(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第4489483号
(P4489483)
(45)発行日　平成22年6月23日(2010.6.23)　(24)登録日　平成22年4月9日(2010.4.9)

(51)Int.Cl.　F I
G06F 12/00 (2006.01)　G06F 12/00 547Q
G06F 12/00 546T

請求項の数 33　(全 24 頁)

(21)出願番号　特願2004-93771(P2004-93771)
(22)出願日　平成16年3月26日(2004.3.26)
(65)公開番号　特開2004-295897(P2004-295897A)
(43)公開日　平成16年10月21日(2004.10.21)
審査請求日　平成19年3月20日(2007.3.20)
(31)優先権主張番号　10/401,244
(32)優先日　平成15年3月26日(2003.3.26)
(33)優先権主張国　米国(US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100077481
弁理士　谷　義一
(74)代理人　100088915
弁理士　阿部　和夫
(72)発明者　ステファン　エイチ．ファリーズ
アメリカ合衆国　９８１０２　ワシントン州　シアトル　イースト　ロイ　ストリート　５２５　アパートメント　３０１

最終頁に続く

(54)【発明の名称】初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変形する方法

(57)【特許請求の範囲】
【請求項１】
様々なタイプのデータオブジェクトを直列化および非直列化することのできるコンピュータ実行可能命令を有する拡張可能な直列化エンジンを含むコンピュータにおいて、前記コンピュータ実行可能命令を前記コンピュータが実行することによって実施される初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変形する方法であって、前記直列化エンジンの実装を置き換える必要なしに前記直列化エンジンのランタイム動作を１つまたは複数の拡張ルーチンによって改変することを可能にし、前記方法は、
前記直列化エンジンによるランタイム処理のために初期タイプの初期オブジェクトを受け取る動作と、
前記初期オブジェクトの前記初期タイプに関するタイプ情報を受け取る動作と、
前記タイプ情報に基づいて、ランタイム修正に適した前記初期オブジェクトの中間表現を生成する動作と、
１つまたは複数のカスタム拡張ルーチンを呼び出して、前記初期オブジェクトの修正済み中間表現へ前記初期オブジェクトの前記中間表現を変更し、それにより前記直列化エンジンの前記ランタイム動作を改変する動作と、
前記初期オブジェクトの前記修正済み中間表現から最終タイプの最終オブジェクトを生成する動作であって、前記最終オブジェクト及び前記初期オブジェクトの少なくとも１つはメモリ内オブジェクトであり、前記最終オブジェクトは、前記初期オブジェクトに基づいてインスタンス化およびポピュレートされることになるメモリ内オブジェクトを含み、

前記直列化エンジンは、前記初期オブジェクトを非直列化して前記最終オブジェクトを生成し、前記初期オブジェクトはメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを直列化して前記最終オブジェクトを生成する、動作とを備え、

前記中間表現を変更するために呼び出される前記１つまたは複数のカスタム拡張ルーチンは、前記中間表現を実際には変更せずに、前記中間表現をどのように変更するかを指定し、それにより前記中間表現の変更は、バッファリング要件を削減するために前記最終オブジェクトの生成まで延期される

ことを特徴とする方法。

【請求項２】

前記中間表現は、前記初期オブジェクトについての全体的なタイプ、および前記初期オブジェクト内に含まれる１つまたは複数のオブジェクトについてのオブジェクト名、オブジェクトタイプ、オブジェクトデータを含む

ことを特徴とする請求項１に記載の方法。

【請求項３】

前記全体的なオブジェクトタイプ、および前記初期オブジェクト内に含まれる前記１つまたは複数のオブジェクトについての前記オブジェクト名、前記オブジェクトタイプ、前記オブジェクトデータのうちの、少なくとも１つを変更する動作

をさらに備えることを特徴とする請求項２に記載の方法。

【請求項４】

１つまたは複数の標準ルーチンを呼び出して、前記初期オブジェクトの前記中間表現を修正する動作

をさらに備えることを特徴とする請求項１に記載の方法。

【請求項５】

前記初期オブジェクトの前記中間表現の修正は、前記タイプ情報内の特定パターンに基づく

ことを特徴とする請求項１に記載の方法。

【請求項６】

前記初期オブジェクトの前記中間表現の修正は、前記初期オブジェクト内のオブジェクトデータに基づく

ことを特徴とする請求項１に記載の方法。

【請求項７】

前記直列化エンジンは、１つまたは複数のメッセージを送受信する分散アプリケーション用のメッセージングシステムの一部であり、前記初期オブジェクトおよび前記最終オブジェクトは、メッセージの少なくとも一部を表す

ことを特徴とする請求項１に記載の方法。

【請求項８】

様々なタイプのデータオブジェクトを直列化および非直列化することのできるコンピュータ実行可能命令を有する拡張可能な直列化エンジンを含むコンピュータに、初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変形する方法を実行させるためのコンピュータ実行可能命令を格納したコンピュータ読み取り可能な記録媒体であって、前記方法は、前記直列化エンジンの実装を置き換える必要なしに前記直列化エンジンのランタイム動作を１つまたは複数の拡張ルーチンによって改変することを可能にし、前記方法は、

前記直列化エンジンによるランタイム処理のために初期タイプの初期オブジェクトを受け取る動作と、

前記初期オブジェクトの前記初期タイプに関するタイプ情報を受け取る動作と、

前記タイプ情報に基づいて、ランタイム修正に適した前記初期オブジェクトの中間表現を生成する動作と、

１つまたは複数のカスタム拡張ルーチンを呼び出して、前記初期オブジェクトの修正済み中間表現へ前記初期オブジェクトの前記中間表現を変更し、それにより前記直列化エン

ジンの前記ランタイム動作を改変する動作と、
前記初期オブジェクトの前記修正済み中間表現から最終タイプの最終オブジェクトを生成する動作であって、前記最終オブジェクト及び前記初期オブジェクトの少なくとも１つはメモリ内オブジェクトであり、前記最終オブジェクトは、前記初期オブジェクトに基づいてインスタンス化およびポピュレートされることになるメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを非直列化して前記最終オブジェクトを生成し、前記初期オブジェクトはメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを直列化して前記最終オブジェクトを生成する、動作とを備え、
前記中間表現を変更するために呼び出される前記１つまたは複数のカスタム拡張ルーチンは、前記中間表現を実際には変更せずに、前記中間表現をどのように変更するかを指定し、それにより前記中間表現の変更は、バッファリング要件を削減するために前記最終オブジェクトの生成まで延期される
ことを特徴とするコンピュータ読み取り可能な記録媒体。
【請求項９】
前記中間表現は、前記初期オブジェクトについての全体的なタイプ、および前記初期オブジェクト内に含まれる１つまたは複数のオブジェクトについてのオブジェクト名、オブジェクトタイプ、オブジェクトデータを含む
ことを特徴とする請求項８に記載のコンピュータ読み取り可能な記録媒体。
【請求項１０】
前記方法は、前記全体的なオブジェクトタイプ、および前記初期オブジェクト内に含まれる前記１つまたは複数のオブジェクトについての前記オブジェクト名、前記オブジェクトタイプ、前記オブジェクトデータのうちの、少なくとも１つを変更する動作をさらに備えることを特徴とする請求項９に記載のコンピュータ読み取り可能な記録媒体。
【請求項１１】
前記方法は、１つまたは複数の置換可能な標準ルーチンを呼び出して、前記初期オブジェクトの前記中間表現を修正する動作をさらに備えることを特徴とする請求項８に記載のコンピュータ読み取り可能な記録媒体。
【請求項１２】
前記初期オブジェクトの前記中間表現の修正は、前記タイプ情報内の特定パターン、メタデータ、および前記初期オブジェクト内のオブジェクトデータのうちの少なくとも１つに基づく
ことを特徴とする８に記載のコンピュータ読み取り可能な記録媒体。
徴とする請求項８に記載のコンピュータ読み取り可能な記録媒体。
【請求項１３】
前記初期オブジェクトと前記最終オブジェクトは両方ともメモリ内オブジェクトである
ことを特徴とする請求項８に記載のコンピュータ読み取り可能な記録媒体。
【請求項１４】
様々なタイプのデータオブジェクトを直列化および非直列化するコンピュータ実行可能命令を有する拡張可能な直列化エンジンを含むコンピュータにおいて、前記コンピュータ実行可能命令を前記コンピュータが実行することによって実施される初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変換する方法であって、前記直列化エンジンの１つまたは複数の他の既存ルーチンを置き換える必要なしに前記直列化エンジンの動作を１つまたは複数の拡張ルーチンによってランタイムに改変することを可能にし、前記方法は、
前記直列化エンジンによるランタイム処理のために受け取った初期タイプの初期オブジェクトに関するタイプ情報を識別するステップと、
前記タイプ情報に基づいて、前記初期オブジェクトを、ランタイム修正に適した前記初期オブジェクトの中間表現に変換するステップと、
前記初期オブジェクトの修正済み中間表現を生成するように前記初期オブジェクトの前記中間表現を１つまたは複数の拡張ルーチンに従って修正し、それにより前記直列化エン

