェア・ファイルは、更に小さな単位に分割することができ、多くの場合これらをモジュールまたはコンポーネントと呼ぶ。UNIX（登録商標）プラットフォームまたはパーソナル・コンピュータ（PC）は、多数のソフトウェア・コンポーネントを含み、これらソフトウェア・コンポーネントの各々は、対応するオペレーティング・システム（OS）が対応するファイル・システムを通じて独立して管理および更新が行われる。UNIX（登録商標）プラットフォームまたはPC上にホストされたソフトウェア・ファイルまたはソフトウェア・コンポーネントを更新する際に用いられる情報は、インターネットを通じて転送したり、あるいはフロッピー・ディスク、コンパクト・ディスク・リード・オンリ・メモリ（CD-ROM）、またはコンパクト・フラッシュ・カードのような二次的な記憶媒体からロードすることができる。

## 【0005】

30  
対照的に、移動ワイヤレス・デバイスでは、リアル・タイム・オペレーティング・システムOS（RTOS）が使われるのが通例であり、この場合全てのソフトウェア・コンポーネントは単一の大きなファイルとしてリンクされている。更に、これらの移動ワイヤレス・デバイスには、ファイル・システムの対応は設けられていないのが通例である。加えて、単一の大きなファイルは、無線、赤外線、またはシリアル・リンクのような低速通信リンクを用いて、デバイス内に予めロードする、即ち、埋め込む必要がある。

## 【0006】

40  
低速通信リンクを通じて移動ワイヤレス・デバイスの大きなファイルを更新する場合の障害には、更新したファイルをデバイスに配信するのに伴う時間が含まれる。差分ファイルを、更新／改訂の目標となるファイルをホストしているデバイスに転送する場合、この時間は、差分ファイルを転送するための通信時間、ならびに受信側装置が差分ファイルを処理し、目標のファイルを更新／改訂するために用いられる時間の双方を含む。これを、ここでは、ホスト・デバイス処理時間と呼ぶ。関連出願に記載されているように、更新したソフトウェア・ファイルまたはソフトウェア・コンポーネントを転送するためにデルタ・ファイルを用いると、通信時間が大幅に短縮するが、ホスト・デバイスの処理時間の短縮は、少なくとも部分的に、ホスト・デバイスのアーキテクチャによって決まる。

## 【0007】

50  
ホスト・デバイスは、プロセッサを用いた多数のデバイスのいずれでも可能であり、セルラ電話機およびその他の移動通信デバイス、パーソナル・デジタル・アシスタント（

PDA)、ならびにパーソナル・コンピュータを含む。一例としてセルラ電話機を用いると、典型的なセルラ電話機のアーキテクチャは、ここではフラッシュROMまたはフラッシュ・メモリと呼ばれる、フラッシュ・リード・オンリ・メモリ(ROM)と、RTOSとを含む。フラッシュ・メモリおよびRTOSの双方は、ホスト・デバイスの処理時間を短縮するという目標に対して障害を招く潜在性がある。

【発明の開示】

【発明が解決しようとする課題】

【0008】

ホスト・デバイスの処理時間を短縮する際にフラッシュ・メモリに関して発生する問題は、フラッシュ・メモリに対して書き込みを行うやり方に関係する。フラッシュ・メモリは、ブロック、セクタ、セグメントと呼ばれることが多い単位で割り当てられ、その各々は多数のバイトを含む。典型的なフラッシュ・メモリへの書き込み(または書き換え)は、ユニット全体の消去および書き込み(または書き換え)を必要とし、単一ビットまたはバイトの書き込み(または書き換え)はできないことを意味する。したがって、フラッシュ・メモリ内に格納されている1バイトの更新または改訂には、当該バイトが格納されている単位全体の消去および書き換えが必要となる。このため、フラッシュ・メモリに格納されているファイルを更新または改訂するために必要となる時間は、単一のバイトを書き込む(または書き換えを行う)ことができる他の種類のストレージ即ちメモリに格納されているファイルを更新または改訂するために必要な時間よりも長いのが通例である。

【0009】

ホスト・デバイスの処理時間を短縮する際にRTOSに関して発生する問題は、ホスト・デバイス・ソフトウェアを編成し管理するやり方に関係する。典型的なセルラ電話機のRTOSは、実行時における関数コールに対するファイル管理および動的アドレス解明(dynamic address resolution)をサポートしていない。したがって、セルラ電話機に埋め込まれている全てのソフトウェア・コンポーネントまたはプログラムは、単一の大きなホスト・プログラムにリンクされており、ホスト・プログラム内にある機能は、ホスト・プログラムのいずれの部分からでも、予めコンパイルされている当該機能の開始アドレスを介して直接コールされる。このため、機能管理のためにファイル管理および動的アドレス解明をサポートする典型的なコンピュータOSとは対照的に、典型的なセルラ電話機のRTOSは、多数のソフトウェア・コンポーネントを別個のファイルとして管理することや、実行時コール元アドレス解明によって、ソフトウェア・コンポーネントの1つから他のソフトウェア・コンポーネントへの関数コールをサポートすることができない。その結果、RTOSを含むシステムのホストは、単一の大きなプログラムを管理し、プログラムの機能に直接アクセスするために静的アドレッシングを用いている。

【0010】

この典型的なデバイス・アーキテクチャは、ソフトウェア・コンポーネントを更新および改訂する際に、問題を生ずる可能性がある。何故なら、改訂または更新の結果ホスト・プログラム・サイズが変化した場合、ホスト・システムにおいて更新/改訂したソフトウェア・コンポーネントに続くソフトウェア・コンポーネントのアドレスも変化する結果となる。このソフトウェア・コンポーネントの開始アドレスの変化の結果、ホスト・プログラムにおいて対応するコール元アドレス、およびソフトウェア・コンポーネント内にある命令/データ・アドレスを更新する必要性が生ずる。これは、処理時間に関して費用が嵩む可能性がある。このように、埋め込みソフトウェアの更新/改訂に伴うホスト・デバイスの処理時間は、大部分が、フラッシュ・メモリへの非効率的なデータの書き換え、およびソース・コード・レベルでの埋め込みソフトウェアの小さな変化が、二進コード・レベルでの埋め込みソフトウェアに対する大きな変化となり得るという事実によるものである。

【発明を実施するための最良の形態】

【0011】

図面においては、同じ参照番号は、同一のまたは実質的に同様のエレメントあるいはア

クトを識別するものとする。いずれの特定のエレメントまたはアクトの論述であっても容易に識別するために、参照番号の内最上位の1つまたは複数の桁は、当該エレメントが最初に紹介された図番を示す（例えば、エレメント130は、図1において最初に紹介され、これについて論ぜられた）。

#### 【0012】

以下で特に記載しない限り、図に示す種々のブロックおよび構造の構成および動作は、従来の設計通りとする。その結果、このようなブロックは、ここでは更に詳細に説明する必要はなくなる。何故なら、これらは、当業者には理解されるはずであるからである。そのような更なる詳細は、簡潔性のため、そして本発明の詳細な説明を曖昧にしないようにするために、省略する。図に必要なあらゆる変更は、ここに示す詳細な説明に基づけば、当業者には容易に行うことができよう。

10

#### 【0013】

これより、メモリ管理システムおよび方法について詳細に説明する。メモリ管理を実行する際、ホスト・デバイスのアップグレード・クライアントは、第1および第2メモリ・エリアの順次探索を実行することによって、新たなソフトウェア・コンポーネントを収容できる十分な大きさのメモリ・ブロックを特定し予約する。新たなソフトウェア・コンポーネントは、元のソフトウェア・ファイルの更新またはアップグレード・バージョンである。新たなコンポーネントのサイズが、元の電子ファイルの対応するコンポーネントのサイズと同等以下の場合、アップデータ・クライアントは、第1メモリ・エリアの元のメモリ・ブロックを割り当て、この中に、受け取ったコンポーネントを書き込む。元のメモリ・ブロックは、元の電子ファイルの対応するコンポーネントを含む。

20

#### 【0014】

新たなコンポーネントのサイズが、元の電子ファイルの対応するコンポーネントのサイズを超過した場合、アップグレード・クライアントは、第1メモリ・エリアの未使用部分を探索して、受け取ったコンポーネントを格納できる十分な大きさのメモリ・ブロックを求める。第1メモリ・エリア内で突き止められた場合、アップグレード・クライアントは、しかるべきサイズのメモリ・ブロックを予約し、それ以外の場合、アップグレード・クライアントは第2メモリ・エリアを探索する。第2メモリ・エリア内で突き止められた場合、アップグレード・クライアントは、しかるべきサイズのメモリ・ブロックを予約する。

30

#### 【0015】

しかしながら、新たなコンポーネントのサイズが、第1および第2メモリ・エリアの使用可能なブロックのサイズを超過した場合、アップグレード・クライアントは、第1メモリ・エリアを書き換え、未使用メモリ・ブロックを排除する。第1メモリ・エリアの書き換えに続いて、アップグレード・クライアントは、順次第2メモリ・エリアのソフトウェア・コンポーネントを、第1メモリ・エリアのコンポーネント・サイズおよび使用可能なブロックが許される、第1メモリ・エリアの未使用部分に書き込むことによって、第1および第2メモリ・エリアを再分配する。続いて、アップグレード・クライアントは、第1メモリ・エリアの残りの未使用部分を、第2メモリ・エリアの一部として指定し直す。再分配に続いて、アップグレード・クライアントは、新たなコンポーネントを第2メモリ・エリアに書き込み、ベクトル・テーブルを更新する。

40

#### 【0016】

ホスト・デバイスのソフトウェア・コンポーネントへのアクセスに関して、アップグレード・クライアントは、ホスト・デバイスの主プログラムから関数コールを受け取る。ここで、関数コールは、対応するソフトウェア・ファイルの識別情報を含む。アップグレード・クライアントは、識別情報を用いてベクトル・テーブルにアクセスし、ベクトル・テーブルから対応するソフトウェア・ファイルの開始アドレスを読み出す。識別情報と共にベクトル・テーブルからの開始アドレスを用いて、アップグレード・クライアントは、対応するソフトウェア・ファイルに対するコールを発生する。

#### 【0017】

50

図1は、一実施形態の下において、バイト・レベル・ファイル差別化および更新(FDU)アルゴリズムを用いて電子ファイルを更新するシステムのブロック図である。このアルゴリズムを、ここではFDUアルゴリズムと呼ぶ。FDUアルゴリズムは、差別化コンポーネントと更新コンポーネントを含む。差別化コンポーネントは、ここではファイル差別化アルゴリズムと呼び関連出願において詳細に記載されているが、電子ファイルの元のバージョンおよび新バージョンから、第1コンピュータ・システムにおいて差分ファイルを生成する。更新コンポーネントは、ここではファイル更新アルゴリズムと呼び関連出願において詳細に記載されているが、差分ファイルおよび元のファイルの常駐コピー(hosted copy)を用いて、第2コンピュータ・システム上に新ファイルのコピーを生成する。以下の説明では、本発明の実施形態を完全に理解するため、そしてその説明を可能にするために、多数の具体的な詳細を紹介する。しかしながら、本発明は、これら具体的な詳細の1つ以上が無くても、または他のコンポーネント、システム等を用いても実施できることを、当業者は認めよう。その他の場合には、周知の構造または動作については、本発明の態様を曖昧にするのを防ぐために、図示も詳細な説明も行わない。

#### 【0018】

図1を参照すると、第1コンピュータ・システム102および第2コンピュータ・システム104は、通信経路106を通じて通信する。これらのコンピュータ102および104は、一緒に動作する計算デバイスのあらゆる集合体を含む。これは、当技術分野では周知である。通信経路106は、コンピュータ・システム102および104間でファイルを伝達または転送するためのあらゆる媒体を含む。したがって、この経路106は、ワイヤレス接続、ワイヤード接続、ハイブリッド・ワイヤレス／ワイヤード接続を含む。また、通信経路106は、ローカル・エリア・ネットワーク(LAN)、市内区域ネットワーク(MAN:metropolitan area network)、ワイド・エリア・ネットワーク(WAN)、企業固有のネットワーク、オフィス間またはバックエンド・ネットワーク、およびインターネットを含むネットワークに対する結合部即ち接続部も含む。更に、通信経路106は、フロッピー・ディスク、ハード・ディスク・ドライブ、およびCD-ROMディスクのような、着脱可能な固定媒体、ならびに電話線、バス、および電子メール・メッセージを含む。

#### 【0019】

第1通信システム102は、元の、即ち、古いバージョン110および新バージョン112の電子ファイルを受信する。新ファイル112は、一般には、元のファイル110の更新または改訂バージョンであるが、そのように限定される訳ではない。電子ファイル110および112に含まれるソフトウェア・ファイルには、ダイナミック・リンク・ライブラリ・ファイル、共有オブジェクト・ファイル、埋め込みソフトウェア・コンポーネント(EBSC)、ファームウェア・ファイル、実行可能ファイル、16進データ・ファイルを含むデータ・ファイル、システム・コンフィギュレーション・ファイル、および個人使用データを含むファイルが含まれるが、これらに限定される訳ではない。いずれの形式のファイルでもバイト・ストリームと見なすことができるので、以後文脈によってはファイルのことを、バイト・ストリームと記載することもあり得る。

#### 【0020】

ファイル差別化アルゴリズム114は、新ファイル112を受信し、それを元のファイル110と比較し、比較したファイル間におけるバイト・レベルの差を計算する。これについては以下で説明する。また、ファイル差別化アルゴリズム114は、元のファイル110および新ファイル112の前処理も行い、ファイルの差を計算するのに先だって、ファイル・サイズを縮小することもできる。ファイル差別化アルゴリズム114は、比較の間に、差分ファイル116を生成する。ここでは、これをデルタ・ファイルと呼ぶ。

#### 【0021】

デルタ・ファイル116の内容は、新ファイルおよび元のファイル間のバイト・レベルの相違を効率的に表す。デルタ・ファイル116は、実際の置換データおよび／または関連するファイルの最新即ち現バージョンとそのファイルの以前のバージョンとの間の差を表

す挿入動作と共に、メタデータを含む。これについては以下で説明する。ファイル差別化アルゴリズム 114 は、最少数のバイトおよび以下で説明する既定のフォーマットまたはプロトコルを用いて、デルタ・ファイル 116 の中に、元のファイル 110 と新ファイル 112 との間のあらゆる相違を提示することによって、空間的に最適化したデルタ・ファイルを提供する。

#### 【0022】

デルタ・ファイル 116 は、通信経路 106 を通じて、別のコンピュータ・システムに転送または送信される。転送の前に、当技術分野では公知の圧縮技法を用いて、デルタ・ファイル 116 を圧縮することもできるが、そのように限定される訳ではない。受信側のコンピュータ・システム上にホストされたファイル更新アルゴリズム 118 は、デルタ・ファイル 116 を、ホストされた元のファイル 110 と共に用いて、新ファイル 112 のコピーを生成即ち作成する。この新ファイル 112 のコピーは、次に、改訂または更新の対象である、クライアント・デバイス上にホストされた元のファイル 110 を更新する際に用いられる。この更新プロセスの完了時に、第 2 コンピュータ・システム上に現在格納されている新ファイルは、第 1 コンピュータ・システムにおいて受信した新ファイルと同一となる。

#### 【0023】

元のファイルと新ファイルとの間の相違は、通例では、新ファイル自体よりも小さいので、新たなバイト・ストリーム全体の代わりに、差分を送信し格納すれば、格納および送信において大幅な節約となる。これは、通例では遅くしかも費用がかかる可能性が高い接続部、例えば、ワイヤレスまたはセルラ接続部を通じて更新するプログラムをホストしている移動電子デバイスにとっては、特に重要である。

