(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5359704号
(P5359704)
(45)発行日　平成25年12月4日(2013.12.4)　(24)登録日　平成25年9月13日(2013.9.13)

(51)Int.Cl.　F I
G06F 9/44 (2006.01)　G06F 9/06 620K

請求項の数 6 (全 20 頁)

(21)出願番号　特願2009-205497(P2009-205497)
(22)出願日　平成21年9月7日(2009.9.7)
(65)公開番号　特開2011-59751(P2011-59751A)
(43)公開日　平成23年3月24日(2011.3.24)
審査請求日　平成24年8月27日(2012.8.27)

(73)特許権者　390002761
キヤノンマーケティングジャパン株式会社
東京都港区港南２丁目１６番６号
(73)特許権者　312000206
キヤノンＭＪアイティグループホールディングス株式会社
東京都品川区東品川２丁目４番１１号
(73)特許権者　301015956
キヤノンソフトウェア株式会社
東京都品川区東品川二丁目４番１１号
(74)代理人　100189751
弁理士　木村　友輔
(74)代理人　100188938
弁理士　榛葉　加奈子

最終頁に続く

(54)【発明の名称】プログラム生成システムおよびプログラム生成装置およびプログラム生成方法およびプログラムならびに記録媒体

(57)【特許請求の範囲】
【請求項１】
Ｗｅｂサービス記述言語ファイルを有するＷｅｂサービスを提供するＷｅｂサービス提供装置、及び前記Ｗｅｂサービスの処理の代理として呼び出されるプロキシクラスファイルを介して前記Ｗｅｂサービスを利用してＷｅｂアプリケーションプログラムを実行するＷｅｂアプリサーバ装置と通信可能なプログラム生成装置であって、
前記Ｗｅｂサービス提供装置の前記Ｗｅｂサービス記述言語ファイルを読み込むＷｅｂサービス記述言語ファイル読込手段と、
前記Ｗｅｂサービス記述言語ファイル読込手段により読み込まれたＷｅｂサービス記述言語ファイルを解析してＷｅｂサービス記述言語ファイルの要素を分類し、前記プロキシクラスファイルの入出力に必要な要素を解析するＷｅｂサービス記述言語ファイル解析手段と、
前記Ｗｅｂサービス記述言語ファイル解析手段により解析された要素を抽出し、抽出した情報を基に前記プロキシクラスファイルと前記Ｗｅｂアプリケーションプログラムとの入出力データモデルとの対応を記憶したマッピングファイルを作成する際のテンプレートとなるマッピングファイルテンプレートを生成するマッピングファイルテンプレート生成手段と、
前記Ｗｅｂアプリサーバ装置から既存の第１のマッピングファイルを読み込むマッピングファイル読込手段と、
前記マッピングファイル読込手段により読み込まれた第１のマッピングファイルのユー