ジンのランタイム動作を改変するステップと、
前記初期オブジェクトの前記修正済み中間表現を最終タイプの最終オブジェクトに変換するステップであって、前記最終オブジェクト及び前記初期オブジェクトの少なくとも１つはメモリ内オブジェクトであり、前記最終オブジェクトは、前記初期オブジェクトに基づいてインスタンス化およびポピュレートされることになるメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを非直列化して前記最終オブジェクトを生成し、前記初期オブジェクトはメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを直列化して前記最終オブジェクトを生成する、ステップと、
前記中間表現によるバッファリング修正を回避するために、前記中間表現が前記最終オブジェクトに変換されるまでは前記中間表現の修正を延期するステップと
を備えることを特徴とする方法。
【請求項１５】
前記初期オブジェクトの前記中間表現は、オブジェクト名、オブジェクトタイプ、オブジェクトデータのうちの少なくとも１つを含む
ことを特徴とする請求項１４に記載の方法。
【請求項１６】
前記初期オブジェクトの前記中間表現を１つまたは複数の拡張ルーチンに従って修正するステップは、前記オブジェクト名、前記オブジェクトタイプ、前記オブジェクトデータのうちの少なくとも１つを修正することを含む
ことを特徴とする請求項１５に記載の方法。
【請求項１７】
前記初期オブジェクトの前記中間表現を、前記直列化エンジン内の１つまたは複数の標準ルーチンに従って修正するステップ
をさらに備えることを特徴とする請求項１４に記載の方法。
【請求項１８】
前記初期オブジェクトの前記中間表現の修正は、前記タイプ情報内の特定パターンと、前記初期オブジェクト内のオブジェクトデータとのうちのどちらか、または両方に基づく
ことを特徴とする請求項１４に記載の方法。
【請求項１９】
前記初期オブジェクトと前記最終オブジェクトは両方ともメモリ内オブジェクトである
ことを特徴とする請求項１４に記載の方法。
【請求項２０】
様々なタイプのデータオブジェクトを直列化および非直列化するコンピュータ実行可能命令を有する拡張可能な直列化エンジンを含むコンピュータに、初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変換する方法を実行させるためのコンピュータ実行可能命令を格納したコンピュータ読み取り可能な記録媒体であって、前記方法は、前記直列化エンジンの１つまたは複数の他の既存ルーチンを置き換える必要なしに前記直列化エンジンの動作を１つまたは複数の拡張ルーチンによってランタイムに改変することを可能にし、前記方法は、
前記直列化エンジンによるランタイム処理のために受け取った初期タイプの初期オブジェクトに関するタイプ情報を識別するステップと、
前記タイプ情報に基づいて、前記初期オブジェクトを、ランタイム修正に適した前記初期オブジェクトの中間表現に変換するステップと、
前記初期オブジェクトの修正済み中間表現を生成するように前記初期オブジェクトの前記中間表現を１つまたは複数の拡張ルーチンに従って修正し、それにより前記直列化エンジンのランタイム動作を改変するステップと、
前記初期オブジェクトの前記修正済み中間表現を最終タイプの最終オブジェクトに変換するステップであって、前記最終オブジェクト及び前記初期オブジェクトの少なくとも１つはメモリ内オブジェクトであり、前記最終オブジェクトは、前記初期オブジェクトに基づいてインスタンス化およびポピュレートされることになるメモリ内オブジェクトを含み

、前記直列化エンジンは、前記初期オブジェクトを非直列化して前記最終オブジェクトを生成し、前記初期オブジェクトはメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを直列化して前記最終オブジェクトを生成する、ステップと、

前記中間表現を前記最終オブジェクトに変換するまでは前記中間表現の修正を延期するステップと

を備えることを特徴とするコンピュータ読み取り可能な記録媒体。

【請求項２１】

前記初期オブジェクトの前記中間表現は、オブジェクト名、オブジェクトタイプ、オブジェクトデータのうちの少なくとも１つを含む

ことを特徴とする請求項２０に記載のコンピュータ読み取り可能な記録媒体。

【請求項２２】

前記初期オブジェクトの前記中間表現を１つまたは複数の拡張ルーチンに従って修正するステップは、前記オブジェクト名、前記オブジェクトタイプ、前記オブジェクトデータのうちの少なくとも１つを修正することを含む

ことを特徴とする請求項２０に記載のコンピュータ読み取り可能な記録媒体。

【請求項２３】

前記方法は、前記初期オブジェクトの前記中間表現を１つまたは複数の標準ルーチンに従って修正するステップをさらに備える

ことを特徴とする請求項２０に記載のコンピュータ読み取り可能な記録媒体。

【請求項２４】

前記初期オブジェクトの前記中間表現の修正は、前記タイプ情報内の特定パターンと、前記初期オブジェクト内のオブジェクトデータとのうちのどちらか、または両方に基づく

ことを特徴とする請求項２０に記載のコンピュータ読み取り可能な記録媒体。

【請求項２５】

前記初期オブジェクトと前記最終オブジェクトは両方ともメモリ内オブジェクトである

ことを特徴とする請求項２０に記載のコンピュータ読み取り可能な記録媒体。

【請求項２６】

様々なタイプのデータオブジェクトを直列化および非直列化するコンピュータ実行可能命令を有する拡張可能な直列化エンジンを含むコンピュータに、１つまたは複数の初期タイプである１つまたは複数の初期オブジェクトを、１つまたは複数の最終タイプである１つまたは複数の最終オブジェクトに変形する方法を実行させるためのコンピュータ実行可能命令を有するコンピュータ実行可能モジュールを格納したコンピュータ読み取り可能な記録媒体であって、前記直列化エンジンのランタイム動作は、前記直列化エンジンの既存部分を再実装する必要なしに改変することができ、前記モジュールは、

前記直列化エンジンによるランタイム処理のために受け取った初期タイプの初期オブジェクトに関するタイプ情報を識別するためのランタイム置換可能な反映モジュールと、

識別されたタイプ情報に基づいて前記初期オブジェクトの中間表現を生成および修正するための１つまたは複数のランタイム置換可能な変換モジュールであって、前記直列化エンジンの前記ランタイム動作を改変する１つまたは複数の拡張ルーチンを含む１つまたは複数のランタイム置換可能な変換モジュールと、

前記変換モジュールによって生成された修正済み中間表現から最終タイプの最終オブジェクトを生み出すためのランタイム置換可能な生成モジュールであって、前記最終オブジェクト及び前記初期オブジェクトの少なくとも１つはメモリ内オブジェクトであり、前記最終オブジェクトは、前記初期オブジェクトに基づいてインスタンス化およびポピュレートされることになるメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを非直列化して前記最終オブジェクトを生成し、前記初期オブジェクトはメモリ内オブジェクトを含み、前記直列化エンジンは、前記初期オブジェクトを直列化して前記最終オブジェクトを生成する、生成モジュールとを備え、

前記１つまたは複数のランタイム置換可能な変換モジュールは、前記初期オブジェクトの前記中間表現に１つまたは複数の修正を加えることに関連するバッファ要件を回避する

ために、前記中間表現が前記最終オブジェクトに変換されるまでは、前記中間表現に対する前記１つまたは複数の修正を延期することができる
ことを特徴とするコンピュータ読み取り可能な記録媒体。
【請求項２７】
前記中間表現は、前記初期オブジェクトと前記初期オブジェクト内に含まれる任意のオブジェクトとに関するオブジェクト名、オブジェクトタイプ、オブジェクトデータを含む
ことを特徴とする請求項２６に記載のコンピュータ読み取り可能な記録媒体。
【請求項２８】
前記１つまたは複数のランタイム置換可能な変換モジュールは、前記初期オブジェクトと前記初期オブジェクト内に含まれるオブジェクトとに関するオブジェクト名、オブジェクトタイプ、オブジェクトデータのうちの、少なくとも１つを変更することができる
ことを特徴とする請求項２６に記載のコンピュータ読み取り可能な記録媒体。
【請求項２９】
前記１つまたは複数のランタイム置換可能な変換モジュールは、前記中間表現を修正することなく前記初期オブジェクトに関する情報を追跡することができる
ことを特徴とする請求項２８に記載のコンピュータ読み取り可能な記録媒体。
【請求項３０】
前記１つまたは複数のランタイム置換可能な変換モジュールは、前記中間表現を修正するための１つまたは複数の標準ルーチンを含む
ことを特徴とする請求項２６に記載のコンピュータ読み取り可能な記録媒体。
【請求項３１】
前記１つまたは複数のランタイム置換可能な変換モジュールは、前記タイプ情報内の特定パターンと、前記初期オブジェクト内のオブジェクトデータとのうちのどちらか、または両方に基づいて前記初期オブジェクトの前記中間表現を修正することができる
ことを特徴とする請求項２６に記載のコンピュータ読み取り可能な記録媒体。
【請求項３２】
前記ランタイム置換可能な生成モジュールは、前記最終オブジェクトをＸＭＬフォーマットで生み出すことができる
ことを特徴とする請求項２６に記載のコンピュータ読み取り可能な記録媒体。
【請求項３３】
前記ランタイム置換可能な生成モジュールは、前記中間表現に基づいて前記最終オブジェクトをインスタンス化およびポピュレートすることができる
ことを特徴とする請求項２６に記載のコンピュータ読み取り可能な記録媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変形する方法に関する。より詳細には、オブジェクトの直列化に関連し、直列化エンジンのランタイム動作を改変する拡張ルーチンを介して、直列化エンジン内の他の既存ルーチンを置き換える必要なしに、あるタイプのオブジェクトを別のタイプのオブジェクトに変形するための、初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変形する方法に関する。
【背景技術】
【０００２】
一般的な意味で、直列化（serialization）は、単一のまたはグラフの（ネストされた）メモリ内オブジェクトを、リモート位置に送信したりディスク上に永続させたりするのに適した線形のバイトシーケンスに変換することである。反対に非直列化（deserialization）は、線形のバイトシーケンスをとり、対応する単一のまたはグラフのメモリ内オブジェクトを生み出すものである。直列化も非直列化も共に、通常は元のオブジェクトの正確なクローンを生み出すことになる。