#### 【0024】

図 2 は、一実施形態のソフトウェア・アップグレード・システム 200 のブロック図である。このシステムを、ここではアップグレード・システムと呼ぶ。アップグレード・システム 200 は、一実施形態のデルタ・ファイルおよびファイル更新アルゴリズムを用いて、クライアント・デバイスのためにソフトウェア保守およびアプリケーション管理をサポートする。クライアント・デバイスには、移動電子デバイス、移動通信デバイス、セルラ電話機、パーソナル・ディジタル・アシスタント、コンピュータ、およびプロセッサを用いたデバイスが含まれる。このサポートは、通信業者(carrier)およびデバイス製造業者に、ワイヤレス・インフラストラクチャを通じて効率的に電子ファイル・コンポーネントやアプリケーションを流通することを可能にすることによって、ファームウェアから埋め込みアプリケーションまでの範囲に及ぶあらゆるデバイス・ソフトウェアに対して提供されている。

#### 【0025】

アップグレード・システム 200 は、サービス・プロバイダのワイヤレス・インフラストラクチャの種々の機構を用いて、新たなおよび改訂したソフトウェアのファイルの供給をサポートすることによって、デバイスのリコールを防止する。これらのシステムは、新たなおよび改訂したソフトウェアをソフトウェア流通業者から受け取り、ファイル差別化アルゴリズムを用いて新たなソフトウェアからデルタ・ファイルを生成することによって機能する。デルタ・ファイルは、サービス・プロバイダのインフラストラクチャを通じて、クライアント・デバイスに転送される。受信側即ちクライアント・デバイスのアップグレード・クライアントは、ファイル更新アルゴリズムを含み、デルタ・ファイルを用いて、クライアント・デバイス上にホストされた目標とされているソフトウェアを更新する。

#### 【0026】

アップグレード・システム 200 は、デバイスのソフトウェアおよびハードウェア間の異なるライフ・サイクルを可能にする。したがって、このアップグレード・システムは、Java（登録商標）Connected Limited Device Configuration（CLDC：接続限定デバイス・コンフィギュレーション）ライブラリ、ムービング・ピクチャ・エキスパーツ・グループ 1（MPEG-1）レイヤ III（MP3）ドライバ、通信ソフトウェア、お

10

20

30

40

50

よびブラウザ・アプリケーションが、ホスト・デバイスのハードウェアよりも速く発展し、その結果、移動デバイスの製造業者が新世代のデバイスを配給するよりも頻繁に更新されるという事実を鑑み、最新のデバイス機能をサポートする。アップグレード・システム200については、以下で更に詳細に説明する。

#### 【0027】

図2を参照すると、アップグレード・システム200は、デバイス212とのワイヤレス接続部を介して、クライアント・デバイス104上の埋め込みソフトウェア・コンポーネントを維持することによって、ワイヤレス通信業者が連続的に最新のデータ・サービスを全てのユーザに提供できるようにしている。アップグレード・システム200は、ソフトウェア・コンポーネント分配部(SCD)202、サービス・プロバイダ・アップグレード・コンポーネント203～205、およびクライアント・デバイス122上にホストされているアップグレード・クライアント130を含むが、これらに限定される訳ではない。サービス・プロバイダ・アップグレード・コンポーネントは、ソフトウェア・コンポーネント証明サーバ203およびアップグレード・マネージャ205との間に結合されているアップグレード・サーバ204を含む。

10

#### 【0028】

一実施形態のソフトウェア・コンポーネント分配部202は、ウェブに基づいたユーザ・インターフェースを設け、これによってソフトウェア供給業者は、改良したMP3ドライバ、アップグレードしたJava(登録商標)2プラットフォーム、Micro Edition(J2ME™) Mobile Information Device Profile(MIDP:移動情報デバイス特性)、または特徴付加アドレス・ブック・アプリケーションのような新たな埋め込みデバイス・ソフトウェア・コンポーネントをパッケージ化し、公表する。ソフトウェア・コンポーネント分配部202の機能には、デバイス情報を登録すること、およびデバイス情報をソフトウェア・コンポーネント証明サーバに提出することが含まれる。また、ソフトウェア・コンポーネント分配部202は、新EBS Cおよび元のEBS Cを受信し、ファイル差別化アルゴリズムを用いてこれらからファイル差を生成し、埋め込みソフトウェアを登録およびパッケージ化し、埋め込みソフトウェア・パッケージをソフトウェア・コンポーネント証明サーバに提出する。新たなソフトウェアは、公表に続いて、ワイヤード、ワイヤレス、あるいは混成ワイヤード/ワイヤレス・ネットワーク結合部または接続部220を通じて、サービス・プロバイダ・アップグレード・コンポーネント203～205に供給されるが、そのように限定される訳ではない。

20

30

#### 【0029】

一実施形態のソフトウェア・コンポーネント分配部202は、クライアント・デバイス製造業者の処理システム上にホストされている。代替実施形態では、ソフトウェア・コンポーネント分配部202は、アプリケーションまたはシステム・ソフトウェア供給業者の処理システム上にホストされている。別の代替実施形態では、ソフトウェア・コンポーネント分配部202は、通信サービス・プロバイダの処理システム上、例えば、アップグレード・コンポーネント203～205上にホストされている。

#### 【0030】

サービス・プロバイダ・アップグレード・コンポーネント203～205は、ソフトウェア・コンポーネント分配部202、クライアント・デバイス104、既存のゲートウェイ210や通信インフラストラクチャ212、請求書発行サーバ214、ロギング・サーバ(logging server)216、および認証サーバ218を含むサービス・プロバイダのインフラストラクチャ201～128の既存のコンポーネント間で結合されている。ソフトウェア・コンポーネント証明サーバ203は、クライアント・デバイスの製造業者へのインターフェースを設け、したがって、デバイス製造業者からの埋め込みソフトウェア・パッケージに関する新たなデバイス情報を受信する。また、ソフトウェア・コンポーネント証明サーバ203は、ソフトウェア・コンポーネント分配部からソフトウェア・コンポーネント提出要求を受け、新たなソフトウェア・パッケージの承認/拒絶を、提出したアップグレード・サーバに通知し、提出され承認されたソフトウェア・パッケージに対してディ

40

50

スク管理を行い、承認されたソフトウェア・パッケージを再パッケージ化(repackage)しアップグレード・サーバに分配する。更に、ソフトウェア・コンポーネント証明サーバ203は、潜在的に可能性がある侵入やデータ改竄に対して、ソフトウェア・コンポーネント提出プロセスの間に通信業者級のセキュリティ制御を行う。

#### 【0031】

アップグレード・マネージャ205は、ソフトウェア・コンポーネント証明サーバ203およびアップグレード・サーバ204間でインターフェースとして機能し、ウェブに基づくユーザ・インターフェースを設け、これによってワイヤレス・キャリア・システム管理運営者は、埋め込みデバイスのソフトウェア・コンポーネントのアップグレードを検証し承認する。また、アップグレード・マネージャ205は、最適なデバイス管理のためにソフトウェアおよびデータのパッケージングを行い、遠隔変更通知(remote change notification)のスケジューリングを行い、更新ポリシ・モニタ・システムを制御する。更に、アップグレード・マネージャ205は、既存のインフラストラクチャのシステム、即ち、バック・エンド・システム（請求書発行、ユーザ・データベース認証、ウェブ・ポータル）との統合を行うことによって、認証、アクセス制御、ならびにそれらの既存の請求書発行サーバ214およびロギング・サーバ216への統合を判定するワークフローを備えている。

10

#### 【0032】

アップグレード・サーバ204は、認証、接続、移動クライアント・デバイス122との通信を含む機能を備えており、埋め込みソフトウェア・コンポーネントのアップグレードを行う。クライアント・デバイス104との通信は、クライアント・デバイス122とのワイヤレス結合部212、ワイヤード結合部、混成ワイヤード／ワイヤレス結合部、および対応するサービス・プロバイダに適したその他のネットワーク結合部を介して行うことができる。加えて、アップグレード・サーバ204は、サービス・プロバイダの既存の請求書発行、データ収集、およびロギング・サービスにも対応する。

20

#### 【0033】

アップグレード・サーバ204およびクライアント・デバイス104間の通信の一例として、アップグレード・ファイルがアップグレード・サーバ204からクライアント・デバイス122に転送できるようになったときに、サーバ204はユーザ通知を送って、クライアント・デバイスのユーザに、更新に利用できるソフトウェア・コンポーネントがあることを通知する。ユーザ通知は、ショート・メッセージ・サービス(SMS)プッシュ・プロトコル、ハイパーテキスト・トランスファ・プロトコル(HTTP)、またはワイヤレス・アプリケーション・プロトコル(WAP)によるテキスト・メッセージの形態を取ることができるが、そのように限定されるのではない。ハンドセット・ユーザからの確認を受信すると、アップグレード・サーバ204は、元のハンドセット・データ通信プロトコルを用いて、アップグレード・ファイルを要求元のハンドセットに送る。

30

#### 【0034】

ハンドセットからの確認の受信に応答して、アップグレード・サーバ204は、ユーザおよび／または要求元デバイスを認証および許可し、要求元デバイスの欠くことができない機能および制限を検証する。認証に続いて、アップグレード・サーバ204は、クライアント・デバイス・コンフィギュレーション・データのマネージャとして、要求元デバイス104の埋め込みソフトウェア・コンポーネントの現バージョンを特定し、該当するデルタ・ファイルを特定し要求元デバイス104に転送し、アップグレード・トランザクションのステータスを記録し、結果をアップグレード・マネージャ205に報告する。アップグレード・サーバ204の一実施形態は、自動故障復元機構を含む。加えて、アップグレード・サーバ204は、無線によってソフトウェア・アップグレード・サービスの活性化／不活性化を行い、遠隔ユーザにソフトウェアの変更を通知する。

40

#### 【0035】

図1を参照すると、アップグレード・クライアント130は、クライアント・デバイス104のデバイス・メモリ300に埋め込まれているが、そのように限定される訳ではな

50

い。アップグレード・クライアント130は、ホスト・デバイス104のコンフィギュレーション・データを格納し維持し、ファイル更新アルゴリズム118を用いて、埋め込みデバイス・ソフトウェア・コンポーネントの維持および更新を担う。アップグレード・クライアント130は、単純なユーザ・インターフェースをサポートし、移動デバイスのソフトウェアに組み込まれる。実行時に、アップグレード・クライアント130は、自動的にあらゆる埋め込みソフトウェア・コンポーネントの起こりそうにない変化(remote change)を検出し、ユーザに埋め込みソフトウェア・コンポーネントのアップグレードを通知し、通信業者および／またはユーザの制御に基づいて、個々のサービス・プロバイダに対してしかるべくソフトウェア・コンポーネントをアップグレードする。また、アップグレード・クライアントは、自動障害復元機構も含む。

10

**【0036】**

クライアント・デバイスは、更新手順に関与する前に、多数のデバイス・パラメータのステータスを判断する。これを行うのは、更新手順にデバイスが適格性があるか予め調べるため、またはクライアント・デバイスの状態が、更新手順を一旦開始したなら完了できるようにになっていることを検証するためである。クライアント・デバイスの事前適格性検査は、クライアント・デバイスがクレードルまたは充電モードにあるか否か、クライアント・デバイスがシリアル・ケーブルに接続されているか否か、バッテリー充電の状態が更新プロセスを実行するのに十分か、受信信号強度指示(RSSI)または信号強度がデータ転送のために十分か、そして目標のEBS Cが現在使用中かについての判断を含む。

**【0037】**

20

一実施形態のアップグレード・システム200は、デルタ・ファイルによって、種々の形式のソフトウェア・ファイルまたはコンポーネントの更新をサポートする。更新をサポートするファイル形式には、実行可能ファイル、バイト・ストリーム・ファイル、およびデータ・ファイルが含まれるが、このように限定されるのではない。実行可能ファイル、または画像ファイルは、クライアント・デバイス内において用いられて、タスク、例えば、オペレーティング・システム(OS)、ハードウェア・デバイス・ドライバ、およびK仮想機械(KVM)ファイルを実行するソフトウェア・ファイルが含まれる。バイト・ストリーム・ファイルには、他の実行可能ファイルが用いるファイル、例えば、アイコン・ファイル、ロゴ・ファイル、およびMP3ファイルが含まれる。データ・ファイルには、個人使用データを収容したファイル、ならびにハンドセット基準ファイル、例えば、校正

30

**【0038】**

図3は、一実施形態の下における、デバイス・メモリ300のブロック図である。一実施形態のデバイス・メモリ300はフラッシュROMであるが、デバイス・メモリ300の代替実施形態では、多くの形式のメモリおよび／またはメモリ形式の組み合わせを用いることもできる。デバイス・メモリ300は、埋め込みソフトウェア・エリア302、アップグレード・クライアント・デバイス・パラメータのエリア306、およびデバイス・メモリ管理エリア308を含む。アップグレード・クライアント130は、ファイル更新アルゴリズム118と共に、一実施形態の埋め込みソフトウェア・エリア302に格納されている。デバイス・メモリ管理エリア308は、デバイス・メモリ管理(DMM)ライブラリ310、ベクトル・テーブル312、およびガベージ・テーブル314をホストしている。これらについては、以下で詳しく説明する。

40

**【0039】**

データ・ファイル、または代わりに新EBS Cの受信時に、アップグレード・クライアント130は、埋め込みソフトウェア、埋め込みソフトウェア・コンポーネント(EBS C)、およびEBS Cグループを含むファイルの改訂および更新を制御する。アップグレード・クライアントは、更新するファイル形式、およびこれらの更新をサポートするためにクライアント・デバイスの製造業者によって割り当てられたリソースに応じて、EBS Cを更新する多数の方法を用いる。これらの更新方法には、上記の関連する出願に記載さ

50

れた以下のものが含まれるが、それらに限定される訳ではない。即ち、オペレーティング・システム（OS）、通信プロトコル、およびその他のクリティカル・ソフトウェア・コンポーネントの、予約したRAMを用いての更新、予約ROMやRAMがない場合の通信プロトコルの更新、および非クリティカルEBS Cの単一線更新(single-line update)である。

#### 【0040】

一実施形態の更新方法は、非クリティカル・コンポーネントの更新およびクリティカル・コンポーネントの更新を含む。これらの分類は、更新の目標としたクライアント・デバイスのソフトウェア・コンポーネントの使用に基づいており、以下で更に説明する。

#### 【0041】

非クリティカル・コンポーネントは、更新プロセス中の障害に続いて無線で容易に復元される、埋め込みソフトウェア・コンポーネント（EBS C）を含む。非クリティカル・コンポーネントの例には、ブラウザおよびKVMファイルが含まれるが、そのように限定される訳ではない。図4は、一実施形態の下における非クリティカル・コンポーネントの更新のフロー図400である。デルタ・ファイルを用いて更新するとき、クライアント・デバイスは、ブロック402において、ネットワーク接続部を介してアップグレード・サーバからデルタ・ファイルを受信する。デルタ・ファイルは、クライアント・デバイスに受信されると、404において、デバイス・メモリ、例えば、RAMおよび／またはROMストレージの指定エリアに書き込まれる。次に、クライアント・デバイスのアップグレード・クライアントは、ブロック406において、デルタ・ファイルを用いて、元のファイルまたはEBS Cから新ファイルまたはEBS Cのコピーを生成する。ブロック408において、新ファイルのコピーは、予約したRAMまたはROMに書き込まれる。ブロック410において、新ファイルのコピーを、予約メモリから、元のファイルまたはEBS Cを収容するメモリ位置に順次書き込む。