【０００３】
従来、直列化コードは、実装全体を置き換えることを除いてはカスタマイズの余地のない、モノリシック実装として記述されてきた。カスタマイズ性または拡張性がないことにより、柔軟性のない直列化機構が、開発者やその他の当事者を含めた市場に押し付けられてきた。モノリシック実装では、特定の問題に直接に対処するためのインクリメンタルな改良またはカスタマイズは不可能であることが多く、不便な次善策が必要であったり、あるいは一定の所望の動作が単に妨げられたりする場合がある。何らかの状況でカスタマイズをしようとする場合、所望の動作を実施する標準ルーチンは、一般に開発者にはアクセス不可能であり、そのため再実装する必要があり、所望のカスタマイズを開発するのに必要な労力が大幅に（しばしば困難になるほどに）増大する。この結果、通常は直列化コード開発者だけしか新しい機能を直列化コードに追加することができず、エンドユーザは、独自の拡張を開発することや既存の機能を改良することができない。
【０００４】
オブジェクトの正確なコピーが直列化および非直列化の目標であることが多いが、場合によっては、オブジェクトのタイプ（型）、名前、データをランタイム変形するのが望ましいことがある。上に示したように、例えばオブジェクトをリモート位置に送信する際に直列化および非直列化が用いられる場合がある。リモート位置は、ソースオブジェクトとは異なるいくつかのオブジェクトタイプ、オブジェクトデータ、およびオブジェクト名を予想することがある。従来の直列化コードは、オブジェクト変形を実施するように記述することができるが、変形は、ランタイムに追加することができず、すべてのユーザにとって同じであり、様々なユーザが異なるニーズを有する可能性を無視するものである。所与の変形が特定時の特定ユーザにとって極度に重要な場合がある一方、この変形の全体の関連性は、全体としてのユーザベースにとっては重大ではなく、そのため開発されない場合がある。
【０００５】
従来の直列化コードはまた、変形すべきオブジェクトを識別する点で、またはオブジェクトに含まれるデータを変形の基礎とする点で、ほとんど柔軟性をもたらさない傾向がある。したがって、直列化および非直列化をランタイムに改変するためのカスタマイズされたルーチンに従って、標準ルーチンを再実装する必要なしに、あるタイプのオブジェクトを別のタイプのオブジェクトに変形するための方法、システム、およびコンピュータプログラム製品が必要とされている。
【発明の開示】
【発明が解決しようとする課題】
【０００６】
従来のシステムには上述したような種々の問題があり、さらなる改善が望まれている。
【０００７】
本発明は、このような状況に鑑みてなされたもので、その目的とするところは、柔軟にあるタイプのオブジェクトを別のタイプのオブジェクトに変形することができる、初期タイプの初期オブジェクトを最終タイプの最終オブジェクトに変形する方法を提供することにある。
【課題を解決するための手段】
【０００８】
本発明は、初期タイプのオブジェクトを最終タイプのオブジェクトに変換するための方法、システム、およびコンピュータプログラム製品に関するものであり、変換プロセスのランタイム動作を改変またはカスタマイズすることを可能にする。以下でより完全に述べる本発明の例示的な実施形態によれば、拡張可能な直列化エンジンが、様々なタイプのオブジェクトを直列化、非直列化、および変形する。直列化エンジンのランタイム動作は、所望のカスタマイズまたは拡張を実施する１つまたは複数の拡張ルーチンによって改変される。これらの拡張ルーチンは、他の既存ルーチンを置き換える必要なしに、直列化エンジンのランタイム動作を改変する。

【０００９】
例示的な一実施形態では、直列化エンジンによって処理のために受け取られた初期オブジェクトについて、タイプ情報を識別する。タイプ情報に基づいて初期オブジェクトを中間表現に変換するが、この中間表現により、オブジェクト名、オブジェクトタイプ、およびオブジェクトデータの修正を含めたランタイム修正が可能になる。初期オブジェクトの中間表現を、直列化エンジンのランタイム動作を改変する１つまたは複数の拡張ルーチンに従って修正し、中間表現を最終タイプの最終オブジェクトに変換する。
【００１０】
初期オブジェクトの中間表現は、オブジェクト名、オブジェクトタイプ、およびオブジェクトデータを含むものとすることができ、これらはそれぞれ、拡張ルーチンによって修正することができる。中間表現はまた、直列化エンジン内の１つまたは複数の標準ルーチンによって修正することもできる。中間表現の修正は、タイプ情報内の特定パターン、初期オブジェクト内のオブジェクトデータ、メタデータ、またはこれらの組合せに基づくものとすることができる。
【００１１】
初期オブジェクトがメモリ内オブジェクトである場合は、直列化エンジンは、初期オブジェクトを直列化して最終オブジェクトを生成する。最終オブジェクトは、ＸＭＬ（eXtensible Markup Language）で、または直列化済みオブジェクトを表すのに適した他の何らかのフォーマットでフォーマットすることができる。同様に、最終オブジェクトがメモリ内オブジェクトである場合は、直列化エンジンは、初期オブジェクトを非直列化して最終オブジェクトを生成する。最終オブジェクトは、非直列化プロセスの一部としてインスタンス化およびポピュレート（populate）することができる。場合によっては、初期オブジェクトも最終オブジェクトも両方ともメモリ内オブジェクトであることがあり、あるいは、直列化エンジンがオブジェクト変形を実施するときなどのように、両方とも直列化済みオブジェクトであることがある。バッファ要件を削減するために、中間表現が最終オブジェクトに変換されるまで中間表現の修正を延期することができる。
【００１２】
本発明のその他の特徴および利点については後続の説明で述べるが、その一部はこの説明から自明であろうし、あるいは本発明の実施によってわかるであろう。本発明の特徴および利点は、添付の特許請求の範囲に特に示す手段および組合せによって実現および得ることができる。本発明のこれらおよび他の特徴は、後続の説明および添付の特許請求の範囲からより完全に明らかになるであろうし、あるいは以下に示す本発明の実施によってわかるであろう。
【発明の効果】
【００１３】
以上説明したように本発明によれば、柔軟にあるタイプのオブジェクトを別のタイプのオブジェクトに変形することができる。
【発明を実施するための最良の形態】
【００１４】
以下、図面を参照して本発明を適用できる実施形態を詳細に説明する。本発明の前述のまたは他の利点および特徴を得ることのできる方式について述べるために、添付の図面に示す本発明の特定の実施形態を参照しながら、上に簡単に述べた本発明をより具体的に述べる。これらの図面は本発明の典型的な実施形態を表したものにすぎず、したがって本発明の範囲を限定するものと考えるべきではないという理解の上に、本発明について、添付の図面を使用してさらに具体的かつ詳細に記述および説明する。
【００１５】
本発明は、初期タイプのオブジェクトを最終タイプのオブジェクトに変換するための方法、システム、およびコンピュータプログラム製品に及び、変換プロセスのランタイム動作を改変またはカスタマイズすることを可能にする。本発明の実施形態は、後で図６に関してより詳細に解説するように、様々なコンピュータハードウェアを備えた１つまたは複

数の専用コンピュータおよび／または１つまたは複数の汎用コンピュータを含むことができる。

【００１６】

図１に、本発明による例示的な直列化モジュール／直列化インフラストラクチャ１００（直列化エンジンとも呼ぶ）を示す。オブジェクトインスタンス１１０に対して、直列化モジュール１００は、対応する直列化済みＸＭＬオブジェクト１５０を作成する。同様に、ＸＭＬオブジェクト１６０に対して、直列化モジュール１００は、対応する非直列化済みオブジェクトインスタンス１７０を作成する。この出願全体を通して、直列化（例えば単一のまたはグラフのメモリ内オブジェクトを、リモート位置に送信したりディスク上に永続させたりするのに適した線形のバイトシーケンスに変換すること）、非直列化（線形のバイトシーケンスから対応する単一のまたはグラフのメモリ内オブジェクトを生み出すこと）、変形（あるオブジェクトを別のオブジェクトに変換すること）などに対する総称的な用語として、直列化を用いる場合がしばしばあることに留意されたい。例えば、ここで直列化モジュール１００が様々なタイプのオブジェクトを直列化、非直列化、変形することが、これに当てはまる。

【００１７】

直列化モジュール１００は、１つまたは複数の反映（reflection）モジュール１２０と、１つまたは複数の変換モジュール１３０と、１つまたは複数の生成モジュール１４０を備える。この例示的な実施形態では、直列化モジュール１００は、受け取ったメモリ内（in-memory）オブジェクトインスタンス１１０を、リモート位置への送信に適したＸＭＬオブジェクト１５０に変換し、受け取ったＸＭＬオブジェクトインスタンス１６０を、メモリ内オブジェクトインスタンス１７０に変換する。当然ながら、メモリ内（in-memory）およびＸＭＬは、直列化モジュール１００が生み出すことまたは受け取ることのできるオブジェクトタイプの例にすぎない。直列化モジュール１００内の各モジュール（反映モジュール１２０、変換モジュール１３０、生成モジュール１４０）は、カスタマイズされた直列化、非直列化、または変形のために、ランタイムに置き換えることができる。

【００１８】

反映モジュール１２０は、受け取ったオブジェクトインスタンス１１０および受け取ったＸＭＬオブジェクト１６０のタイプ情報を識別することを担う。タイプ情報は、マネージコード環境内のマネージドタイプ（ｍａｎａｇｅｄ　ｔｙｐｅ）に関連する、記憶されたまたは受け取られたメタデータを含むものとすることができる。あるいはタイプ情報は、コンパイル時の自動生成や、手動生成や、標準タイプ情報などを含めて、様々なソースから反映モジュール１２０に供給することができる。

【００１９】

変換モジュール１３０は、異なるタイプのオブジェクト間で変換を行う。例示的な変換プロセスは、後で図２～４に関してより詳細に述べる。異なるオブジェクト間での変換は、任意に複雑にすることができ、中間オブジェクトの生成を含めることができる。この複雑さの一部には、オブジェクト内のデータに基づいて、またオブジェクトに関連するタイプのパターンに基づいて変換することを含めることができる。例えば、どの変換を行うかは、一定のオブジェクトタイプまたはタイプ名や、タイプに関して一定の名前付けされたまたはタイプ付けされたプロパティがあることや、一定のメタデータが付加されたプロパティがあることや、オブジェクトに関連するオブジェクト名などに応じて決まるものとすることができる。あるオブジェクトを別のオブジェクトに変換することに通常なら内在するかもしれないバッファ要件を削減または回避するために、変換は、最終オブジェクトの生成まで延期することができる。

【００２０】

生成モジュール１４０は、直列化モジュール１００によって作成される最終オブジェクトを生成することを担う。ＸＭＬオブジェクト１５０の場合、生成モジュール１４０は、オブジェクトを生み出し（すなわちオブジェクトにとって適切なＸＭＬを生み出し）、オブジェクトをストリームに書き込むことができる。オブジェクトインスタンス１７０の場