#### 【0042】

クリティカル・コンポーネントは、更新手順で用いられるソフトウェア・コンポーネント、またはデバイスの動作に対して重要なEBS Cを含む。更に、クリティカル・コンポーネントは、更新プロセス中の障害に続いて無線で容易に復元されないEBS Cも含む。クリティカル・コンポーネントの例には、オペレーティング・システムのファイル、プロトコル・スタック、アップグレード・クライアント・ファイル、通信ライブラリ、およびディスプレイまたはLCDドライバ・ファイルが含まれるが、そのように限定される訳ではない。更新手順は、これら2つの分類間で多少異なる。

#### 【0043】

図5は、一実施形態の下におけるクリティカル・コンポーネントの更新のフロー図500である。デルタ・ファイルを用いて更新を行うとき、クライアント・デバイスは、ブロック502において、ネットワーク接続部を介してアップグレード・サーバからデルタ・ファイルを受信する。前述のように、デルタ・ファイルは、実行可能ファイル、バイト・ストリーム・ファイル、およびデータ・ファイルのファイルまたはEBS Cアップグレードを含むことができる。一実施形態のワイヤレス・ネットワークは、セルラ・サービス・プロバイダのそれであるが、そのように限定される訳ではない。

#### 【0044】

デルタ・ファイルの受信時、一般に、更新の目標とされた元のファイルまたはEBS Cを、ブロック504において、クライアント・デバイスのフラッシュROMから更新プロセス中の使用に指定されたメモリ・エリア、例えば、RAMおよび／またはROMストレージに適宜コピーする。クライアント・デバイスは、この指定したメモリ・エリアを、更新ソフトウェア・コンポーネントを格納する際に用いるために割り当てる。また、ブロック506において、デルタ・ファイルも、更新プロセス中の使用に指定されたメモリ・エリアに書き込まれる。

#### 【0045】

次に、ブロック508において、クライアント・デバイスのアップグレード・クライア

10

20

30

40

50

ントは、元のファイルのホストされたコピーを、デルタ・ファイルと共に用いて、新ファイルのコピーを生成する。ブロック510において、新ファイルのコピーを、クライアント・デバイスの、予め指定したメモリ・エリア、例えば、クライアント・デバイスの予約メモリ内に書き込む。このプロセスを必要に応じて繰り返す。

【0046】

生成に続いて、ブロック512において、新ファイルのコピーを、予約メモリから、元のファイルを収容している元のメモリ位置に書き込む。新ファイルのサイズが元のファイルのサイズ以下である場合、新ファイルを元のファイルのメモリ位置に書き込むことによって、元のファイルを置き換える。

【0047】

前述のように、ファイルの改訂には、新EBS Cおよび更新したEBS Cをホスト・デバイスのメモリ位置に書き込むことを伴う。多くの場合、新EBS Cは、正確に同じサイズではなく、それと置き換える元のEBS Cと正確に同じ開始アドレスを有していない。一実施形態のアップグレード・クライアントは、デバイス・メモリ管理選択肢を与える。選択肢には、EBS Cサイズや開始アドレスには関係なく、新EBS Cの書き込みに対処するためのアップグレード可能なEBS Cの静的アドレッシングおよび動的アドレッシングを含む。

【0048】

アップグレード可能なEBS Cの静的アドレッシングは、一般に、開始アドレスは不変のままにしつつ、予約メモリ・エリアを用いて、更新の間にEBS CまたはEBS Cグループのサイズを変更することができる。したがって、EBS CまたはEBS Cグループの開始アドレスの変更を必要とする場合静的アドレッシングを用いる場合、埋め込みソフトウェア（EBS）ファイル全体を書き換えるが、この実施形態はそのように限定される訳ではない。

【0049】

アップグレード可能なEBS Cの動的アドレッシングは、一層進んだメモリ管理を可能にしつつ、一般に各更新中EBS CおよびEBS Cグループの開始アドレスおよびサイズ双方の更新をサポートする。一実施形態のアップグレード・クライアントは、1組のアプリケーション・プログラムまたはプログラミング・インターフェース（API）および少なくとも1つのデータ・テーブルの間での相互作用を用いて、動的アドレッシングをサポートする。つまり、動的アドレッシングは、メモリ利用およびデバイス・メモリ・マップの設計の効率を高めつつ、更新障害の確率を低くして、更新処理の高速化をサポートする。静的および動的アドレッシングについては、以下で更に詳しく説明する。

【0050】

一実施形態の静的アドレッシングは、2つの選択肢があり、1つはアップグレード可能なEBS C毎に予約メモリの割り当てを含み、もう1つはEBS Cの特定のグループに対する予約メモリの割り当てを含む。予約メモリの割り当ては、ソフトウェア更新に対処するためのEBS CまたはEBS Cグループ毎のメモリ割り当てを変更する。したがって、ソフトウェアの今後の成長に対処するために、追加のROMを予約するが、追加のメモリはROMに限られる訳ではない。

【0051】

図6は、一実施形態の下におけるアップグレード可能なEBS C毎に予約メモリの割り当てを用いた静的アドレッシングのフロー図600である。動作において、アップグレード・クライアントは、ブロック602において、新EBS Cを受信する。新EBS Cは、ソフトウェア・コンポーネント分配部から受信することができ、あるいは元のEBS Cおよびデルタ・ファイルからアップグレード・クライアントによって生成することもできる。ブロック604において、アップグレード・クライアントは、新EBS Cのサイズを決定する。ブロック606において、アップグレード・クライアントは、新EBS Cのサイズが、それと置き換えることになっている対応するEBS Cのサイズを超過しているか否か判定を行う。新EBS Cのサイズが元のEBS Cのサイズを超過しない場合、アップグレ

10

20

30

40

50

ード・クライアントは、ブロック608において、元のEBS Cを格納しているメモリ・ブロックを予約し、続いてこれに新EBS Cを書き込む。新EBS Cサイズが元のEBS Cのサイズを超過する場合、アップグレード・クライアントは、ブロック610において、元のEBS Cを格納しているそのメモリ・ブロックおよび関連の予約メモリ・エリアに新EBS Cを最終的に書き込む。新EBS Cの書き込みに続いて、動作は、ブロック612における追加の新EBS Cの受信に戻る。

#### 【0052】

図7は、図6の実施形態の下において、静的アドレッシング・メモリ管理をサポートするように構成された、ホスト・デバイス・メモリ702のブロック図である。比較のために、静的アドレッシングをサポートするように構成されていないホスト・デバイス・メモリ704の部分が表示されている。静的アドレッシングをサポートするメモリ702は、各々のアップグレード可能なEBS Cに対し予約メモリ・エリアを含んでいる。この例では、アップグレード可能なEBS Cを、デバイスの製造時点で特定する。次いで、各アップグレード可能なEBS Cに対応する予約メモリ・エリアを設ける。予約メモリ・エリアは、対応するEBS Cと同じ位置にあるように示されているが、これらはデバイス・メモリ内のどこに配置しても、または分散してもよい。予約メモリ量を最少に抑えるという観点からは、予約メモリ・エリアは非更新可能EBS Cには設けられないが、実施形態はそうように限定される訳ではない。

#### 【0053】

この例を続けると、更新可能EBS Cは、EBS C1、EBS C2、およびEBS C5として特定され、予約メモリ・エリア1、2、および5がこれらの更新可能EBS C毎に設けられている。各予約メモリ・エリア1、2、および5は、デバイスの製造業者によって決定されたサイズである。今後の更新可能EBS Cに対するアップグレードおよび改訂では、更新／改訂によって、更新したEBS Cのサイズが元のEBS Cのサイズを超過した場合に、この対応する予約メモリ・エリアを用いる。これは、前述の通りである。

#### 【0054】

図8は、一実施形態の下において、アップグレード可能なEBS Cのグループへの予約メモリの割り当てを用いた、静的アドレッシングのフロー図800である。この場合も、新EBS Cをソフトウェア・コンポーネント分配部から受信するか、あるいは元のEBS Cおよびデルタ・ファイルからアップグレード・クライアントが生成してもよい。ブロック804において、アップグレード・クライアントは、新EBS Cのサイズを判定する。ブロック806において、アップグレード・クライアントは、新EBS Cのサイズが、それと置き換えることになる対応するEBS Cのサイズを超過したか否か判定を行う。新EBS Cのサイズが元のEBS Cのそれを超過していない場合、アップグレード・クライアントは、ブロック808において、対応する元のEBS Cを含むEBS Cグループを格納するメモリ・ブロックを予約し、これに新EBS Cを書き込む。新EBS Cのサイズが元のEBS Cのそれを超過する場合、アップグレード・クライアントは、ブロック810において、元のEBS Cを格納しているそのメモリ・ブロックおよび関連の予約メモリ・エリアを予約し、最終的に新EBS Cを書き込む。新EBS Cの書き込みに続いて、動作は、ブロック812において、追加の新EBS Cを受信するために戻る。

#### 【0055】

図9は、図8の実施形態の下において、静的アドレッシング・メモリ管理をサポートするように構成されたホスト・デバイス・メモリ902の一部のブロック図である。比較のために、静的アドレッシングをサポートするように構成されていないホスト・デバイス・メモリ904の部分が表示されている。静的アドレッシングをサポートするホスト・メモリ902は、グレードアップ可能なEBS Cの予め指定されたグループのために予約メモリ・エリアを含む。この代替案では、アップグレード可能なEBS Cは、デバイスの製造時点で特定する。この例では、EBS C1、2、4、5、および8がアップグレード可能である。

#### 【0056】

デバイス製造業者は、多数のファクタのうちの少なくとも1つを使用してアップグレード可能なE B S Cをグループ化する。例えば、類似のアップグレード頻度のE B S Cが1つのグループを形成するようにできる。また、アップグレードまたは改訂を受ける類似の確率をもつE B S Cが、1つのグループを形成するようにもできる。さらに、関係するあるいは相互に作用する機能をもつE B S Cが1つのグループを形成するようにできる。以下のグループ分けは、本例のために行った。すなわち、E B S C 1, 4, 5はE B S Cグループ1を構成し、E B S C 3, 7はE B S Cグループ2を構成し、E B S C 6, 9はE B S Cグループ3を構成し、E B S C 2, 8はE B S Cグループ4を構成する。

**【0057】**

グループ化に続いて、デバイスの製造業者は、各アップグレード可能E B S Cグループに対応する予約メモリ・エリアを設ける。予約メモリ・エリアは、対応するE B S Cグループと同じ位置にあるものとして示されているが、これらはデバイス・メモリ内のどこに位置してもよい。非アップグレード可能E B S Cが1つ以上のグループに置かれているが、予約メモリの量を極力抑える目的のために、予約エリアは設けられていない。しかし、実施形態はそうに限定される訳ではない。

**【0058】**

この例では、アップグレード可能E B S Cグループは、E B S Cグループ1、およびE B S Cグループ4として識別され、予約メモリ・エリアG 1およびG 4がこれらアップグレード可能E B S Cグループの各々に、それぞれ、設けられている。各予約メモリ・エリアG 1およびG 4は、デバイス製造業者が決定したサイズとなっている。アップグレード可能E B S Cグループの今後のアップグレードおよび改訂の際、アップグレード／改訂によってアップグレードされたE B S Cグループのサイズが元のE B S Cグループのサイズを超過した場合、対応する予約メモリを用いる。

**【0059】**

前述のE B S CまたはE B S Cグループの再書き込みの結果、再書き込みしたE B S CまたはE B S Cグループを含むファイル内にあるサブルーチンの開始アドレスが変化する可能性がある。サブルーチンの開始アドレスが変化すると、常に、ホスト・デバイスの他のサブルーチンが再書き込みしたサブルーチンにアクセスする際に用いる、対応のコール・アドレスも変化する。したがって、一実施形態では、アップグレード・クライアントは、再書き込みしたあらゆるE B S CまたはE B S Cグループに対応するコール・アドレスのアップグレードもサポートする。このコール・アドレスのアップグレードは、再書き込みしたE B S CまたはE B S Cグループにおけるサブルーチンに対応するコール・アドレスを含む、ホスト・デバイス・メモリ内の任意のブロックの再書き込みを含む。これらのコール・アドレスの変化は通例数バイトを伴うに過ぎないので、コール・アドレスを含むブロックのみをアップグレードすることは、コール・アドレスを収容するE B S Cを更新することよりも効率的であることがわかるが、実施形態はそうに限定される訳ではない。

**【0060】**

図10は、一実施形態の下において、静的アドレッシングをサポートするクライアント・デバイスROMマップ1000のブロック図である。ROMは、以下のエリア、ブート・コード・エリア1002、埋め込みソフトウェア・エリア1004、E B S Cワーキング・メモリ・エリア1006、アップグレード・クライアント・デバイス・パラメータ・エリア1008、および指定されていない目的のための少なくとも1つの予約エリア1010を含むが、そうに限定される訳ではない。

**【0061】**

ブート・コード・エリア1002は、デバイス・ブーティング・シーケンス・コードを格納する。埋め込みソフトウェア・エリア1004は、クライアント・デバイスの埋め込みソフトウェアを格納する。埋め込みソフトウェアは、例えば、ブラウザのソフトウェア、K仮想機械(KVM)、通信ライブラリ、リアル・タイムOS、グラフィック・ドライバ、およびアップグレード・クライアントを含むが、実施形態はそうに限定される訳

ではない。前述のように、ソフトウェアの更新に対処するために、各アップグレード可能 E B S C または E B S C グループに対するメモリ割り当てを変更する。したがって、ソフトウェアの今後の成長に対処するために予約された追加の R O M がある。この追加の予約 R O M は、対応する E B S C または E B S C グループの初期バージョンのサイズの約 5 % から 2 0 % であるが、そのように限定される訳ではない。

#### 【0062】

E B S C ワーキング・メモリ・エリア 1 0 0 6 は、アップグレード・クライアントがこれらを生成した後で、新バージョンが、現在元のバージョンの E B S C または E B S C グループが占有している埋め込みソフトウェア・エリア 1 0 0 4 のメモリ位置に書き込まれる前に、一時的に新バージョンの E B S C または E B S C グループを格納する。ワーク・  
エリア 1 0 0 6 の推定サイズは、このメモリ・エリアを用いる全ての E B S C または E B S C グループの内少なくとも最大のもののサイズであり、クリティカルおよび非クリティカル・コンポーネントを含む。

10

#### 【0063】

アップグレード・クライアント・デバイス・パラメータ・エリア 1 0 0 8 は、アップグレード・クライアントに特定のクライアント・デバイス・コンフィギュレーション・データおよびパラメータを格納する。このエリア 1 0 0 8 の推定サイズは、クライアント・デバイス上にホストされたデバイス・コンフィギュレーションに関連するアップグレード可能 E B S C の数に対応するが、そのように限定される訳ではない。

#### 【0064】

20

図 1 1 は、一実施形態の下において、静的アドレッシングを用いた埋め込みソフトウェアの開発および展開(deployment)プロセス 1 1 0 0 のブロック図である。この図は、例えば、ソフトウェア・コンポーネント分配部 1 1 0 2 による E B S C の開発から、静的アドレッシングに基づくデバイス・メモリ管理を備えた一実施形態のアップグレード・クライアントを用いたデバイス R O M の初期化までのプロセスを図示する。

#### 【0065】

ソフトウェア・コンポーネント分配部のコンポーネントは、新 E B S C のソース・コードを受信し(1 1 0 2)、新 E B S C ソース・コードをコンパイルする(1 1 0 3)。コンパイル(1 1 0 3)によって、得られた E B S C オブジェクト・コードを、リンク 1 1 0 6 および 1 1 1 6 に結合し、新 E B S C の登録の一部として、新 E B S C オブジェクト・コードから新たな 1 6 進ファイル(テキスト)または二進ファイル 1 1 1 8 を生成する。新たな 1 6 進ファイルは、新 E B S C に対応する。

30

#### 【0066】

静的アドレッシングをサポートするにあたり、ソフトウェア・コンポーネント分配部またはサービス・プロバイダは、元のマップ・ファイルを変更し、アップグレード可能 E B S C 毎に追加のメモリを挿入する。一実施形態において、リンカ 1 1 0 6 は、変更マップ・ファイル 1 1 0 4 を生成する。E B S C グループ化を用いる場合、元のマップ・ファイルを変更して、E B S C をグループ化し、アップグレード可能 E B S C グループ毎に追加のメモリを挿入する。このマップ・ファイル 1 1 0 4 の変更は、リンカ 1 1 0 6 を用いてリンク・ファイルを編集し、空ファイルまたはグループを対応するオブジェクト・ファイルと共に挿入することを含む。

40

#### 【0067】

ベクトル生成ツール 1 1 0 5 が、変更マップ・ファイル 1 1 0 4 からベクトル・テーブル・ファイル 1 1 0 8 を生成する。ベクトル・テーブル・ファイル 1 1 0 8 は、クライアント・デバイス 1 1 1 2 におけるアップグレード・クライアント・デバイス・パラメータ用予約 R O M エリア 1 1 1 0 にダウンロードされる。図 1 を参照すると、アップグレード・クライアント 1 3 0 は、ベクトル・テーブル 1 1 0 8 を維持しており、これは、一般に、埋め込みソフトウェア(E E B S)全体を書き換えるのであれば、不変のままである。また、リンカ 1 1 1 6 も変更マップ・ファイル 1 1 0 4 を用いて、新たな 1 6 進または二進ファイル 1 1 1 8 を生成する。更に、リンカ 1 1 1 6 は、1 6 進ファイル/マップ・

50

ファイル・パスの対の連携リストを指定または生成する。16進ファイル1118は、クライアント・デバイス1112の埋め込みソフトウェア・エリア1120にダウンロードされるが、代替実施形態では、クライアント・デバイス・メモリの他のエリアにダウンロードしてもよい。

#### 【0068】

前述の静的アドレッシングは、ホスト・デバイスの処理時間を短縮し、クリティカル・ソフトウェア・コンポーネントの更新を可能にするが、全てのアップグレードおよび改訂に対処する訳ではない。加えて、静的アドレッシングを用いると、いずれかのEBS Cの書き込みによって対応する予約メモリ・エリアのサイズを超過した場合には、デバイス・メモリ全体の書き換えが必要となる。したがって、動的アドレッシングを静的アドレッシングの代わりに用いる実施形態もある。

10

#### 【0069】

動的アドレッシングでは、新ファイル・バージョンのサイズが元のファイル・バージョンのそれを超過しても、電子ファイルの更新および改訂が可能である。動的アドレッシングをサポートする際、一実施形態のアップグレード・クライアントは、ファイル更新プロセス、および主プログラムが更新EBS Cを含む機能をコールする関数コール・プロセスの双方において、デバイス・メモリを効果的に管理する。

#### 【0070】

動的アドレッシングを用いてメモリ管理を実行する際、アップグレード・クライアントは、新たなソフトウェア・コンポーネントを収容できる程に十分大きなメモリ・ブロックを識別し予約する。その際、少なくとも1つの第1および第2メモリ・エリアを含むホスト・デバイスのメモリ・エリアの順次探索を行う。新たなコンポーネントのサイズが、探索したメモリ・エリアの使用可能なブロックのサイズを超過する場合、以下に述べるように、アップグレード・クライアントは第1メモリ・エリアを書き換えて、使用されていないメモリ・ブロックをなくし、第1および第2メモリ・エリアを再分配し、新たなコンポーネントを第2メモリ・エリアに書き込み、ベクトル・テーブルを更新する。

20

#### 【0071】

図12は、一実施形態の下において動的アドレッシングを用いてEBS CまたはEBS Cグループをアップグレードするためのフロー図1200である。アップグレード・クライアントは、ブロック1202において、新EBS Cバージョンを受信する。新EBS Cバージョンは、元のEBS Cバージョンおよび対応するデルタ・ファイルから生成するか、あるいはソフトウェア・コンポーネント分配部から受信する。ステップ1204において、クライアント・デバイス内で、新EBS Cバージョンの格納に対処するメモリ・エリアを特定し予約する。ブロック1206において、アップグレード・クライアントのコンポーネントは、新EBS Cバージョンを予約したメモリ・エリアに書き込む。ブロック1208において、ベクトル・テーブルに更新を行い、新EBS Cバージョンによって生じた、ベクトル・テーブルの情報に対するあらゆる必要な変更を行う。また、ブロック1210において、ガベージ・テーブルにも変更を行い、新EBS Cバージョンの書き込みによって生じた、未使用メモリ・エリアにおけるあらゆる変化を反映させる。

30

#### 【0072】

動的アドレッシングを用いてホスト・デバイスにおいてソフトウェア・コンポーネントにアクセスするために、アップグレード・クライアントは、対応するソフトウェア・ファイルの識別情報を含む関数コールを、ホスト・デバイスの主プログラムから受信し、ベクトル・テーブルから対応するソフトウェア・ファイルの開始アドレスを読み出し、対応するソフトウェア・ファイルにコールを生成する。こうして、第1EBS Cが第2EBS Cをコールすると、アップグレード・クライアントは関数コール・シーケンスを変更し、静的アドレスを用いて直接第2EBS Cをコールする代わりに、第1EBS Cが静的アドレスを用いて更新クライアントのAPIをコールするようにする。コールされたAPIは、対応する静的アドレスおよび実行時ベクトル・テーブルの引数を用いて、受け取ったコールを、第2EBS Cへのコールに変換する。

40

50

## 【0073】

図13は、一実施形態の下において、動的アドレッシングを用いた関数コールのフロー図1300である。ホスト・デバイスの主プログラムは、ホスト・デバイス・メモリに格納されているEBS Cを用いて、ユーザが要求する機能をサポートする。ユーザが機能を要求したとき、ホスト・デバイスの主プログラムのEBS Cは、静的アドレスを用いてアップグレード・クライアントのコンポーネントを直接コールすることによって、特定の機能に伴うEBS Cをコールする。コールに応答して、アップグレード・クライアントは、ブロック1302において、機能またはEBS Cの識別と当該機能に対する引数とを含む情報を、主プログラムから受け取る。ブロック1304において、アップグレード・クライアントは、EBS Cの対応する開始アドレスと共に、主プログラムに利用可能なEBS C 10  
毎のエントリを含むベクトル・テーブルにアクセスする。ブロック1306において、アップグレード・クライアントは、ベクトル・テーブルから、開始アドレスおよび関連する引数を読み出す。ブロック1308において、主プログラムから受け取った、コールしたEBS Cの情報と共に開始アドレス情報を用いて、アップグレード・クライアントは、実際の関数、従って対応するEBS Cに対するコールを、受け取った全引数と共に生成する。

## 【0074】

図14は、一実施形態の下において、動的アドレッシングをサポートするクライアント・デバイスROMマップ1400のブロック図である。ROMは、以下のエリア、ブート・コード・エリア1402、埋め込みソフトウェア・エリア1404、EBS C更新エリ 20  
ア1406、アップグレード・クライアント・デバイス・パラメータ・エリア1408、デバイス・メモリ管理エリア1410、および未指定の目的のために予約してある少なくとも1つのエリア1412を含むが、そのように限定される訳ではない。これらのメモリ・エリアの各々について以下で説明するが、代替実施形態では、当業者には認められる多くの異なるエリア・コンフィギュレーションを用いることができる。

## 【0075】

ブート・コード・エリア1402は、デバイス・ブーティング・シーケンス・コードを格納する。埋め込みソフトウェア・エリア1404は、クライアント・デバイスの埋め込みソフトウェアを格納する。埋め込みソフトウェアには、例えば、ブラウザのソフトウェア、KVM、通信ライブラリ、リアル・タイムOS、グラフィクス・ドライバ、およびア 30  
ップグレード・クライアントが含まれるが、そのように限定される訳ではない。

## 【0076】

EBS C更新エリア1406は、新バージョンの埋め込みソフトウェア・コンポーネントを格納する。このエリア1406の推定サイズは、デバイス間で変動する。しかしながら、一般に、EBS C更新エリア1406は、埋め込みソフトウェア・エリア1404のサイズの約10%から20%である。

## 【0077】

アップグレード・クライアント・デバイス・パラメータ・エリア1408は、アップグレード・クライアントに特定のクライアント・デバイス・コンフィギュレーション・データおよびパラメータを格納する。アップグレード・クライアント・デバイス・パラメータ 40  
・エリア1408の推定サイズは、クライアント・デバイス上にホストされたデバイス・コンフィギュレーションに関係するアップグレード可能なEBS Cの数に対応する。

## 【0078】

デバイス・メモリ管理エリア1410は、メモリ・パラメータ、ベクトル・テーブル、およびガベージ・エントリ・テーブルを格納するが、そのように限定される訳ではない。このエリア1410の推定サイズは、クライアント・デバイス上にホストされたデバイス・メモリ管理に関係するアップグレード可能なEBS Cの数に対応する。

## 【0079】

デバイス・メモリ管理エリア1410のメモリ・パラメータは、デバイス・メモリ管理のROM番号、開始アドレス、および予約ROMエリアのサイズ、ならびにデバイス・メ 50

メモリ管理 1410 のために予約した ROM エリアにおける開始アドレスおよび次に使用可能なメモリのサイズを含む。また、メモリ・パラメータは、ベクトル・テーブルおよびガベージ・エントリ・テーブルのサイズ、ならびにガベージ・エントリ・テーブルにおける有効エントリ数も含む。ベクトル・テーブルは、アップグレード可能 EBS C の EBS C 識別、ROM 番号、開始アドレス、およびサイズを含む。ガベージ・エントリ・テーブルは、全てのガベージ・エントリの ROM 番号、開始アドレス、およびサイズを含む。

#### 【0080】

図 15 は、図 3 および図 14 の実施形態の下における、アップグレード・クライアント 130、埋め込みソフトウェア・エリア 302、DMM ライブラリ 310、ベクトル・テーブル 312、およびガベージ・テーブル 314 を含むメモリのコンポーネント間の相互作用の一例を示すメモリ・デバイス 300 のブロック図である。DMM ライブラリ 310 は、3 つの API 1502 ~ 1506 を含む。これらの API 1502 ~ 1506 は、デバイス・メモリの管理のために予約した ROM エリア、EBS C のために予約したメモリ、および EBS C 更新のために予約したエリアを操作する際に用いられる。これらの API は、関数アドレス変換用 API 1502、EBS C の新バージョンの予約のための API 1504、およびガベージ・コレクション用 API 1506 を含み、その各々について以下に説明する。

#### 【0081】

関数アドレス変換 API 1502 は、図 13 を参照して先に説明したように、特定のホスト・デバイス機能に関連する EBS C に対するホスト・デバイスの主プログラム 1510 からのコールを発生する。ホスト・デバイスの主プログラム 1510 は、関数アドレス変換 API 1502 を直接コールすることによって、特定の機能に関連する EBS C をコールする。API 1502 の直接コールは、EBS C の直接コールの代わりに行われる。コールの時点において、関数アドレス変換 API 1502 は主プログラム 1510 から情報を受け取る。この情報は、機能識別情報、および機能に対する引数を含む。

#### 【0082】

コールに応答して、関数アドレス変換 API 1502 は、ベクトル・テーブル 312 にアクセスする。ベクトル・テーブル 312 は、主プログラム 1510 にとって利用可能な関数毎のエントリを、デバイス・メモリ 302 における当該関数の対応する開始アドレスと共に含む。関数アドレス変換 API 1502 は、主プログラム 1510 が要求する関数の開始アドレスをベクトル・テーブル 312 から読み取り、この開始アドレス情報を、主プログラム 1510 から受け取ったコール先 EBS C の情報と共に用いて、受け取った引数と共に実際の関数に対するコールを発生する。

#### 【0083】

一実施形態のアップグレード・クライアント 130 は、ここでは予約 API 1504 と呼ぶ、EBS C 1504 の新バージョンの予約のために API を用いて、新バージョンの EBS C または EBS C グループを収容するために、メモリ内のエリアを突き止めて予約する。図 15 を参照すると、予約 API 1504 は、アップグレード・クライアント 130 によってコールされると、新 EBS C バージョンの識別情報、即ち、この新バージョンのファイル・サイズを含む情報を受け取る。新 EBS C バージョンのファイル・サイズは、新 EBS C バージョンを格納するためにデバイス ROM 302 内に予約するエリアのサイズである。予約 API 1504 は、メモリ 302 内の該当するエリアを突き止め、メモリ・エリアの予約に成功したときに、要求された予約エリアの開始アドレスを返す。図 16 は、一実施形態の下において、予約 API 1504 を用いて EBS C の新バージョンを収容するためのメモリ・エリアを予約するためのフロー図 1600 である。

#### 【0084】

要求されたサイズのメモリ・ブロックを割り当てる際、予約 API 1504 は、1 組のルールを以下のように適用するが、代替実施形態では、異なるルールを適用して同等の結果を得ることもできる。図 17 は、一実施形態の下において、予約 API 1504 のルールを用いて EBS C の新バージョンを収容するためのメモリ・ブロックを突き止めるため

のフロー図1700である。ブロック1702において、新EBS Cのサイズに関する情報を受けると、予約API1504は、新EBS Cのサイズが元のEBS Cのそれを超えるか否か、ブロック1704において判定を行う。新EBS Cのサイズが対応する元のEBS Cのサイズ以下であれば、ブロック1706において、予約API1504は、対応する元のEBS Cが現在占有しているメモリ・ブロックを、新EBS Cを受けるために割り当てる。そうでなければ、予約API1504は、要求されたサイズを有するメモリ・ブロックを、予約されているメモリ・エリア内にある使用可能なメモリから突き止めようとする。

#### 【0085】

探索を続ける際、予約API1504は、ブロック1708において、クライアント・デバイスの予約されているメモリ・エリアを探索し、要求されたサイズを有するメモリ・ブロックを求める。適切なサイズのブロックが予約メモリ・エリア内で見つかった場合、ブロック1710において、予約API1504はそのメモリ・ブロックを、新EBS Cを受け取るために割り当てる。予約メモリ・エリア内に適切なサイズのメモリ・ブロックが得られない場合、ブロック1712において、予約API1504はガベージ・テーブル314にアクセスする。ブロック1714において、予約API1504は、ガベージ・テーブル314内のエントリに対応する未使用エリア間で、要求されたサイズのメモリ・ブロックを突き止める試みの中で、ガベージ・テーブル314の情報を用いて、主メモリ・プログラムの未使用メモリ・エリアを探索する。一実施形態の未使用エリアを、ここではガベージ・エリアと呼び、主プログラムの未使用エリアを含むが、実施形態はそのように限定されている訳ではない。適切なサイズのメモリ・ブロックが未使用メモリ・エリア内で見つかった場合、ブロック1716において、予約API1504はそのブロックを、新EBS Cを受け取るために割り当てる。