合、生成モジュール１４０は、オブジェクトをインスタンス化およびポピュレートする。
【００２１】
上に示したように、直列化モジュール１００は直列化エンジンとも呼ぶ。図１に示すように、直列化エンジンは、順序付けられたいくつかのモジュールセットで構成される。これらのモジュールがひとまとまりで、すべての動作を担う。後でより詳細に述べるように、モジュールはあるタイプから別のタイプに変換する（すなわち異なるタイプ間の橋渡しをする）ので、個々のモジュールをタイプブリッジと呼ぶ。タイプブリッジは、タイプおよびインスタンスをランタイムに変形することを可能にし、かつ／あるいは、直列化、非直列化、または変形されているオブジェクトに関する情報を追跡することを可能にする。図２～４に関して、順序付けられたタイプブリッジのセットをタイプブリッジパイプラインと呼ぶが、これは一般に、順序付けられた変換モジュール１３０のセットに対応する。直列化エンジンによって実施される各動作につき、別々のタイプブリッジパイプラインが存在するものとすることができる。直列化のためのパイプライン（例えば図２）、非直列化のためのパイプライン（例えば図３）、変形のためのパイプライン（例えば図４）、オブジェクトコピーのためのパイプラインなどがある。図２～４のそれぞれについて個々に解説する前に、３つの図面のすべてに一般に適用可能な情報を以下に提示する。
【００２２】
図２～４に示す例示的なパイプラインでは、オブジェクトの直列化、非直列化、および変形を担うコード（１つまたは複数のモジュール）が、いくつかの事前定義済みタイプブリッジとして実装される。これらのモジュールは、適切なパイプライン中に配置され、ランタイムに使用される。（図１の破線の四角は、様々なタイプブリッジパイプライン中で使用される利用可能なタイプブリッジモジュールを表すものとする。）図１に示した例示的な直列化エンジン用の公開ＡＰＩ（Application Programming Interface）の大部分は、単にこの事前定義済みパイプラインセットを覆うラッパである。このことは、この直列化エンジンがいかに拡張可能であるかを例証している。すなわち、直列化エンジンは抽象パイプラインの単純なセットである。具体的なロジックの実際の実装形態は、いつでも置き換えることのできるプラグ可能モジュールに見られる。
【００２３】
図２～４に示す例示的なタイプブリッジパイプラインでは、所与のタイプブリッジが、３つのタイプのオブジェクト、すなわち初期タイプオブジェクト、中間タイプオブジェクト、最終タイプオブジェクトのうちの１つを変形することができる。図４では、初期タイプオブジェクトはマネージドコード（managed code）オブジェクトであり、最終タイプオブジェクトは、ワールドワイドウェブコンソーシアム（Ｗ３Ｃ）情報セット（Ｉｎｆｏｓｅｔ）標準に基づくＸＭＬオブジェクトである。３つの図すべてに示されている中間タイプオブジェクトまたは中間表現は、直列化エンジン内に見られる構造であり、後でより詳細に述べるように拡張性ポイントを表す。中間表現は、可変タイプに基づく可変オブジェクトである。したがって可変タイプは、挙動およびタイプ付きデータ記憶を定義する働きをし、可変オブジェクトは、タイプ付きデータを記憶し、タイプに対して定義された挙動を介して記憶済みデータに作用する働きをする。
【００２４】
図２に、初期タイプまたはフォーマット２１０のメモリ内初期オブジェクト２４０を直列化するための例示的なタイプブリッジ２００を示す。（本明細書および特許請求の範囲では、タイプという用語は、任意のオブジェクトタイプまたはフォーマットを広く含むと解釈されたい。）標準ルーチン２５０を使用して、初期オブジェクト２４０を、中間タイプまたはフォーマット２２０を有する中間表現２６０Ａに変換する。後でより詳細に述べるが、この中間表現は可変であり、オブジェクトタイプもオブジェクトデータも両方とも変更することができる。ただし、中間フォーマット２２０と初期フォーマット２１０は同じでもよく、密接に関係していてもよく、いくらか異なってもよく、完全に異なってもよい。
【００２５】

カスタム拡張ルーチン２６０が、初期オブジェクト２４０の中間表現２６０Ａを中間表現２６０Ｂに変換する。この変換には、オブジェクトタイプ、オブジェクト名、オブジェクトデータなどの変更を含めることができる。カスタム拡張ルーチン２６０は、一般的には直列化エンジンの、具体的にはタイプブリッジパイプライン２００のランタイム拡張を表す。カスタム拡張ルーチン２６０の使用により、従来の直列化実装形態の場合に通常なら必要となるような標準ルーチン２５０の再実装が必要でなかったことに留意されたい。

【００２６】

標準ルーチン２７０が、中間表現２６０Ｂを、最終タイプまたはフォーマット２３０の最終オブジェクト２８０に変換する。最終オブジェクト２８０は、リモート位置への送信や永続化などに適したものである。したがって、最終オブジェクト２８０の最終フォーマット２３０には、幅広いオブジェクトタイプが含まれる。この説明の他の部分でも同様だが、ここでオブジェクトのタイプ、フォーマット、および表現は、オブジェクトの全体的なタイプおよびフォーマット、ならびにオブジェクト内に含まれる場合のあるタイプ、フォーマット、名前、およびデータを含む広範な用語である。

【００２７】

図３に、初期タイプまたはフォーマット３３０のオブジェクト３４０を非直列化するための例示的なタイプブリッジ３００を示す。前述の図２と同様、標準ルーチン３５０が、初期オブジェクト３４０を、中間タイプまたはフォーマット３２０を有する中間表現３６０Ａに変換する。カスタム拡張ルーチン３６０が、中間表現３６０Ａを中間表現３６０Ｂに変換する。中間タイプ３２０は１つまたは複数の中間タイプを表すことに留意されたい。したがって、中間表現３６０Ａと中間表現３６０Ｂは異なるタイプとすることができるが、しかしなお、初期タイプ３３０と最終タイプまたはフォーマット３１０とに対して特に相対的な中間タイプとして、適切に指定することができる。

【００２８】

標準ルーチン３７０が、中間表現３６０Ｂを最終タイプ３１０の最終オブジェクト３８０に変換する。タイプブリッジパイプライン３００は非直列化用なので、最終オブジェクト３８０は、インスタンス化およびポピュレートされるメモリ内オブジェクトである。後でより詳細に述べるが、タイプブリッジパイプライン３００は、オブジェクトインスタンスをインスタンス化およびポピュレートするためのコードに接続される。このコードは、インスタンスファクトリまたはライタ、あるいは書込みファクトリと呼ぶことができ、図１に示した生成モジュール１４０に概して対応する。

【００２９】

図４に、初期オブジェクト４４０を最終オブジェクト４８０に変形するための例示的なタイプブリッジパイプライン４００を示す。図４で利用可能な個々のタイプブリッジは、異なる３つのオブジェクトタイプまたはフォーマット、すなわちマネージドコード／ＣＬＲフォーマットのオブジェクト４１０と、中間／フレックス（Ｆｌｅｘ）フォーマットのオブジェクト４２０と、Ｉｎｆｏｓｅｔ／ＸＭＬフォーマットのオブジェクト４３０とのうちの１つを変形することができる。ＣＬＲは、共通言語ランタイム（Ｃｏｍｍｏｎ Ｌａｎｇｕａｇｅ Ｒｕｎｔｉｍｅ）を意味し、Ｍｉｃｒｏｓｏｆｔの．ＮＥＴ（登録商標）マネージド実行環境（ｍａｎａｇｅｄ ｅｘｅｃｕｔｉｏｎ ｅｎｖｉｒｏｎｍｅｎｔ）の一部である。ＣＬＲの利点としてはとりわけ、言語間統合や言語間例外処理などが挙げられる。言語コンパイラが、タイプ、メンバ、参照を記述するためのメタデータを出力する。メタデータは、コードと共に共通言語ランタイムのポータブル実行ファイルに記憶される。当然、ＣＬＲはマネージドコードタイプの一例にすぎない。図４に示すように、オブジェクトが両方ともメモリ内オブジェクト（例えばＣＬＲフォーマットのオブジェクト）であってもよく、あるいはオブジェクトが両方とも直列化済みオブジェクト（例えばＩｎｆｏｓｅｔフォーマットのオブジェクト）であってもよい。言い換えれば、初期オブジェクトと最終オブジェクトは同じタイプとすることができる。

【００３０】

ＣＬＲオブジェクト４１０は、データと挙動の組合せを含むＣＬＲタイプのインスタン

スだが、直列化の目的に関係があるのはデータだけである。上に示したように、Ｉｎｆｏｓｅｔオブジェクトまたは表現４３０は、一定の意味論による事前定義済みデータノードセットに構成された木構造に関するＷ３Ｃ標準に従ってフォーマットされる。フレックスオブジェクト４２０は、直列化エンジン内にみられる構造であり、シリアライザ（serializer）にとっての拡張性ポイントを表す。

【００３１】

フレックスオブジェクトは、可変タイプに基づく可変オブジェクトである。可変タイプをフレックスタイプと呼ぶ。図４に示す例示的なタイプブリッジパイプライン４００では、フレックスタイプは、それに対応するＣＬＲタイプと同じ機能を提供する。すなわち、挙動およびタイプデータ記憶を定義する。同様に、フレックスオブジェクトはＣＬＲオブジェクトと同じ機能を提供する。すなわち、タイプ付きデータを記憶し、タイプに対して定義された挙動を介してこのデータに作用する。フレックスタイプおよびフレックスオブジェクトを使用する理由は、ＣＬＲタイプが不変だからである。

【００３２】

図４に示す例示的なタイプブリッジパイプラインでは、単純かつ拡張可能にするために、直列化することのできるタイプにいくつかの制約を課す。これらの制約により、シリアライザが所与のタイプを直列化および非直列化するために認識する必要のある種々のパターンおよび置換の数が削減される。このためには、直列化エンジンは、コアモデルと呼ばれるものに従うタイプを有するＣＬＲオブジェクトをどのように直列化するかを理解するだけでよい。コアモデルに準拠するタイプは、それらのデータをプロパティ（またはフィールド）として公開するか、あるいは特定のインタフェース（明示的な読出しメソッドおよび書込みメソッドを定義するもの）を実装しなければならない。さらに、これらのタイプは、公開デフォルトコンストラクタを提供する必要がある。コアモデルに従わないタイプは、直列化することができない。

【００３３】

フレックスタイプおよびフレックスオブジェクトを使用して、所与のＣＬＲオブジェクトの形状（メンバやインタフェースなど）を、コアモデルに準拠するように変更する。所与のＣＬＲオブジェクトに対して、インスタンスのＣＬＲタイプとは異なるメンバおよびタイプ情報のセットを公開するフレックスタイプを構築することができる。このフレックスタイプに基づくフレックスオブジェクトをインスタンス化することができ、このフレックスオブジェクトは、ＣＬＲオブジェクト自体へ一定の呼出しを委任する。フレックスオブジェクトはまた、委任の前と後のどちらかで、オプションでＣＬＲオブジェクト内のデータを変形することもできる。この結果、ＣＬＲオブジェクト内のデータを、コアモデルに準拠する方式を含めた様々な方式で公開することができる。したがってタイプブリッジは、コアモデルに準拠しないオブジェクトタイプで開始し、コアモデルに準拠するオブジェクトタイプを作成することができる。