#### 【0086】

一実施形態の予約API1504は、適切なサイズのメモリ・ブロックを突き止めることができない場合、ブロック1718において、ガベージ・コレクションとここでは呼ばれているプロセスを開始する。また、予約API1504は、ガベージ・テーブル214のエントリ全てが塞がっているときにも、ガベージ・コレクションを開始する。一実施形態では、ガベージ・コレクション用API1506、ここではガベージ・コレクション1506と呼び、予約API1504によってコールされるが、そのように限定される訳ではない。ガベージ・コレクション用APIは、一般に引数を受け取らず、EBS CおよびEBS C更新のために予約したエリアにおける全てのガベージを消去するが、そのように限定される訳ではない。

#### 【0087】

前述のように、予約API1504は、要求された予約サイズに十分なメモリを見つけないことができない場合、またはガベージ・エントリ・テーブルが満杯になったときに、ガベージ・コレクションを開始する。図18は、一実施形態の下におけるガベージ・コレクションのフロー図1800である。ガベージ・コレクションを開始する際、予約API1504はガベージ・コレクションAPI1506をコールする。コールされると、ブロック1802において、ガベージ・コレクションAPI1506はガベージ・テーブル・エントリを読み取り、ブロック1804において、デバイス・メモリの埋め込みソフトウェア・エリアを書き直させ、メモリのEBS Cを順次メモリの上に向かってパックしていく。代替実施形態では、EBS Cを主メモリの下に向かってパックしてもよいが、実施形態はそのように限定されている訳ではない。このパッキング処理により、埋め込みソフトウェア・エリアにおける未使用エリアを除去する。

#### 【0088】

このパッキング処理に続いて、ガベージ・コレクションAPI1506は、ブロック1806において、予約メモリ・エリアに格納されている各EBS Cのサイズを評価し、そのサイズを、その書き換え以後における埋め込みソフトウェア・エリアの主プログラムに残っているあらゆる未使用メモリ・ブロックと比較する。この評価は、埋め込みソフトウ

エア・エリアの書き換えの後に残っている未使用メモリの量が、予約メモリ・エリアのいずれのE B S Cでも収容可能であり、可能であれば、E B S Cを予約メモリ・エリアから、デバイス・メモリの埋め込みソフトウェア・エリアに移動させることができるか否か判定を行う。予約メモリ・エリアのE B S Cは順次評価されるが、実施形態はそうに限定される訳ではない。ブロック1808において、E B S Cは、予約メモリ・エリアからE B S Cのサイズが許す埋め込みソフトウェア・エリアに書き直される。一実施形態において、E B S Cが書き直されると、これらは順次、パックされたE B S Cに続いて埋め込みソフトウェア・エリアに格納される。

#### 【0089】

予約メモリ・エリアに格納されたE B S Cの評価は、全てのE B S Cを埋め込みソフトウェア・エリアに移動し終えるまで、または埋め込みソフトウェア・エリア内に、予約メモリ・エリアのE B S Cを収容できる程大きな埋め込みソフトウェア・エリアが残っていないと判定されるまで継続する。埋め込みソフトウェア・エリア内に収容することができないE B S Cはいずれも、予約メモリ・エリア内に残るが、実施形態はそうに限定される訳ではない。

#### 【0090】

パッキングおよび移動処理が完了すると、ブロック1810において、ガベージ・コレクションA P I 1506は、ホスト・デバイスの主メモリの未使用メモリ・ブロックを予約メモリ・エリアとして指定し直すことにより、ホスト・デバイス・メモリを再分配する。更に、ステップ1812において、予約メモリ・エリア・ポインタをリセットして、予約メモリ・エリア内で使用可能なメモリの開始アドレスを維持する。加えて、ブロック1814において、ガベージ・テーブルを書き換え、未使用メモリ・エリアのステータスを反映させる。また、アップグレード・クライアント130は、同じE B S Cの元のバージョンが占有するメモリ・エリア以外のいずれかのメモリ・エリアへのE B S Cの新バージョンの書き込みの後に、ベクトル・テーブル312を更新する。

#### 【0091】

更新およびガベージ・コレクション処理の一例を、図15、図17、図19、および図20を参照しながら、以下に続ける。図19は、一実施形態の下において、予約A P Iのアップグレード処理の前1902および後1904におけるホスト・メモリ・デバイスの一例のブロック図である。図20は、一実施形態の下において、アップグレードおよびガベージ・コレクション処理の後におけるホスト・デバイス・メモリ2004のブロック図である。

#### 【0092】

この例は、予約A P IによるE B S C 8の新バージョンに関する情報の受信から始まる。新バージョンのE B S C 8のファイル・サイズを受け取ると、この新バージョンのサイズは元のE B S Cのサイズ以下か否か判定を行う。この例では、E B S C 8の元のバージョンは400バイトを収容し、一方E B S C 8の新バージョンは380バイトを収容する。したがって、予約A P Iは、現在E B S C 8の元のバージョンが占有しているエリア1920を、E B S C 8の新バージョンを受け取るために割り当てる。

#### 【0093】

本例は、予約A P IによるE B S C 5の新バージョンについての情報の受信に続く。E B S C 5の新バージョンのファイル・サイズを受け取ると、この新バージョンのサイズは元のE B S Cのサイズ以下か否か判定を行う。この例では、E B S C 5の元のバージョンは300バイトを収容し、一方E B S C 5の新バージョンは360バイトを収容する。新バージョンのサイズの方が元のバージョンのそれよりも大きいので、予約A P Iは、予約メモリ・エリア1910内で使用可能なメモリから、要求されたサイズを有するメモリ・ブロックを突き止めようとする。この例では、予約メモリ・エリア1910においてメモリが使用可能であるので、予約A P Iは、E B S C 5の新バージョンを受け取るために、予約メモリ1910内のエリア1922を割り当てる。更に、予約A P Iは、E B S C 5の元のバージョンが現在占有しているエリアを、未使用エリア1924として割り当てる

。

## 【0094】

次に、予約APIはEBS C 7の新バージョンに関する情報を受け取る。EBS C 7の元のバージョンは550バイトを収容し、一方EBS C 7の新バージョンは560バイトを収容する。新バージョンのサイズが元のバージョンのそれよりも大きいので、予約APIは予約メモリ・エリア1910内の使用可能なメモリから、要求されたサイズを有するメモリ・ブロックを突き止めようとする。この例では、予約メモリ・エリア1910内でメモリが使用可能であるので、予約APIは、EBS C 7の新バージョンを受け取るために、予約メモリ1910内で次に使用可能なエリア1926を割り当てる。更に、予約APIは、EBS C 7の元のバージョンが現在占有しているエリアを、未使用エリア1928として割り当てる。

10

## 【0095】

本例が継続すると、予約APIがEBS C 2の新バージョンに関する情報を受け取る。EBS C 2の元のバージョンは330バイトを収容し、EBS C 2の新バージョンは360バイトを収容する。予約APIは、予約メモリ・エリア1910内では適切なサイズのエリアが得られないと判断し、ガベージ・テーブル314にアクセスする。ガベージ・テーブル314は、ガベージ・テーブル314内のエントリを用いて識別されるガベージ・エリア間で、要求されたサイズのメモリ・ブロックを突き止める試みにおいて用いられる。この例では、該当するサイズの未使用エリア1930が特定され、予約APIは、EBS C 2の新バージョンを受け取るために、未使用エリア1930を割り当てる。

20

## 【0096】

前述のように、予約APIは、要求された予約サイズに十分なメモリを見つけることができない場合、またはガベージ・エントリ・テーブルが満杯のときに、ガベージ・コレクションを開始する。この例では、EBS C 2の新バージョンのための予約メモリ・エリア内での割り当てに続いて受け取った次の新EBS Cバージョンで、ガベージ・コレクションが開始されたと仮定する。一実施形態のガベージ・コレクションは、EBS Cをホスト・デバイス・メモリにおいて上に向けてパックし、新たなホスト・デバイス・メモリ・コンフィギュレーション1904を得る。この中では、EBS Cの順序は、EBS C 1、EBS C 4、EBS C 2、EBS C 3、EBS C 6、EBS C 5、およびEBS C 7となる。代替実施形態のガベージ・コレクション・ルーチンは、EBS Cをホスト・デバイス・メモリにおいて下に向けてパックする。

30

## 【0097】

図21は、一実施形態の下において、動的アドレッシングを用いた埋め込みソフトウェアの開発および展開プロセスのブロック図2100である。この図は、ソフトウェア・コンポーネント分配部2102によるEBS Cの開発から、動的アドレッシングに基づくデバイス・メモリ管理を行う一実施形態のアップグレード・クライアントを用いたデバイスROMの初期化までのプロセスを示す。

## 【0098】

ソフトウェア・コンポーネント分配部2102のコンポーネントは、新EBS Cのソース・コードを受信し、新EBS Cソース・コードをコンパイルする(2103)。コンパイル(2103)によって、得られたEBS Cオブジェクト・コードをリンカ2106および2116と結合し、新EBS Cの登録の一部として、新EBS Cオブジェクト・コードから新たな16進ファイル(テキスト)または二進ファイル2118を生成する。新たな16進ファイルは、新EBS Cに対応する。

40

## 【0099】

動的アドレッシングをサポートするに当たり、ソフトウェア・コンポーネント分配部2102は、コンパイラ2103およびリンカ2106を用いて、マップ・ファイル2104を生成する。このマップ・ファイル2104は、ベクトル生成ツール2105が、EBS Cの対応する初期ベクトル・テーブル2108を生成するために用いる。次いで、ベクトル・テーブル2108は、リンカ2116に供給され、リンカ2116は、初期ベクト

50

ル・テーブル 2108 を用いて、16 進（テキスト）または二進ファイル 2118 を生成する。16 進（テキスト）または二進ファイル 2118 は、クライアント・デバイス 2112 の埋め込みソフトウェア・エリア 2120 にダウンロードされるが、代替実施形態では、16 進ファイル 2118 をクライアント・デバイス 2112 の他のメモリ・エリアに書き込んでもよい。アップグレード・クライアントの DMM ライブラリのコンポーネントは、ベクトル・テーブル 2208 を維持し、こうして、EBC の動的アドレッシングをサポートする。DMM ライブラリは、アップグレード・クライアントのデバイス・パラメータ 2110 のために予約したエリアに格納されているベクトル・テーブル 2108 と重複する。

#### 【0100】

10

前述の電子ファイル更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、電子ファイルの更新システムを含み、このシステムは、デルタ・ファイルを生成する、ファイル差別化および更新システムの第 1 コンポーネントを含む第 1 デバイスと、第 1 デバイスから第 1 デルタ・ファイルを少なくとも 1 つの結合部を介して受け取る第 2 デバイスであって、ファイル差別化および更新システムの第 2 コンポーネントを含む第 2 デバイスとを備えている。

#### 【0101】

第 2 デバイスは、ファイル差別化および更新システムの第 2 コンポーネントを含み、元の電子ファイルの更新バージョンである新電子ファイルの少なくとも 1 つの新たなコンポーネントをデルタ・ファイルから読み出し、第 1 および第 2 メモリ・エリアの順次探索を行うことにより、新コンポーネントを収容するのに十分大きいメモリ・ブロックを識別し、予約し、未使用メモリ・ブロックを排除するために第 1 メモリ・エリアを書き換え、新コンポーネントのサイズが第 1 および第 2 メモリ・エリアの使用可能なメモリ・ブロックのサイズを超過するとき、第 1 および第 2 メモリ・エリアを再分配し、新コンポーネントを予約したメモリ・ブロックに書き込み、新コンポーネントを、元の電子ファイルの対応するコンポーネントを含む元のメモリ・ブロック以外のメモリ・ブロックに書き込んだときに、ベクトル・テーブルを更新することによって第 2 デバイスの電子ファイルを更新するように構成されている。

20

#### 【0102】

また、第 2 デバイスは、ファイル差別化および更新システムの第 2 コンポーネントも含み、デバイスの主プログラムからの、対応する電子ファイルの識別情報を含む関数コールを受け、ベクトル・テーブルから対応する電子ファイルの開始アドレスを読み取り、開始アドレスおよび識別情報を用いて、対応する電子ファイルに対するコールを生成することによってデバイスの電子ファイルにアクセスするように構成されている。

30

#### 【0103】

一実施形態のシステムの第 2 デバイスは、更に、電子ファイルを更新する際に用いるための第 1 および第 2 アプリケーション・プログラミング・インターフェース（API）を含む。

#### 【0104】

一実施形態のシステムの第 2 デバイスは、更に、電子ファイルにアクセスする際に用いるための第 3 アプリケーション・プログラミング・インターフェース（API）を含む。

40

一実施形態のシステムの第 1 デバイスは、第 2 デバイス上に ホストされた ソフトウェアの少なくとも 1 つのプロバイダによってアクセス可能な、プロセッサに基づくデバイスを含む。

#### 【0105】

一実施形態のシステムの第 2 デバイスは、パーソナル・コンピュータ、携帯用計算機、セルラ電話機、携帯用通信デバイス、およびパーソナル・ディジタル・アシスタントの中から選択した、少なくとも 1 つのプロセッサに基づくデバイスを含む。

#### 【0106】

一実施形態のシステムの結合部は、ワイヤレス結合部、ワイヤード結合部、混合ワイヤ

50

レス／ワイヤード結合部、およびローカル・エリア・ネットワーク（LAN）、都市区域ネットワーク（MAN）、ワイド・エリア・ネットワーク（WAN）、企業固有のネットワーク、バックエンド・ネットワーク、インターネットを含む少なくとも1つのネットワークとの結合部、ならびにフロッピ・ディスク、ハード・ディスク・ドライブ、およびCD-ROMディスクを含む着脱可能固定媒体、更に電話回線、バス、および電子メール・メッセージの内少なくとも1つである。

**【0107】**

一実施形態の元のファイルおよび新ファイルは、ダイナミック・リンク・ライブラリ・ファイル、共有オブジェクト・ファイル、埋め込みソフトウェア・コンポーネント（EBS C）、ファームウェア・ファイル、実行可能ファイル、16進データ・ファイルを含むデータ・ファイル、システム・コンフィギュレーション・ファイル、および個人的使用データを含むファイルを含むソフトウェア・ファイルを含む。

10

**【0108】**

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、ホスト・デバイスにおけるデバイス・メモリ管理方法を含み、この方法は、新電子ファイルの少なくとも1つの受信コンポーネントの識別情報を受信するステップであって、新電子ファイルが元の電子ファイルの更新バージョンであり、識別情報が受信したコンポーネントのサイズを含む、ステップと、第1および第2メモリ・エリアの順次探索を行うことによって、受信したコンポーネントを収容することができる十分大きなメモリ・ブロックを識別し予約するステップであって、未使用メモリ・ブロックを排除するように第1メモリ・エリアを書き換え、サイズが第1および第2メモリ・エリアの使用可能メモリ・ブロックのサイズを超過する場合、第1および第2メモリ・エリアを再分配する、ステップと、予約したメモリ・ブロックのアドレスを与えるステップと、受信したコンポーネントを予約メモリ・ブロックに書き込むことによって、元の電子ファイルを更新するステップとを含む。

20

**【0109】**

ホスト・デバイスにおけるデバイス・メモリ管理方法は、更に、受信したコンポーネントを、元の電子ファイルの対応するコンポーネントを含む第1メモリ・エリア内の元のメモリ・ブロック以外のメモリ・ブロックに書き込む場合、第1テーブルを更新するステップを含み、第1テーブルは、ホスト・デバイスの電子ファイルのコンポーネントのコンポーネント情報を含み、このコンポーネント情報は、コンポーネント識別、リード・オンリ・メモリ（ROM）番号、開始アドレス、およびサイズを含む。

30

**【0110】**

一実施形態の方法では、識別および予約するステップは、元の電子ファイルの対応するコンポーネントのサイズよりもサイズが小さい場合、第1メモリ・エリアの元のメモリ・ブロックを予約することを含み、元のメモリ・ブロックは、元の電子ファイルの対応するコンポーネントを含む。

**【0111】**

一実施形態の方法では、識別および予約するステップは、サイズが元の電子ファイルの対応するコンポーネントのサイズを超過する場合、第2メモリ・エリアと第1メモリ・エリアの未使用部分との少なくとも一方を探索して、受信したコンポーネントを格納できる十分大きなメモリ・ブロックを求めることを含む。

40

**【0112】**

一実施形態の方法では、第1メモリ・エリアの書き換えおよび第2メモリ・エリアの再分配のステップは、更に、第1エリアの未使用メモリ・ブロックに対応する少なくとも1つのエントリを含む第2テーブルを読み取り、第1メモリ・エリアのコンポーネントを順次バックし、未使用メモリ・ブロックを統合するために、第1メモリ・エリアのコンポーネントを書き換え、第2メモリ・エリアの各コンポーネントのサイズを順次評価し、第2メモリ・エリアのコンポーネントを、第2メモリ・エリアのコンポーネントのサイズおよび統合した未使用メモリ・ブロックのサイズが許す、第1メモリ・エリアに再書き込みし

50

、順次評価および再書き込みに続いて、統合した未使用メモリ・ブロックの残りのブロックを識別し、統合した未使用メモリ・ブロックの残りのブロックを、第2メモリ・エリアの一部として指定することによって、第2メモリ・エリアを再分配し、再分配した第2メモリ・エリアの使用可能なメモリの開始アドレスを維持するために、第2メモリ・エリアのポインタを設定することを含む。

#### 【0113】

一実施形態の方法では、第2テーブルのエントリは、リード・オンリ・メモリ（ROM）番号、開始アドレス、およびサイズを含む未使用メモリ・ブロックの情報を含む。

前述のホスト・デバイスにおけるデバイス・メモリ管理方法は、更に、ホスト・デバイスの主プログラムからの関数コールを受け、この関数コールが対応する電子ファイルの識別情報を含み、対応する電子ファイルの開始アドレスをベクトル・テーブルから読み出し、開始アドレスおよび識別情報を用いて、対応する電子ファイルに対するコールを発生することによって、ホスト・デバイスの電子ファイルによって与えられる機能にアクセスするステップを含む。

#### 【0114】

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、ファイル更新中におけるデバイス・メモリ管理方法を含み、この方法は、受信コンポーネントの識別情報を用いて、新たな電子ファイルの少なくとも1つの受信コンポーネントのサイズを判定するステップであって、新たな電子ファイルが、元の電子ファイルの更新バージョンである、ステップと、サイズが元の電子ファイルの対応するコンポーネントのサイズ以下である場合、受信コンポーネントを書き込む第1メモリ・エリアの元のメモリ・ブロックを割り当て、元のメモリ・ブロックが元の電子ファイルの対応するコンポーネントを含む、ステップと、サイズが元の電子ファイルの対応するコンポーネントのサイズを超過する場合、第2メモリ・エリアと第1メモリ・エリアの未使用部分とを探索して、受信コンポーネントを格納できる十分に大きなメモリ・ブロックを求めるステップと、サイズが第1および第2メモリ・エリアの使用可能なメモリ・ブロックのサイズを超過する場合、未使用部分を除去するように第1メモリ・エリアを書き換え、第1および第2メモリ・エリアを再分配し、受信コンポーネントを書き込む、第2メモリ・エリアのメモリ・ブロックを割り当てるステップとを含む。

#### 【0115】

一実施形態の方法では、第1メモリ・エリアの書き換えは、更に、第1メモリ・エリアの未使用部分に対応する少なくとも1つのエントリを含むテーブルを読み込み、第1メモリ・エリアのコンポーネントを順次パックし、未使用メモリ・ブロックを統合するために、第1メモリ・エリアのコンポーネントを書き直すことを含む。

#### 【0116】

一実施形態の方法では、第1および第2メモリ・エリアを再分配するステップは、更に、第2メモリ・エリアの各コンポーネントのサイズを順次評価し、第2メモリ・エリアのコンポーネントのサイズおよび書き換えた第2メモリ・エリアの統合した未使用メモリ・ブロックのサイズが許すとき、第2メモリ・エリアのコンポーネントを書き換えた第1メモリ・エリアに書き込み、順次評価および書き込みに続いて、統合した未使用メモリ・ブロックの残りのブロックを特定し、統合した未使用メモリ・ブロックの残りのブロックを第2メモリ・エリアの一部として指定することにより、第1および第2メモリ・エリアを再分配し、再分配した第2メモリ・エリアの使用可能なメモリの開始アドレスを維持するために、第2メモリ・エリアのポインタを設定し、テーブルを評価することを含む。

#### 【0117】

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、電子デバイスのメモリを管理する方法を含み、この方法は、元の電子ファイルの更新バージョンである新電子ファイルの少なくとも1つの新コンポーネントを受信し、第1および第2メモリ・エリアの順次探索を行うことによって、受信したコンポーネントを収容することができる十分大きなメモリ・ブロックを識別し予約し、未使用メモ

リ・ブロックを排除するように第1メモリ・エリアを書き換え、サイズが第1および第2メモリ・エリアの使用可能メモリ・ブロックのサイズを超過する場合、第1および第2メモリ・エリアを再分配し、新コンポーネントを予約したメモリ・ブロックに書き込み、新コンポーネントを、元の電子ファイルの対応するコンポーネントを含む元のメモリ・ブロック以外のメモリ・ブロックに書き込んだときに、ベクトル・テーブルを更新することによって、デバイスの電子ファイルを更新するステップを含む。

#### 【0118】

前述の電子デバイスのメモリ管理方法は、更に、デバイスの主プログラムからの関数コールを受け、関数コールは対応する電子ファイルの識別情報を含み、ベクトル・テーブルから対応する電子ファイルの開始アドレスを読み取り、開始アドレスおよび識別情報を用いて、対応する電子ファイルに対するコールを生成することによって、デバイスの電子ファイルにアクセスするステップを備えている。

10

#### 【0119】

一実施形態の方法では、識別情報は、関数識別、および関数に対する引数を含む。

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、新電子ファイルの少なくとも1つの新コンポーネントを受信する手段であって、新電子ファイルは、元の電子ファイルの更新バージョンである、手段と、第1および第2メモリ・エリアの順次探索を行うことによって、受信したコンポーネントを収容することができる十分大きなメモリ・ブロックを識別し予約する手段であって、未使用メモリ・ブロックを排除するように第1メモリ・エリアを書き換え、サイズが第1および第2メモリ・エリアの使用可能メモリ・ブロックのサイズを超過する場合、第1および第2メモリ・エリアを再分配する、手段と、新コンポーネントを予約したメモリ・ブロックに書き込むことによって、元の電子ファイルを更新する手段と、新コンポーネントを、元の電子ファイルの対応するコンポーネントを含む元のメモリ・ブロック以外のメモリ・ブロックに書き込む場合、ベクトル・テーブルを更新する手段とを備えている装置を含む。

20

#### 【0120】

一実施形態の装置は、パーソナル・コンピュータ、携帯用計算機、セルラ電話機、携帯用通信デバイス、およびパーソナル・ディジタル・アシスタントの中から選択した、少なくとも1つのプロセッサに基づくデバイスを含む、装置。

#### 【0121】

一実施形態の装置は、第1アプリケーション・プログラミング・インターフェース（API）を、メモリ・ブロックを識別し予約する手段として含む。

30

一実施形態の装置は、更に、デバイスの主プログラムから関数コールを受け取る手段であって、関数コールは、対応する電子ファイルの識別情報を含む、手段と、ベクトル・テーブルから対応する電子ファイルの開始アドレスを読み取る手段と、開始アドレスおよび識別情報を用いて、対応する電子ファイルに対するコールを発生する手段とを備えている。

#### 【0122】

一実施形態の装置は、第2アプリケーション・プログラミング・インターフェース（API）を、関数コールを受け取る手段として含む。

40

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、実行可能な命令を含むコンピュータ読み取り可能媒体を含み、処理システムにおいて実行すると、新電子ファイルの少なくとも1つの受信コンポーネントの識別情報を受信し、新電子ファイルは、元の電子ファイルの更新バージョンであり、識別情報は受信したコンポーネントのサイズを含み、第1および第2メモリ・エリアの順次探索を行うことによって、受信したコンポーネントを収容することができる十分大きなメモリ・ブロックを識別し予約し、未使用メモリ・ブロックを排除するように第1メモリ・エリアを書き換え、サイズが第1および第2メモリ・エリアの使用可能メモリ・ブロックのサイズを超過する場合、第1および第2メモリ・エリアを再分配し、予約したメモリ・ブロックのアドレスを与え、受信したコンポーネントを予約メモリ・ブロックに書き込むことによ

50

って、元の電子ファイルを更新することによって電子ファイルおよびファイル・コンポーネントを更新する。

【0123】

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、実行可能な命令を含む電磁媒体を含み、処理システムにおいて実行すると、新電子ファイルの少なくとも1つの受信コンポーネントの識別情報を受信し、新電子ファイルは元の電子ファイルの更新バージョンであり、識別情報は受信したコンポーネントのサイズを含み、第1および第2メモリ・エリアの順次探索を行うことによって、受信したコンポーネントを収容することができる十分大きなメモリ・ブロックを識別し予約し、未使用メモリ・ブロックを排除するように第1メモリ・エリアを書き換え、サイズが第1および第2メモリ・エリアの使用可能メモリ・ブロックのサイズを超過する場合、第1および第2メモリ・エリアを再分配し、予約したメモリ・ブロックのアドレスを与え、受信したコンポーネントを予約メモリ・ブロックに書き込むことによって、元の電子ファイルを更新する、ことによって電子ファイルおよびファイル・コンポーネントを更新する。

10

【0124】

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、電子ファイルの更新システムを含み、このシステムは、デルタ・ファイル生成する、ファイル差別化および更新システムの第1コンポーネントを含む第1デバイスと、第1デバイスから第1デルタ・ファイルを少なくとも1つの結合部を介して受け取る第2デバイスとを備えている。一実施形態の第2デバイスは、デルタ・ファイルを通じて新電子ファイルの少なくとも1つのコンポーネントを受信し、新電子ファイルは、元の電子ファイルの更新バージョンであり、受信したコンポーネントのサイズを判定し、サイズが元の電子ファイルの対応するコンポーネントのサイズ以下である場合、受信したコンポーネントを、元の電子ファイルの対応するコンポーネントを含む元のメモリ・エリアに書き込み、サイズが元の電子ファイルの対応するコンポーネントのサイズを超過する場合、受信したコンポーネントを、元のメモリ・エリアと関連付けられた予約メモリ・エリアに書き込むことにより第2デバイスの電子ファイルを更新するように構成されているファイル差別化および更新システムの第2コンポーネントを含む。

20

【0125】

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、電子ファイルの更新方法を含み、この方法は、新電子ファイルの少なくとも1つのコンポーネントを受信するステップであって、新電子ファイルは、元の電子ファイルの更新バージョンである、ステップと、受信したコンポーネントのサイズを判定するステップと、サイズが元の電子ファイルの対応するコンポーネントのサイズ以下である場合、受信したコンポーネントを、元の電子ファイルの対応するコンポーネントを含む元のメモリ・エリアに書き込むステップと、サイズが元の電子ファイルの対応するコンポーネントのサイズを超過する場合、受信したコンポーネントを、元のメモリ・エリアと関連付けられた予約メモリ・エリアに書き込むステップとを含む。一実施形態の方法では、少なくとも1つの成分は、コンポーネントのグループを含む。

30

【0126】

前述の電子ファイルの更新中におけるデバイス・メモリ管理のためのシステムおよび関連する方法は、新電子ファイルの少なくとも1つのコンポーネントを受信する手段であって、新電子ファイルは、元の電子ファイルの更新バージョンである、手段と、受信したコンポーネントのサイズを判定する手段と、サイズが元の電子ファイルの対応するコンポーネントのサイズ以下である場合、受信したコンポーネントを、元の電子ファイルの対応するコンポーネントを含む元のメモリ・エリアに書き込む手段と、サイズが元の電子ファイルの対応するコンポーネントのサイズを超過する場合、受信したコンポーネントを、元のメモリ・エリアと関連付けられた予約メモリ・エリアに書き込む手段とを備えている装置を含む。

40

【0127】

50

本発明の態様は、フィールド・プログラマブル・ゲート・アレイ（FPGA）、プログラマブル・アレイ・ロジック（PAL）デバイス、電氣的プログラム可能ロジックおよびメモリ・デバイスのようなプログラマブル・ロジック・デバイス（PDL）、標準的なセルに基づくデバイス、ならびに特定用途集積回路（ASIC）を含む種々の回路のいずれにもプログラムされる機能として実現することができる。更に、本発明の態様は、ソフトウェアに基づく回路エミュレーション、ディスクリート・ロジック（連続および組み合わせ）、カスタム・デバイス、ファジイ（ニューラル）デバイス、量子デバイス、および上述のデバイスの種類のいずれのハイブリッドを有するマイクロプロセッサにおいても具体化することができる。勿論、基盤となるデバイス技術は、種々のコンポーネントの種類、例えば、相補金属酸化物半導体（CMOS）のような金属酸化物半導体電界効果トランジスタ（MOSFET）技術、エミッタ結合ロジック（ECL）のようなバイポーラ技術、ポリマ技術（例えば、シリコン共役ポリマおよび金属共役ポリマー金属構造）、混合アナログおよびデジタルなどにおいて提供することができる。

10

#### 【0128】

文脈上特に必要であることが明らかでない限り、明細書および特許請求の範囲を通じて、「備える」、「備えている」等の言葉は、排他的または網羅的意味ではなく、包含の意味で解釈するものとする。即ち、「含むが、限定されない」という意味である。単数または複数を用いる単語も、それぞれ、複数および単数を含むものとする。加えて、「ここでは」、「以下で」という単語、および同様な意義の単語は、本願において用いられる場合、本願全体を指すのであり、本願のいずれの特定部分を指すのではないとする。

20

#### 【0129】

本発明の例示した実施例についてのこれまでの説明は、それで全てであることも、開示した正確な形態に本発明を限定することも意図していない。本発明の具体的な実施形態および例を例示の目的のためにここで記載したが、種々の等価な変更も本発明の範囲内で可能であることは、当業者は認めよう。ここに提示した本発明の教示は、他の処理システムや通信システムにも適用することができ、前述のFファイル更新のみに適用できるのではない。

#### 【0130】

前述した種々の実施形態のエレメントおよび作用を組み合わせれば、更に別の実施形態も得ることができる。これらおよびその他の変更は、前述の詳細な説明に鑑みれば、容易に行うことができる。

30

#### 【0131】

先の引例は全て、ここで引用したことにより、本願にも含まれるものとする。必要であれば、本発明の態様を変更して前述の種々の特許および出願のシステム、機能および概念を採用すれば、本発明の更に別の実施形態を得ることができる。