【００３４】

タイプブリッジは、ＣＬＲオブジェクト、フレックスオブジェクト、およびＩｎｆｏｓｅｔ表現を、様々な方式で変形することができる。所与のタイプブリッジは、それが作用する入力タイプと、それが作成または生成する出力タイプとを有する。この出力は、パイプライン中の次のタイプブリッジに渡される。例示的なタイプブリッジパイプライン４００では、以下の変形が可能である。

【００３５】

【表１】

入力タイプ	出力タイプ	説明
CLR	CLR	CLRオブジェクトを新しいCLRオブジェクトに変形する
CLR	Flex	CLRオブジェクトをフレックスオブジェクトに変形する
CLR	Infoset	CLRオブジェクトをInfosetオブジェクトに変形する
Flex	Flex	フレックスオブジェクトを新しいフレックスオブジェクトに変形する
Flex	CLR	フレックスオブジェクトをCLRオブジェクトに変形する
Flex	Infoset	フレックスオブジェクトをInfosetオブジェクトに変形する
Infoset	Infoset	Infosetオブジェクトを新しいInfosetに変形する
Infoset	Flex	Infosetオブジェクトをフレックスオブジェクトに変形する
Infoset	CLR	InfosetオブジェクトをCLRオブジェクトに変形する

【００３６】
様々な分類のタイプブリッジが構成されて、直列化エンジンの基礎動作が提供される。（図２および３では総称的なタイプを参照しているが、以下では、さらにコンテキストを提供するために、図４に示した特定タイプを使用してこれらの図に言及する。）

【００３７】
１．直列化は、ＣＬＲオブジェクトをＩｎｆｏｓｅｔオブジェクトまたは表現に変形する。この動作を実施するために、ＣＬＲ－フレックスタイプブリッジ（例えば標準ルーチン２５０）と、任意の数のフレックス－フレックスブリッジと、フレックス－Ｉｎｆｏｓｅｔタイプブリッジ（例えば標準ルーチン２７０）とを含むタイプブリッジパイプライン（図２に示したものなど）が存在する。

【００３８】
２．非直列化は、Ｉｎｆｏｓｅｔ表現をＣＬＲオブジェクトに変形する。この動作を実施するために、Ｉｎｆｏｓｅｔフレックスタイプブリッジ（例えば標準ルーチン３５０）と、任意の数のフレックス－フレックスブリッジと、フレックス－ＣＬＲタイプブリッジ（例えば標準ルーチン３７０）とを含むタイプブリッジパイプライン（図３に示したものなど）が存在する。

【００３９】
３．オブジェクトコピーは、ＣＬＲオブジェクトのディープコピーを生み出すのに使用される。この動作を実施するために、ＣＬＲ－ＣＬＲタイプブリッジを含むタイプブリッジパイプラインが存在する。

【００４０】
４．オブジェクト変形（図４）は、インスタンスデータ（中間表現４６０Ａおよび４６０Ｂ）に対してオプションの変形（標準またはカスタム拡張ルーチン４６０）を行いながら、１つまたは複数のＣＬＲオブジェクト（初期オブジェクト４４０）のディープコピー（最終オブジェクト４８０）を生み出す。この動作を実施するために、ＣＬＲ－フレックスタイプブリッジ（標準またはカスタムルーチン４５０）と、変形を行う１つまたは複数のオプションのフレックス－フレックスタイプブリッジ（標準またはカスタム拡張ルーチン４６０）と、フレックス－ＣＬＲタイプブリッジ（標準またはカスタムルーチン４７０

）とを含むタイプブリッジパイプラインが存在する。

【００４１】

５．Ｉｎｆｏｓｅｔ変形は、コピーを生み出し、オプションでＩｎｆｏｓｅｔオブジェクトを変形する。オブジェクトコピーと同様、この動作を実施するために、ＩｎｆｏｓｅｔＩｎｆｏｓｅｔタイプブリッジを含むタイプブリッジパイプラインが存在する。

【００４２】

最後の３つのオプションは、これらが実施される方式のため注目に値する。他の実装形態ではオブジェクトデータまたはＩｎｆｏｓｅｔデータをバッファに入れるが、本発明の実施形態では、バッファリング要件を削減するために変形を延期することができる。この結果、性能およびリソース管理を大きく改善することができる。

【００４３】

上記の動作をサポートするために、直列化エンジンは、適切な変形を実施するストックまたはベースタイプブリッジを提供する。図４で、標準またはカスタムルーチン４５０、４６０、４７０のいずれかは、ストックタイプブリッジすなわちカスタム代替とすることができる。拡張可能な構成機構を使用して、ランタイムに、適切なタイプブリッジを識別しパイプラインにロードする。直列化エンジンは、これらのストックパイプラインを使用して、要求された動作を実施する。しかし、エンジンは特定のストック実装形態ではなく抽象タイプブリッジの概念を用いるので、ストックタイプブリッジはいつでも置き換えることができる。例示的な一実施形態では、パイプラインは単にパイプラインに対するタイプブリッジのリストを含むだけであり、リストを変更するとパイプラインが変更される。この例示的な実施形態では、特定のタイプブリッジがオブジェクトのために呼び出された後は、他のタイプブリッジがこのオブジェクトのために呼び出されることはない。

【００４４】

例示的な一実施形態では、ＣＬＲ４１０、フレックス４２０、およびＩｎｆｏｓｅｔ４３０は、図２に示した直列化の場合の初期フォーマット２１０、中間フォーマット２２０、および最終フォーマット２３０に対応し、図３に示した非直列化の場合の最終タイプ３１０、中間タイプ３２０、および初期タイプ３３０に対応することに留意されたい。フレックスオブジェクトは、ＣＬＲとＩｎｆｏｓｅｔとの間の中間フォーマットである。この例示的な実施形態では、タイプブリッジは、ＣＬＲからＩｎｆｏｓｅｔに、またその逆に直接に変形することはできない。このことはとりわけ、この例示的な直列化エンジンを単純化するのに役立つ。直列化エンジンの基礎機能または動作はストックタイプブリッジによって定義されるが、開発者が予想することのできる追加機能（レガシープログラミングモデルのサポートなど）も多くある。ストックタイプブリッジは、これらの機能を実装するように設計されたものとすることもできるが、この代わりに、この目的に役立ついくつかのストックフレックス－フレックスタイプブリッジがある。この手法により、ストックタイプブリッジは単純かつ拡張可能になる。この結果、様々な開発者が、標準機能に修正を加え、開発者自身の新しい機能を提供することができる。

【００４５】

この例示的な実施形態で、ＦｉｒｓｔＮａｍｅおよびＬａｓｔＮａｍｅという２つのプロパティを有するＰｅｒｓｏｎという名前のＣＬＲタイプに対する直列化プロセスを考えてみる。このタイプのインスタンスを直列化（図２参照）するには、ストックＣＬＲ－フレックスタイプブリッジと、ストックフレックス－Ｉｎｆｏｓｅｔタイプブリッジとを含むパイプラインが必要である。直列化エンジンは、ＰｅｒｓｏｎインスタンスをＣＬＲ－フレックスタイプブリッジに渡す。このタイプブリッジは、Ｐｅｒｓｏｎインスタンスに基づいて、かつＰｅｒｓｏｎインスタンスに委任して、新しいフレックスオブジェクトインスタンスを返す。次いでフレックスオブジェクトは、フレックス－Ｉｎｆｏｓｅｔタイプブリッジに渡される。

【００４６】

フレックス－Ｉｎｆｏｓｅｔタイプブリッジは、フレックスオブジェクトをＩｎｆｏｓｅｔ表現に変形または変換することを担う。変換前に、ストックフレックス－Ｉｎｆｏｓ

ｅｔタイプブリッジは、フレックスオブジェクトをＩｎｆｏｓｅｔにマッピングする方式を判定する。この例でのストック実装は、スキーマ言語を使用し、この言語で定義される構造でマッピングを定義する。タイプブリッジは置き換え可能なので、新しいスキーマ言語に対するサポートを含めた新しいマッピング機構を導入することもでき、これは直列化エンジン内の別の拡張性ポイントを表す。マッピングプロセスが完了すると、フレックスオブジェクトはＩｎｆｏｓｅｔ表現に変形され、ストリームに書き込まれる。

【００４７】

先に簡単に触れたように、直列化エンジン内のタイプブリッジは、ライタファクトリに接続される。ライタファクトリは、データを書き込むことのできるリソースを生み出すことを担う。このリソースは任意のターゲットにデータを書き込むことができるが、最も一般的な宛先は、データストリーム（搬送に向けて直列化の後で）、およびＣＬＲオブジェクト（非直列化の後で）である。この例示的な実施形態でのストックライタファクトリは、ユーザから供給されたデータストリームに書き込むリソースを返す。このファクトリによって作成されたリソースを、任意の所望のフォーマットでストリームに書き込むことができる。したがって、これはＸＭＬ直列化フォーマットに限定されず、これによりライタファクトリは置き換え可能になり、直列化エンジン内にさらに別の拡張性ポイントがもたらされる。

【００４８】

この例示的な実施形態でのＩｎｆｏｓｅｔ表現の非直列化（例えば図３参照）は、ストックＩｎｆｏｓｅｔフレックスタイプブリッジと、ストックフレックス－ＣＬＲタイプブリッジとを含むパイプラインを必要とする。直列化エンジンは、ソースＩｎｆｏｓｅｔを表すユーザ提供のストリームと、非直列化の対象であるＣＬＲタイプ（Ｐｅｒｓｏｎ）とを、第１のタイプブリッジ（Ｉｎｆｏｓｅｔフレックス）に渡す。このタイプブリッジは、ストリームに委任するＰｅｒｓｏｎタイプに基づいて、新しいフレックスオブジェクトインスタンスを生み出す。得られたフレックスオブジェクトはフレックス－ＣＬＲタイプブリッジに渡され、フレックス－ＣＬＲタイプブリッジは、Ｐｅｒｓｏｎのインスタンスに、フレックスオブジェクトからのデータをポピュレートする（フレックスオブジェクトは委任しているので、実際にはストリーム中にある）。直列化の場合と同様、非直列化パイプラインはライタファクトリ中で終了する。非直列化パイプラインの場合のストック書込みファクトリは、非直列化の対象であるＣＬＲタイプのインスタンスを生み出すことを担う。

【００４９】

直列化および非直列化に加えて、Ｐｅｒｓｏｎタイプを変形することが望ましい場合もある。上に示したように、Ｐｅｒｓｏｎタイプの形状は、ＦｉｒｓｔＮａｍｅおよびＬａｓｔＮａｍｅという２つのプロパティを含む。例えば、このＰｅｒｓｏｎ定義を使用するあるアプリケーションが、異なるＰｅｒｓｏｎ定義（例えばＦｕｌｌＮａｍｅという１つのプロパティだけを有するＰｅｒｓｏｎ）を使用する別のアプリケーションと対話すると仮定する。両方のアプリケーションが同じＰｅｒｓｏｎタイプを使用するようにすることが１つのオプションだが、これは常に可能というわけではない（両方のアプリケーションがすでに記述されている場合がある）。

【００５０】

述べた例示的な実施形態によれば、一方のアプリケーション中のＰｅｒｓｏｎインスタンスの形状を他方のアプリケーション中で予想される形状に変形するタイプブリッジを生み出すことができる。この変形（図２参照）を行うには、新しいフレックス－フレックスタイプブリッジ（例えばカスタム拡張ルーチン２６０）を構築し、直列化パイプライン中のストックＣＬＲ－フレックスタイプブリッジ（例えば標準ルーチン２５０）の後に配置する必要がある。直列化プロセスの間、このタイプブリッジには、Ｐｅｒｓｏｎインスタンスに委任するフレックスオブジェクトが渡される。タイプブリッジは、異なる形状（単一のＦｕｌｌＮａｍｅプロパティ）を有する新しいフレックスタイプを構築する。このフレックスタイプに基づいて、元のフレックスオブジェクト上にみられるＦｉｒｓｔＮａｍ

ｅおよびＬａｓｔＮａｍｅを連結する新しいフレックスオブジェクト（これもまたＰｅｒｓｏｎインスタンスに委任する）が生み出される。このフレックスオブジェクトは、２つのプロパティではなく１つのプロパティを直列化するストックフレックス－Ｉｎｆｏｓｅｔタイプブリッジ（例えば標準ルーチン２７０）に渡される。フレックス－Ｉｎｆｏｓｅｔタイプブリッジが新しいＦｕｌｌＮａｍｅプロパティの値を要求するまでは、連結が実際に行われなくてもよいことは、注目に値する。このため変形は、Ｉｎｆｏｓｅｔすなわち最終オブジェクトが生み出されるまで延期される。

【００５１】

したがって、本発明による直列化エンジンは、システム間およびタイプ間で変形するための拡張可能アーキテクチャを提供することができ、これは、プラグ可能なタイプおよびデータ変形に対するサポート、可変タイプおよびオブジェクトに対するサポート、プラグ可能なスキーマタイプシステムに対するサポート、プラグ可能なデータフォーマットに対するサポートなどを含む。

【００５２】

本発明はまた、機能的ステップおよび／または非機能的動作を含む方法の点から述べることができる。以下、本発明を実施する際に行うことのできる動作およびステップについて述べる。通常、機能的ステップは、達成される結果の点から本発明を記述するものであり、非機能的動作は、特定の結果を達成するためのより具体的なアクションを記述するものである。機能的ステップおよび非機能的動作を特定の順序で記述し特許請求する場合があるが、本発明は、どんな特定の順序付けにも、動作および／またはステップの組合せにも限定されるとは限らない。

【００５３】

図５Ａ，５Ｂに、本発明によりオブジェクトを直列化および非直列化する方法の例示的な動作およびステップを示す。これは、直列化エンジンによるランタイム処理のために初期タイプの初期オブジェクトを受け取る動作（５１２）を含むことができる。初期オブジェクトのタイプ情報を識別するステップ（５２０）は、タイプ情報を受け取る動作（５２２）を含むことができる。タイプ情報は、マネージドコードに関連するメタデータとして供給することができる。初期タイプ情報に基づいて初期オブジェクトを中間表現に変換するステップ（５３０）は、タイプ情報に基づいて中間表現を生成する動作（図示せず）と、中間表現を修正するために、１つまたは複数のカスタム拡張ルーチンを呼び出す動作（５３２）と、１つまたは複数の標準ルーチンを呼び出す動作（５３４）とを含むことができる。１つまたは複数の拡張ルーチンは、直列化エンジンのランタイム動作を改変する。

【００５４】

中間表現は、オブジェクト名、オブジェクトタイプ、および／またはオブジェクトデータを含むことができることに留意されたい。図示していないが、中間表現を修正するステップ（５４０）もまた、中間表現を修正するために、１つまたは複数のカスタム拡張ルーチンを呼び出す動作（図示せず）および１つまたは複数の標準ルーチンを呼び出す動作（図示せず）を含むことができる。中間表現を修正する動作（５４０）はさらに、オブジェクトの名前、タイプ、および／またはデータを変更する動作（５４２）を含むことができる。修正を延期するステップ（５５０）は、中間表現を実際には修正せずに、中間表現をどのように修正するかを指定する動作（５５２）を含むことができる。延期することは、中間表現をその場で修正することに通常なら関連するバッファ要件および処理要件を削減することに役立つ。

【００５５】

初期オブジェクトの中間表現を最終タイプまたはフォーマットの最終オブジェクトに変換するステップ（５６０）は、以下の動作を含むことができる。直列化（５６３）のときは、このステップは、最終オブジェクトを生み出すまたは生成する動作（５６５）を含むことができる。前述の例示的な一実施形態では、最終オブジェクトを搬送に向けてＸＭＬにフォーマットする。したがって、最終オブジェクトを生み出すまたは生成すること（５６５）は、適切なＸＭＬを生成し、最終オブジェクトをストリームに書き込むことを含む

ことができる。あるいは最終オブジェクトは、ディスクに永続させるためにフォーマットするか、または直列化された初期オブジェクトを表すのに適した他の任意のフォーマットにフォーマットすることができる。非直列化（５６４）のときは、このステップは、最終オブジェクトをインスタンス化する動作（５６６）およびポピュレートする動作（５６８）を含むことができる。変換するステップ（５６０）の間、どのように変更を行うかを示しているが実際には変更を行っていない延期された修正があれば、カスタム拡張ルーチンおよび標準ルーチンを呼び出す。

【００５６】

本発明の範囲内の実施形態には、コンピュータ実行可能命令またはデータ構造を搬送するまたは記憶したコンピュータ可読媒体も含まれる。このようなコンピュータ可読媒体は、汎用または専用コンピュータからアクセスすることのできる任意の利用可能な媒体とすることができる。限定ではなく例として、このようなコンピュータ可読媒体には、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ（electrically erasable programmable read-only memory）、ＣＤ（compact disc）－ＲＯＭまたは他の光ディスク記憶装置、磁気ディスク記憶装置または他の磁気記憶デバイスを含めることができ、あるいは、所望のプログラムコード手段をコンピュータ実行可能命令またはデータ構造の形で搬送または記憶するのに使用でき、汎用または専用コンピュータからアクセスできるその他の任意の媒体を含めることができる。情報が、ネットワークまたは他の通信接続（配線式または無線式、あるいは配線式と無線式の組合せ）を介してコンピュータに転送または提供されるとき、コンピュータはこの接続をコンピュータ可読媒体と見なすのが適切である。したがって、このような接続はどれも、コンピュータ可読媒体と呼ぶのが適切である。以上の組合せもまた、コンピュータ可読媒体の範囲に含めるべきである。コンピュータ実行可能命令は例えば、汎用コンピュータ、専用コンピュータ、または専用処理デバイスに何らかの機能または機能グループを実施させる命令およびデータを含む。

【００５７】

図６および以下の解説は、本発明を実施することのできる適したコンピューティング環境に関する簡潔で一般的な説明を提供するものである。必須ではないが本発明は、ネットワーク環境でコンピュータによって実行されるプログラムモジュールなどのコンピュータ実行可能命令の一般的な状況で述べる。一般にプログラムモジュールは、特定のタスクを実施するか特定の抽象データ型を実装するルーチン、プログラム、オブジェクト、コンポーネント、データ構造などを含む。コンピュータ実行可能命令、関連するデータ構造、およびプログラムモジュールは、本明細書に開示した方法のステップを実行するためのプログラムコード手段の例を表す。このような実行可能命令または関連するデータ構造の特定のシーケンスは、このようなステップで述べた機能を実施するための対応する動作の例を表す。

【００５８】

当業者なら理解するであろうが、本発明は、パーソナルコンピュータ、ハンドヘルドデバイス、マルチプロセッサシステム、マイクロプロセッサベースのまたはプログラム可能な家庭用電化製品、ネットワークＰＣ（personal computer）、ミニコンピュータ、メインフレームコンピュータなど、多くのタイプのコンピュータシステム構成を含むネットワークコンピューティング環境で実施することができる。本発明は分散コンピューティング環境で実施することもでき、その場合、タスクは、通信ネットワークを介して（配線式リンクまたは無線リンク、あるいは配線式リンクと無線リンクの組合せによって）リンクされたローカルおよびリモートの処理デバイスによって実施される。分散コンピューティング環境では、プログラムモジュールは、ローカルとリモートの両方のメモリ記憶デバイスに位置することができる。

【００５９】

図６を参照すると、本発明を実施するための例示的なシステムは、従来型のコンピュータ６２０の形をとる汎用コンピューティングデバイスを含み、コンピュータ６２０は、プロセッサ６２１と、システムメモリ６２２と、システムメモリ６２２を含めた様々なシス

テムコンポーネントをプロセッサ６２１に結合するシステムバス６２３とを備える。システムバス６２３は、様々なバスアーキテクチャのいずれかを用いた、メモリバスまたはメモリコントローラ、周辺バス、ローカルバスを含めて、いくつかのタイプのバス構造のいずれかとすることができる。システムメモリは、読取り専用メモリ（ＲＯＭ）６２４およびランダムアクセスメモリ（ＲＡＭ）６２５を含む。ＲＯＭ６２４には、起動中などにコンピュータ６２０内の要素間で情報を転送するのを助ける基本ルーチンを含むＢＩＯＳ（ｂａｓｉｃ　ｉｎｐｕｔ／ｏｕｔｐｕｔ　ｓｙｓｔｅｍ）６２６を記憶することができる。