#### 【0132】

一般に、特許請求の範囲においては、用いられる用語は、明細書に開示される具体的な実施形態および特許請求の範囲に本発明を限定するように解釈するのではなく、特許請求の範囲の下で動作してファイル差別化および更新方法を実施するあらゆる処理システムを含むように解釈するのが当然である。したがって、本発明は、本開示に限定されるのではなく、むしろ発明の範囲は特許請求の範囲によって全体的に決定されるものとする。

40

#### 【0133】

本発明のある種の態様をある請求項の形式で提示するが、本願の発明者は、本発明の種々の態様をいかなる数の請求項形式でも想定している。例えば、本発明の一態様のみを、コンピュータ読み取り可能媒体に具体化するものとして明記するが、他の態様も同様にコンピュータ読み取り可能媒体に具体化することができる。したがって、本発明は、本願を出願した後でも、更に別の請求項を追加し、本発明の別の態様に対する、このような更に別の請求項を追求する権利を保有するものとする。

#### 【図面の簡単な説明】

#### 【0134】

50

【図 1】図 1 は、一実施形態の下における、バイト・レベルのファイル差別化および更新（F D U）アルゴリズムを用いて、電子ファイルを更新するシステムのブロック図である。

【図 2】図 2 は、一実施形態の下における、ソフトウェア更新システムのブロック図である。

【図 3】図 3 は、一実施形態の下における、デバイス・メモリのブロック図である。

【図 4】図 4 は、一実施形態の下における、非クリティカル・コンポーネント更新のフロー図である。

【図 5】図 5 は、一実施形態の下におけるクリティカル・コンポーネントの更新のフロー図である。

10

【図 6】図 6 は、一実施形態の下において、更新可能な E B S C 毎に予約メモリ割り当てを用いた静的アドレッシングのフロー図である。

【図 7】図 7 は、図 6 の実施形態の下において、静的アドレッシング・メモリ管理をサポートするように構成したホスト・デバイス・メモリの一部のブロック図である。

【図 8】図 8 は、一実施形態の下において、更新可能な E B S C のグループ単位での予約メモリ割り当てを用いた静的アドレッシングのフロー図である。

【図 9】図 9 は、図 8 の実施形態の下において、静的アドレッシング・メモリ管理をサポートするように構成したホスト・デバイス・メモリの一部のブロック図である。

【図 10】図 10 は、一実施形態の下において、静的アドレッシングをサポートするクライアント・デバイス R O M マップのブロック図である。

20

【図 11】図 11 は、一実施形態の下において、静的アドレッシングを用いた埋め込みソフトウェアの開発および展開プロセスのブロック図である。

【図 12】図 12 は、一実施形態の下において、動的アドレッシングを用いて E B S C または E B S C グループを更新するためのフロー図である。

【図 13】図 13 は、一実施形態の下において、動的アドレッシングを用いた関数コールのフロー図である。

【図 14】図 14 は、一実施形態の下において、動的アドレッシングをサポートするクライアント・デバイスの R O M メモリ・マップのブロック図である。

【図 15】図 15 は、図 3 および図 14 の実施形態の下において、アップグレード・クライアント、埋め込みソフトウェア・エリア、DMM ライブラリ、ベクトル・テーブル、およびガベージ・テーブルを含むメモリのコンポーネント間の相互作用の一例を示すホスト・デバイス・メモリのブロック図である。

30

【図 16】図 16 は、一実施形態の下において、予約 A P I を用いて E B S C の新バージョンを収容するためにメモリ・エリアを予約するためのフロー図である。

【図 17】図 17 は、一実施形態の下において、予約 A P I のルールを用いて、E B S C の新バージョンを収容するためにメモリ・エリアを突き止めるためのフロー図である。

【図 18】図 18 は、一実施形態の下における、埋め込みソフトウェア・エリア（ガベージ・コレクション）の未使用エリアの収集のためのソフトウェアである。

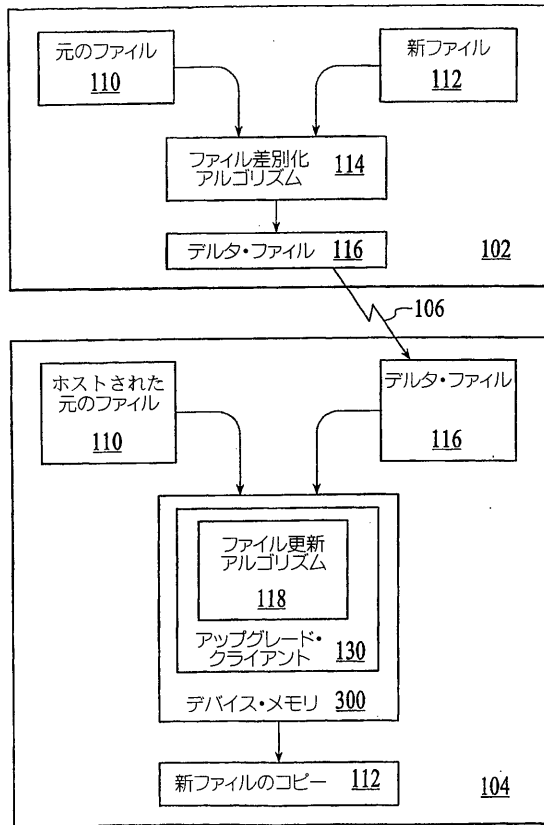
【図 19】図 19 は、一実施形態の下における、予約 A P I の更新処理の前後でのホスト・デバイス・メモリの一例のブロック図である。

40

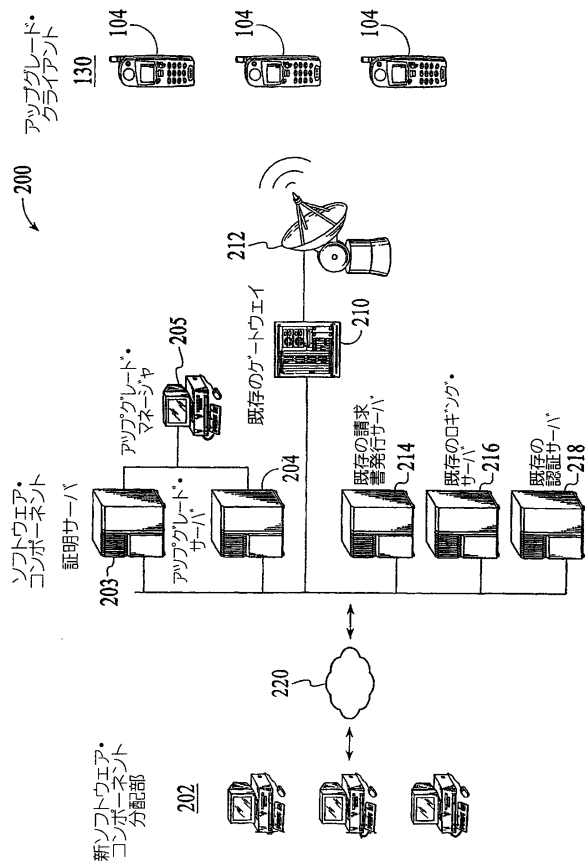
【図 20】図 20 は、一実施形態の下において、アップグレードおよびガベージ・コレクション処理の後におけるホスト・デバイスのブロック図である。