【００６０】

コンピュータ６２０はまた、磁気ハードディスク６３９に対して読み書きするための磁気ハードディスクドライブ６２７と、リムーバブル磁気ディスク６２９に対して読み書きするための磁気ディスクドライブ６２８と、ＣＤ－ＲＯＭやその他の光媒体などリムーバブル光ディスク６３１に対して読み書きするための光ディスクドライブ６３０を備えることもできる。磁気ハードディスクドライブ６２７、磁気ディスクドライブ６２８、および光ディスクドライブ６３０は、それぞれハードディスクドライブインタフェース６３２、磁気ディスクドライブインタフェース６３３、および光ドライブインタフェース６３４によってシステムバス６２３に接続される。これらのドライブおよび関連するコンピュータ可読媒体は、コンピュータ実行可能命令、データ構造、プログラムモジュール、およびその他のデータの記憶域をコンピュータ６２０に提供する。本明細書に述べるこの例示的環境は、磁気ハードディスク６３９、リムーバブル磁気ディスク６２９、およびリムーバブル光ディスク６３１を採用しているが、その他のタイプのデータ記憶用コンピュータ可読媒体を使用することもでき、これらには、磁気カセット、フラッシュメモリカード、ＤＶＤ（digital versatile disk）、ベルヌーイカートリッジ、ＲＡＭ、ＲＯＭなどが含まれる。

【００６１】

ハードディスク６３９、磁気ディスク６２９、光ディスク６３１、ＲＯＭ６２４、またはＲＡＭ６２５には、１つまたは複数のプログラムモジュールを含むプログラムコード手段を記憶することができる。これらには、オペレーティングシステム６３５、１つまたは複数のアプリケーションプログラム６３６、その他のプログラムモジュール６３７、およびプログラムデータ６３８が含まれる。ユーザは、キーボード６４０、ポインティングデバイス６４２を介して、またはマイクロホン、ジョイスティック、ゲームパッド、衛星放送受信アンテナ、スキャナなど他の入力デバイス（図示せず）を介して、コンピュータ６２０にコマンドおよび情報を入力することができる。これらおよび他の入力デバイスは、システムバス６２３に結合されたシリアルポートインタフェース６４６を介してプロセッサ６２１に接続されることが多い。あるいは入力デバイスは、パラレルポート、ゲームポート、ＵＳＢ（Universal Serial Bus）など、他のインタフェースで接続してもよい。モニタ６４７または他の表示デバイスもまた、ビデオアダプタ６４８などのインタフェースを介してシステムバス６２３に接続される。モニタに加えて、パーソナルコンピュータは通常、スピーカやプリンタなど他の周辺出力デバイス（図示せず）も備える。

【００６２】

コンピュータ６２０は、リモートコンピュータ６４９ａおよび６４９ｂなど１つまたは複数のリモートコンピュータへの論理接続を用いて、ネットワーク化された環境で動作することができる。リモートコンピュータ６４９ａおよび６４９ｂはそれぞれ、別のパーソナルコンピュータ、サーバ、ルータ、ネットワークＰＣ、ピアデバイス、またはその他の一般的なネットワークノードとすることができ、図６にはメモリ記憶デバイス６６０ａおよび６６０ｂとそれらに関連するアプリケーションプログラム６３６ａおよび６３６ｂしか示していないが、通常は、コンピュータ６２０に関して上述した要素の多くまたはすべてを備える。図６に示す論理接続は、ローカルエリアネットワーク（ＬＡＮ）６６１およびワイドエリアネットワーク（ＷＡＮ）６６２を含むが、これらはこの図に限定としてではなく例として提示するものである。このようなネットワーキング環境は、オフィス全体

または企業全体のコンピュータネットワーク、イントラネット、およびインターネットでよくみられるものである。

【００６３】

コンピュータ６２０は、ＬＡＮネットワーキング環境で使用されるときは、ネットワークインタフェースまたはアダプタ６６３を介してローカルネットワーク６６１に接続される。ＷＡＮネットワーキング環境で使用されるときは、インターネットなどのワイドエリアネットワーク６６２を介した通信を確立するためのモデム６６４、無線リンク、またはその他の手段を備えることができる。モデム６６４は内蔵でも外付けでもよく、シリアルポートインタフェース６４６を介してシステムバス６２３に接続される。ネットワーク化された環境では、コンピュータ６２０に関して示したプログラムモジュールまたはその一部をリモートのメモリ記憶デバイスに記憶することができる。図示のネットワーク接続は例示的なものであり、ワイドエリアネットワーク６６２を介してコンピュータ間で通信を確立する他の手段を使用することもできることは理解されるであろう。

【００６４】

本発明は、その趣旨または本質的な特徴を逸脱することなく、他の特定の形で実施することもできる。述べた実施形態は、あらゆる点で、単に例示的であって限定的ではないものと考えるべきである。したがって、本発明の範囲は、以上の記述によってではなく添付の特許請求の範囲によって示す。特許請求の範囲の均等物の意味および範囲に含まれるすべての変更は、本発明の範囲に含まれる。

【図面の簡単な説明】

【００６５】

【図１】本発明を適用できる実施形態の例示的な直列化モジュールおよび直列化インフラストラクチャを示す図である。

【図２】本発明を適用できる実施形態の例示的な直列化パイプラインのコンテキストにおけるオブジェクト変換を示す図である。

【図３】本発明を適用できる実施形態の例示的な非直列化パイプラインのコンテキストにおけるオブジェクト変換を示す図である。

【図４】本発明を適用できる実施形態の例示的なタイプ変形パイプラインのコンテキストにおけるオブジェクト変換を示す図である。

【図５Ａ】本発明を適用できる実施形態のオブジェクトを直列化、非直列化、および変形する方法の例示的な動作およびステップを示す図である。

【図５Ｂ】本発明を適用できる実施形態のオブジェクトを直列化、非直列化、および変形する方法の例示的な動作およびステップを示す図である。

【図６】本発明を適用できる実施形態の動作環境を提供する例示的なシステムを示す図である。

【符号の説明】

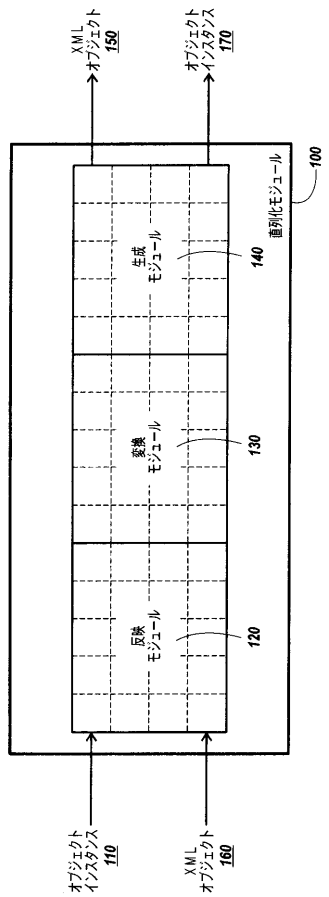
【００６６】

１００　直列化モジュール
１１０　オブジェクトインスタンス
１２０　反映モジュール
１３０　変換モジュール
１４０　生成モジュール
１５０　ＸＭＬオブジェクト
１６０　ＸＭＬオブジェクト
１７０　オブジェクトインスタンス
２００　タイプブリッジパイプライン
２１０　初期フォーマット
２２０　中間フォーマット
２３０　最終フォーマット
２４０　初期オブジェクト

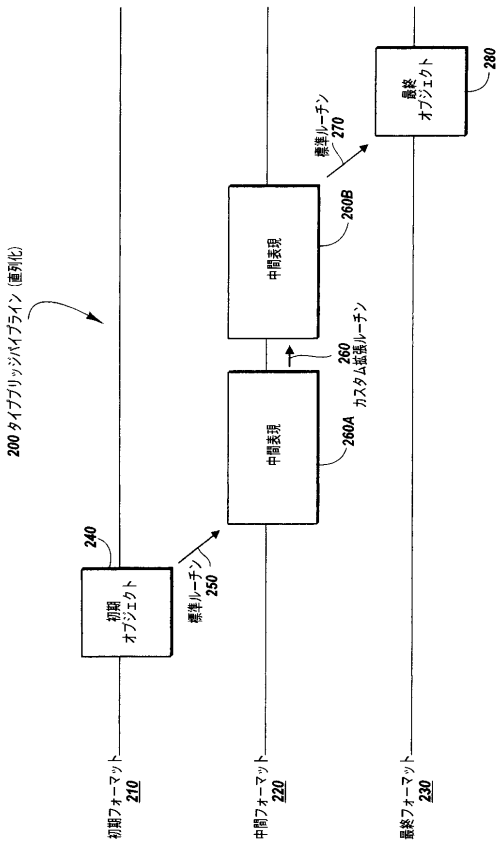
２５０　標準ルーチン
２６０　カスタム拡張ルーチン
２６０Ａ　中間表現
２６０Ｂ　中間表現
２７０　標準ルーチン
２８０　最終オブジェクト
３００　タイプブリッジパイプライン
３１０　最終タイプ
３２０　中間タイプ
３３０　初期タイプ
３４０　初期オブジェクト
３５０　標準ルーチン
３６０　カスタム拡張ルーチン
３６０Ａ　中間表現
３６０Ｂ　中間表現
３７０　標準ルーチン
３８０　最終オブジェクト
４００　タイプブリッジパイプライン
４１０　マネージドコード／ＣＬＲフォーマットのオブジェクト
４２０　中間／フレックスフォーマットのオブジェクト
４３０　Ｉｎｆｏｓｅｔ／ＸＭＬフォーマットのオブジェクト
４４０　初期オブジェクト
４５０　標準またはカスタムルーチン
４６０　標準またはカスタム拡張ルーチン
４６０Ａ　中間表現
４６０Ｂ　中間表現
４７０　標準またはカスタムルーチン
４８０　最終オブジェクト
６２０　コンピュータ
６２１　プロセッサ
６２２　システムメモリ
６２３　システムバス
６２４　読取り専用メモリ（ＲＯＭ）
６２５　ランダムアクセスメモリ（ＲＡＭ）
６２６　ＢＩＯＳ
６２７　磁気ハードディスクドライブ
６２８　磁気ディスクドライブ
６２９　リムーバブル磁気ディスク
６３０　光ディスクドライブ
６３１　光ディスク
６３２　ハードディスクドライブインタフェース
６３３　磁気ディスクドライブインタフェース
６３４　光ドライブインタフェース
６３５　オペレーティングシステム
６３６　アプリケーションプログラム
６３６ａ　アプリケーションプログラム
６３６ｂ　アプリケーションプログラム
６３７　その他のプログラムモジュール
６３８　プログラムデータ
６３９　磁気ハードディスク

６４０　キーボード
６４２　ポインティングデバイス
６４６　シリアルポートインタフェース
６４７　モニタ
６４８　ビデオアダプタ
６４９ａ　リモートコンピュータ
６４９ｂ　リモートコンピュータ
６６１　ローカルエリアネットワーク（ＬＡＮ）
６６２　ワイドエリアネットワーク（ＷＡＮ）
６６３　ネットワークインタフェース
６６４　モデム

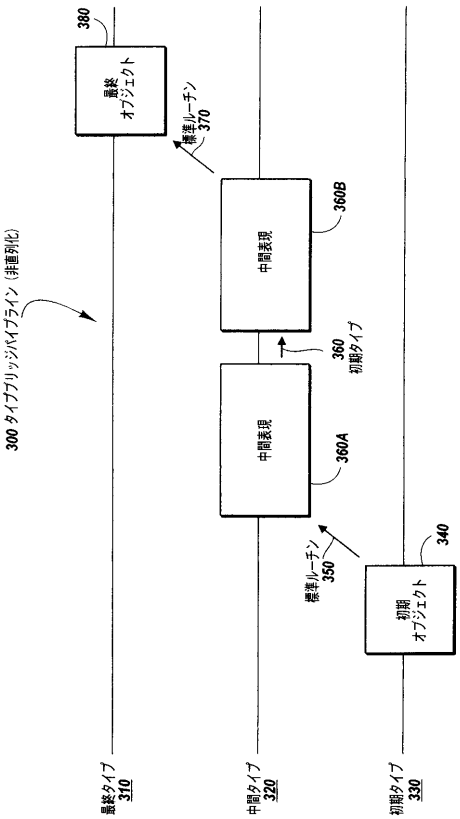
【図１】



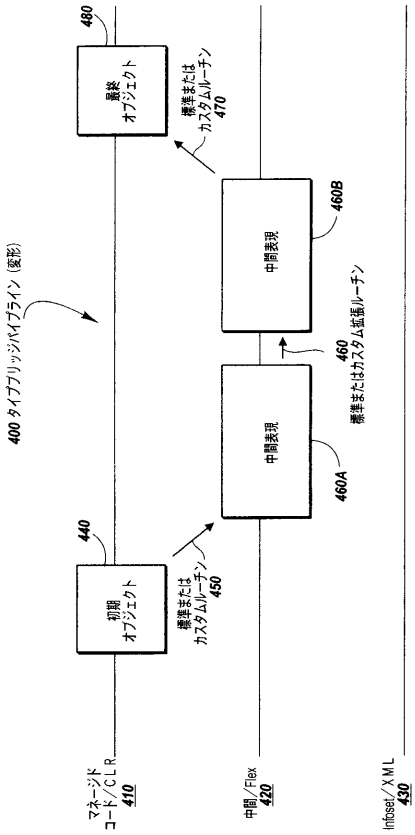
【図２】



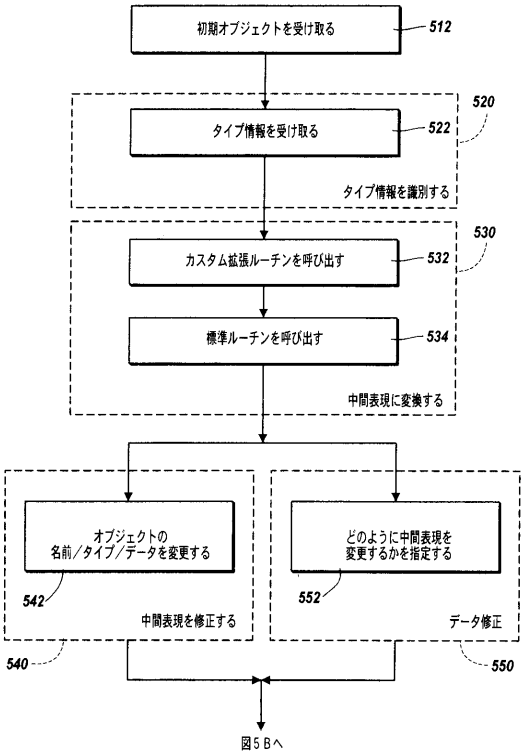
【図３】


380
最終オブジェクト
標準ルーチン 370
中間表現
360B
360 初期タイプ
中間表現
360A
標準ルーチン 350
340
初期オブジェクト
300 タイプブリッジパイプライン（非直列化）
最終タイプ 310
中間タイプ 320
初期タイプ 330

【図４】


480
最終オブジェクト
標準またはカスタムルーチン 470
中間表現
460B
460
標準またはカスタム拡張ルーチン
中間表現
460A
440
初期オブジェクト
標準またはカスタムルーチン 450
400 タイプブリッジパイプライン（変形）
マネージドコード/CLR 410
中間/Flex 420
Infoset/XML 430

【図５Ａ】


初期オブジェクトを受け取る 512
520
タイプ情報を受け取る 522
タイプ情報を識別する
530
カスタム拡張ルーチンを呼び出す 532
標準ルーチンを呼び出す 534
中間表現に変換する
オブジェクトの名前/タイプ/データを変更する
542
中間表現を修正する
どのように中間表現を変更するかを指定する
552
データ修正
540
550
図５Ｂへ

【図５Ｂ】

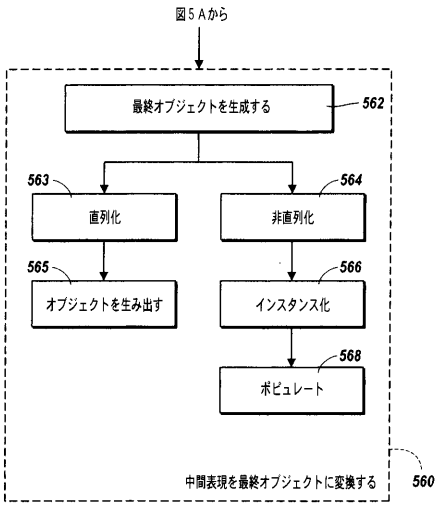
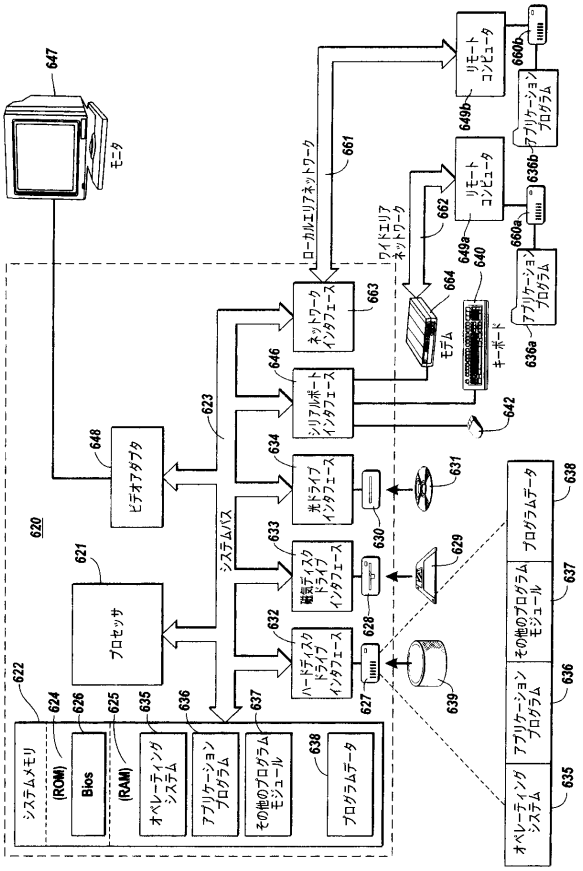


図５Ａから
最終オブジェクトを生成する 562
563
直列化
564
非直列化
565
オブジェクトを生み出す
566
インスタンス化
568
ポピュレート
中間表現を最終オブジェクトに変換する
560

【図６】


モニタ
647
ローカルエリアネットワーク
661
リモートコンピュータ
649b
660b
アプリケーションプログラム
636b
ワイドエリアネットワーク
662
リモートコンピュータ
649a
660a
アプリケーションプログラム
636a
ネットワークインタフェース
663
モデム
664
キーボード
640
シリアルポートインタフェース
646
642
ビデオアダプタ
648
623
システムバス
光ドライブインタフェース
634
631
磁気ディスクドライブインタフェース
633
630
ハードディスクドライブインタフェース
632
628
629
627
639
620
プロセッサ
621
システムメモリ
622
(ROM)
624
Bios
626
(RAM)
625
オペレーティングシステム
635
アプリケーションプログラム
636
その他のプログラムモジュール
637
プログラムデータ
638
オペレーティングシステム
635
アプリケーションプログラム
636
その他のプログラムモジュール
637
プログラムデータ
638

フロントページの続き

(72)発明者 ソーミー ケー. スリニバサン
アメリカ合衆国 ９８０５３ ワシントン州 レッドモンド ２２８ テラス ノースイースト ９７５５
(72)発明者 ナターシャ エイチ. ジェサナンダニ
アメリカ合衆国 ９８００５ ワシントン州 ベルビュー １３７ アベニュー ノースイースト ８００ シー９－２０４
(72)発明者 ヤン エリック クリステンセン
アメリカ合衆国 ９８１２２ ワシントン州 シアトル １６ アベニュー ８１７
(72)発明者 エレナ エー. カリティディ
アメリカ合衆国 ９８０７４ ワシントン州 サマミッシュ ２３９ アベニュー ノースイースト ４１３
(72)発明者 ダグラス エム. パーディー
アメリカ合衆国 ９８０１４ ワシントン州 カーネーション ノースイースト ６３ ウェイ ２８６３４

審査官 田川 泰宏

(56)参考文献 Don Box, HOUSE OF WEB SERVICES, ｍｓｄｎ ｍａｇａｚｉｎｅ, 日本, 株式会社アスキー, ２００２年 3月１８日, No.24, p.88-95
Scott Short, XML Ｗｅｂサービスとスキーマを読み解く, ｍｓｄｎ ｍａｇａｚｉｎｅ, 日本, 株式会社アスキー, ２００３年 1月２５日, No.33, p.25-32

(58)調査した分野(Int.Cl., ＤＢ名)
Ｇ０６Ｆ １２／００