【図 21】図 21 は、一実施形態の下において、動的アドレッシングを用いた埋め込みソフトウェアの開発および展開プロセスのブロック図である。

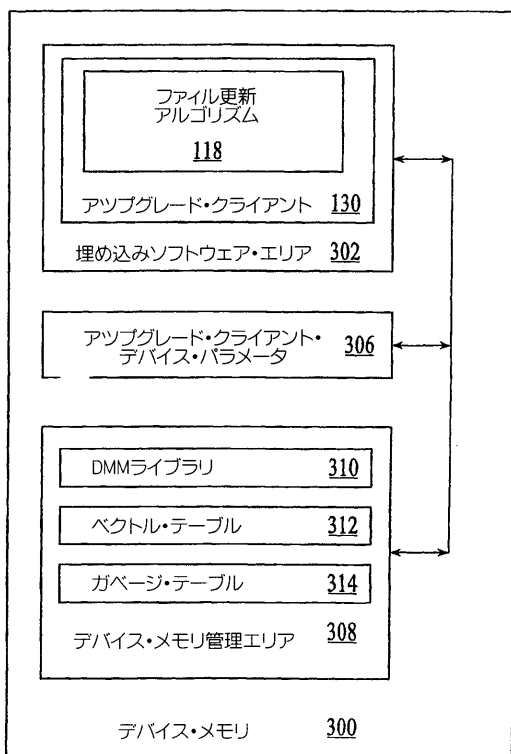
【図 1】



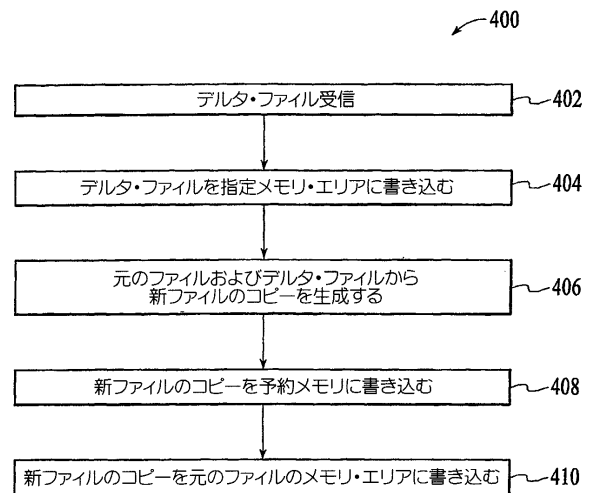
【図 2】



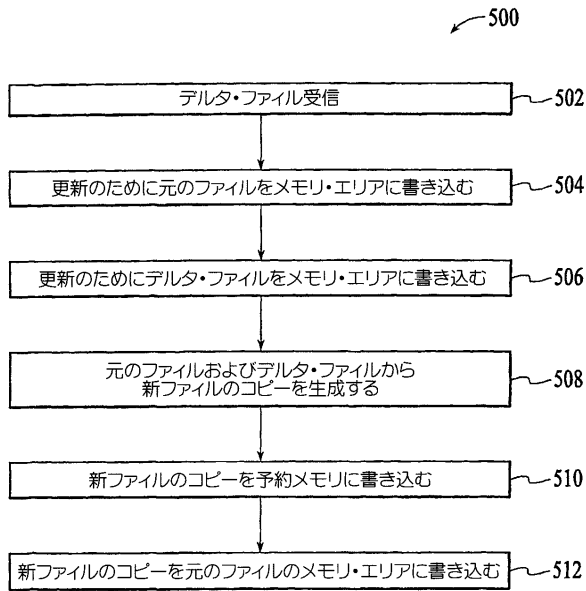
【図 3】



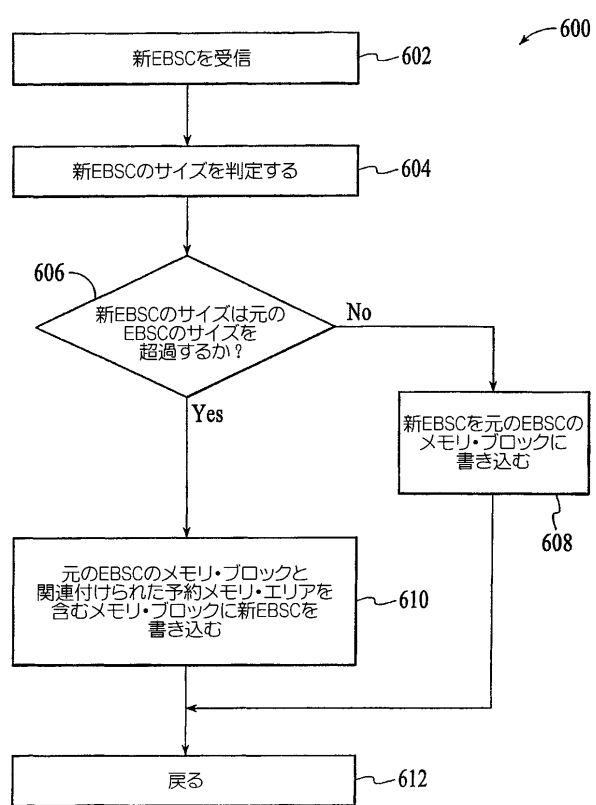
【図 4】



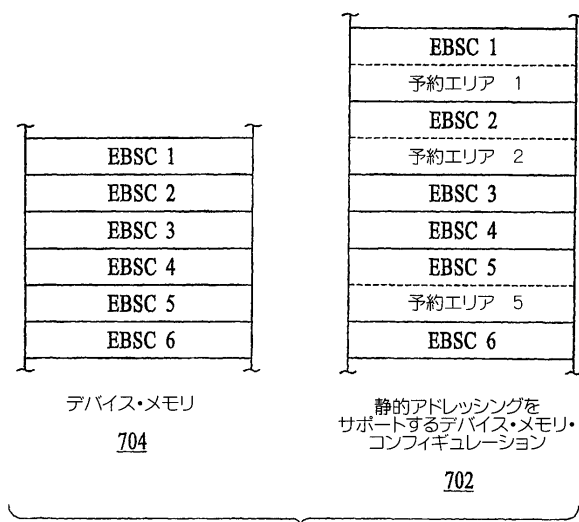
【図 5】



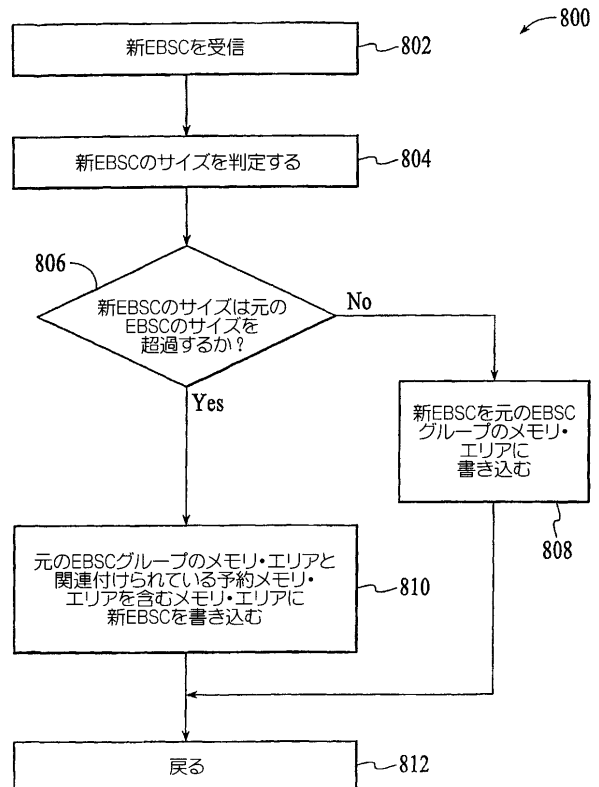
【図 6】



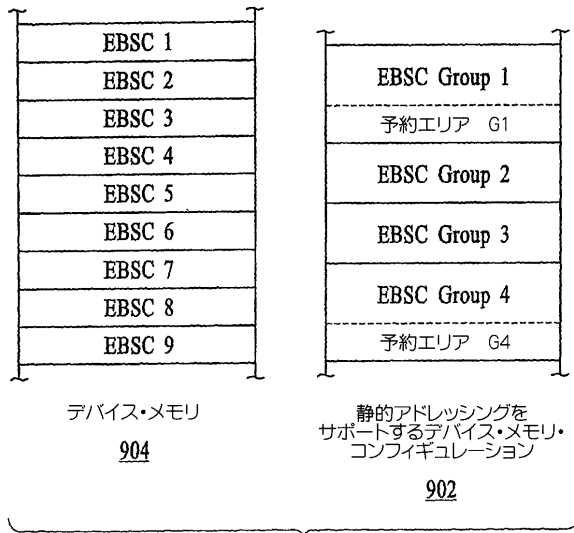
【図 7】



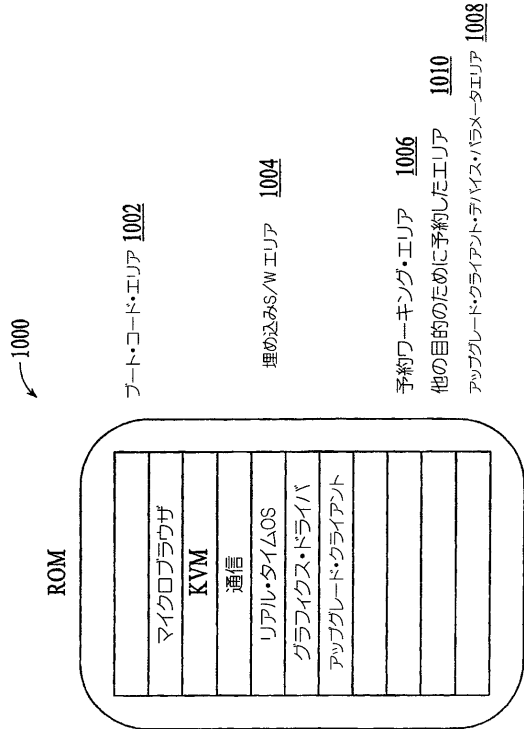
【図 8】



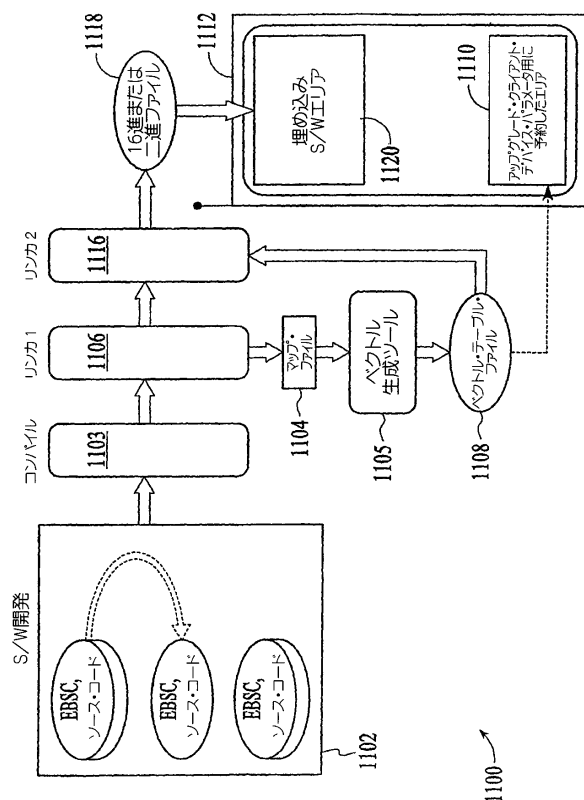
【図 9】



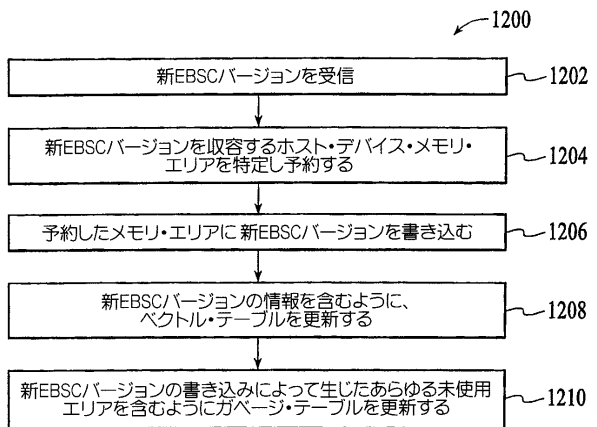
【図 10】



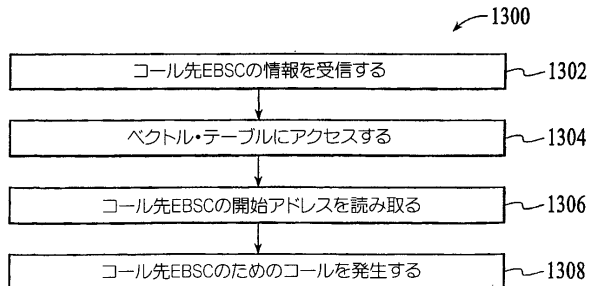
【図 11】



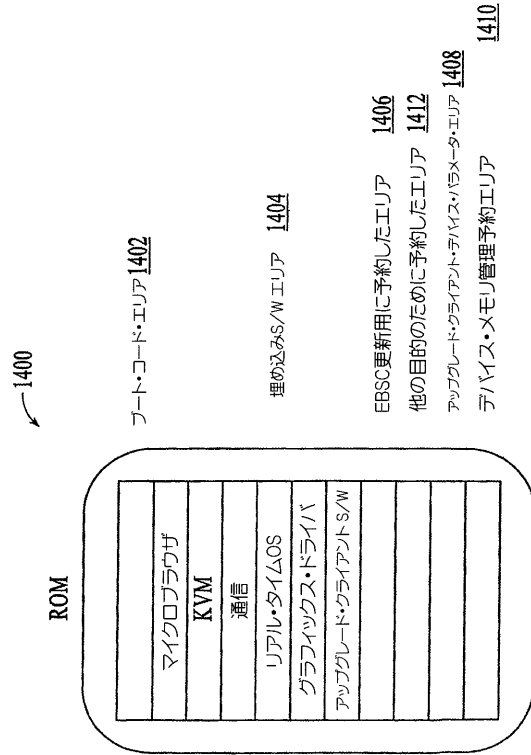
【図 12】



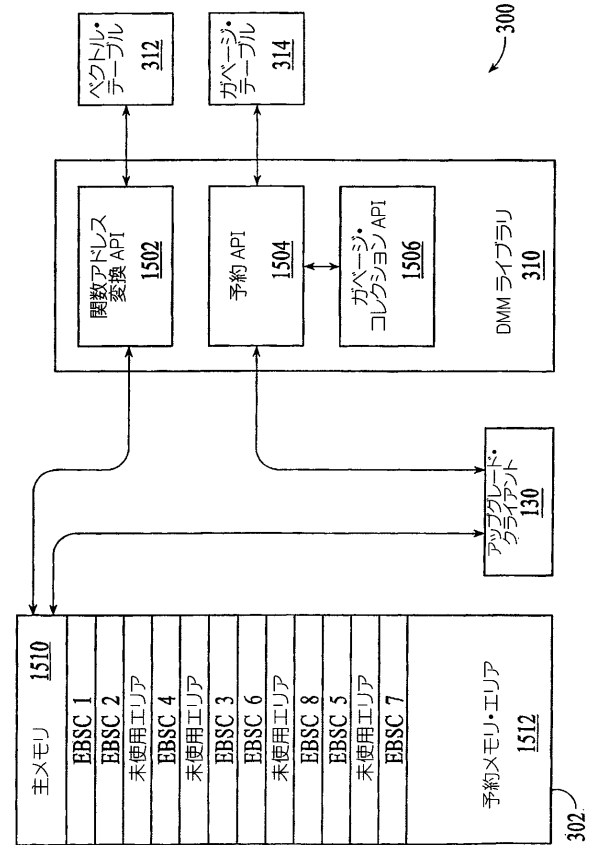
【図 13】



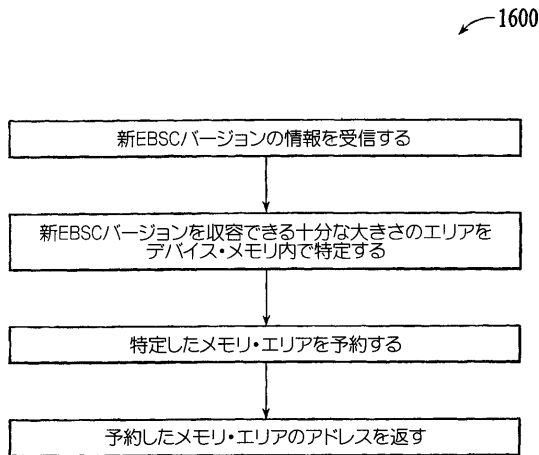
【図 14】



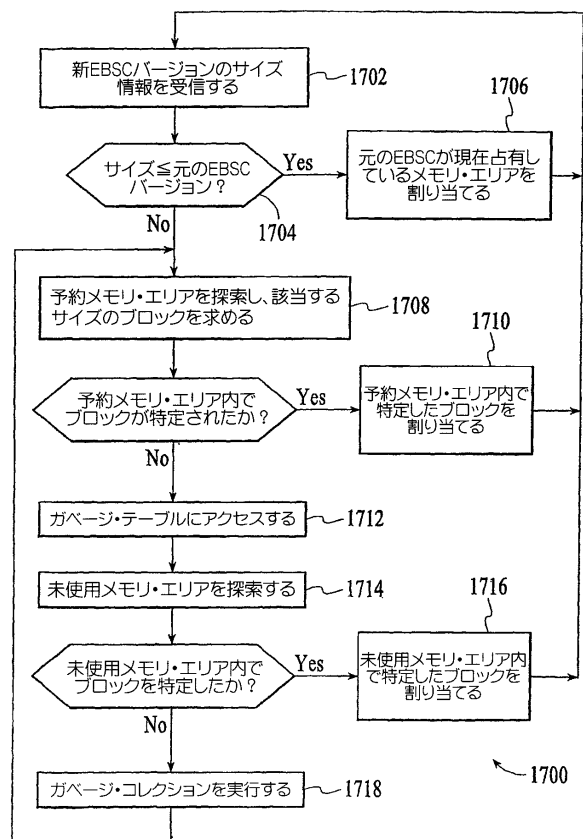
【図 15】



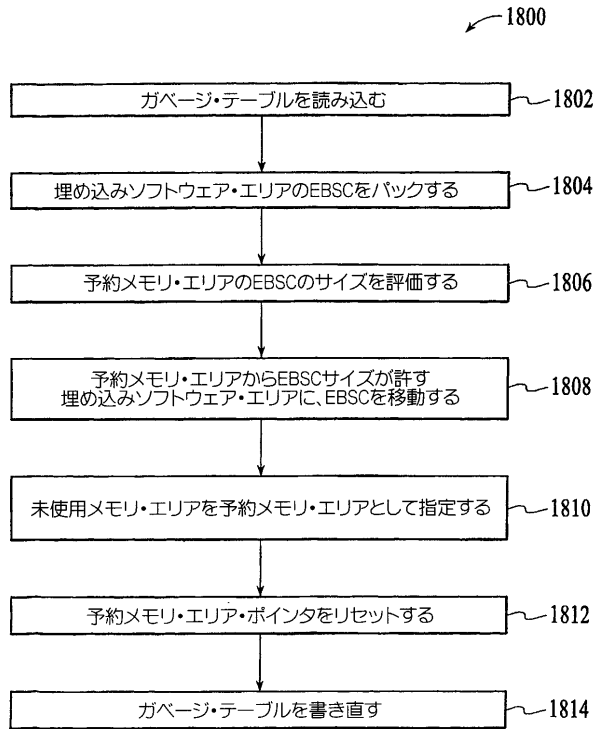
【図 16】



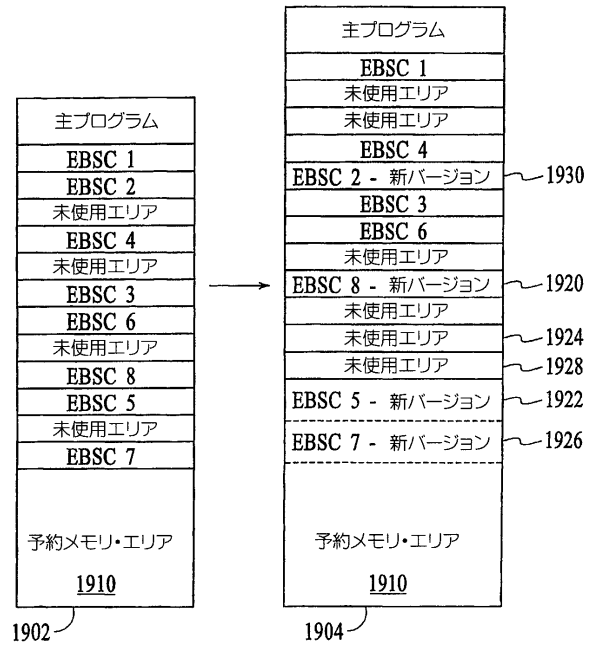
【図 17】



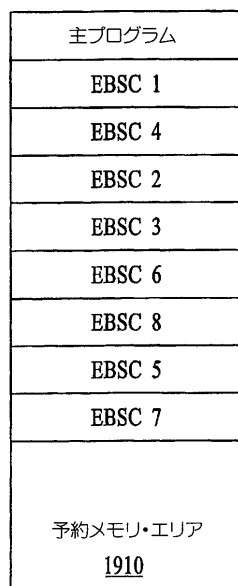
【図 18】



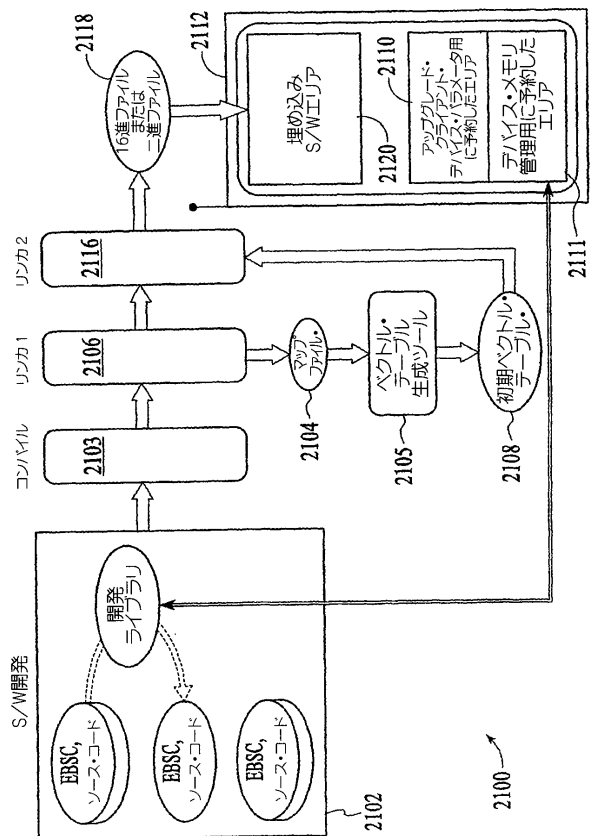
【図 19】



【図 20】



【図 21】



---

フロントページの続き

(74)代理人 100096013

弁理士 富田 博行

(74)代理人 100120112

弁理士 中西 基晴

(72)発明者 ペン, ルオシェン

アメリカ合衆国カリフォルニア州95002, アルビソ, ゴールド・ストリート 2130, スウィート 200

審査官 工藤 嘉晃

(56)参考文献 特開2002-259172 (JP, A)

特開平04-165541 (JP, A)

(58)調査した分野(Int.Cl., DB名)

G06F 12/00

JSTPlus(JDreamII)