ザ入力項目を抽出して、該抽出されたユーザ入力項目を前記マッピングファイルテンプレート生成手段で生成されたマッピングファイルテンプレートに適応させて第２のマッピングファイルを生成するマッピングファイル生成手段と、
を有することを特徴とするプログラム生成装置。
【請求項２】
前記マッピングファイルテンプレート生成手段は前記マッピングファイルテンプレート生成の際に前記Ｗｅｂサービス記述言語ファイルのＵＲＩ及び、前記第１のマッピングファイルの所在情報の入力を受け付けることを特徴とする請求項１に記載のプログラム生成装置。
【請求項３】
前記マッピングファイル生成手段は、前記第１のマッピングファイルと前記マッピングファイルテンプレート生成手段で生成されたマッピングファイルテンプレートとの属性情報の一致により、ユーザ入力項目を抽出することを特徴とする、請求項１または２に記載のプログラム生成装置。
【請求項４】
Ｗｅｂサービス記述言語ファイルを有するＷｅｂサービスを提供するＷｅｂサービス提供装置、及び前記Ｗｅｂサービスの処理の代理として呼び出されるプロキシクラスファイルを介して前記Ｗｅｂサービスを利用してＷｅｂアプリケーションプログラムを実行するＷｅｂアプリサーバ装置と通信可能なプログラム生成装置におけるプログラム生成方法であって、
前記Ｗｅｂサービス提供装置の前記Ｗｅｂサービス記述言語ファイルを読み込むＷｅｂサービス記述言語ファイル読込工程と、
前記Ｗｅｂサービス記述言語ファイル読込工程により読み込まれたＷｅｂサービス記述言語ファイルを解析してＷｅｂサービス記述言語ファイルの要素を分類し、前記プロキシクラスファイルの入出力に必要な要素を解析するＷｅｂサービス記述言語ファイル解析工程と、
前記Ｗｅｂサービス記述言語ファイル解析工程により解析された要素を抽出し、抽出した情報を基に前記プロキシクラスファイルと前記Ｗｅｂアプリケーションプログラムとの入出力データモデルとの対応を記憶したマッピングファイルを作成する際のテンプレートとなるマッピングファイルテンプレートを生成するマッピングファイルテンプレート生成工程と、
前記Ｗｅｂアプリサーバ装置から既存の第１のマッピングファイルを読み込むマッピングファイル読込工程と、
前記マッピングファイル読込工程により読み込まれた第１のマッピングファイルのユーザ入力項目を抽出して、該抽出されたユーザ入力項目を前記マッピングファイルテンプレート生成工程で生成されたマッピングファイルテンプレートに適応させて第２のマッピングファイルを生成するマッピングファイル生成工程と、
を有することを特徴とするプログラム生成方法。
【請求項５】
請求項４に記載されたプログラム生成方法を前記プログラム生成装置に実行させるためのプログラム。
【請求項６】
請求項４に記載されたプログラム生成方法を前記プログラム生成装置に実行させるためのプログラムをコンピュータが読み取り可能に記憶した記憶媒体。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、コンピュータで実行するプログラムを生成するプログラム生成制御に関する。
【背景技術】

【０００２】
サーバサイドのアプリケーションを他のアプリケーションから呼び出し、利用する仕組みとして、Ｗｅｂサービスが知られている。Ｗｅｂサービスとは、ＨＴＴＰ（Ｈｙｐｅｒ Ｔｅｘｔ Ｔｒａｎｓｐｏｒｔ Ｐｒｏｔｏｃｏｌ）などのインターネット関連技術を応用して、ＳＯＡＰ（Ｓｉｍｐｌｅ Ｏｂｊｅｃｔ Ａｃｃｅｓｓ Ｐｒｏｔｏｃｏｌ）と呼ばれるＸＭＬ（Ｅｘｔｅｎｓｉｂｌｅ Ｍａｒｋｕｐ Ｌａｎｇｕａｇｅ）形式のプロトコルを用いメッセージの送受信を行うサービスのことである。
【０００３】
上記の場合、Ｗｅｂサービスを呼び出すクライアントでは、公開されているＷＳＤＬ（Ｗｅｂ Ｓｅｒｖｉｃｅｓ Ｄｅｓｃｒｉｐｔｉｏｎ Ｌａｎｇｕａｇｅ）を元に生成されたプロキシクラスファイルを利用することが多い。プロキシクラスとは、クライアントサイドにあって、サーバサイドのアプリケーションをあたかもローカルのライブラリにアクセスするのと同じ要領でアクセスできる役割を担うものである。
【０００４】
そのＷｅｂサービスの仕様が変更された場合、サーバサイドのＷＳＤＬが変更され、ＷＳＤＬから生成されるプロキシクラスを変更してクライアントサイドで利用することになる。このＷｅｂサービスの変更に対して、ＷＳＤＬの変更を検知し、プロキシクラスを自動的に変更する方法が特許文献１に記載されている。
【特許文献１】特開２００６－１９５５５５号公報
【発明の開示】
【発明が解決しようとする課題】
【０００５】
しかしながら、プロキシクラスを呼び出すＷｅｂアプリケーションプログラムについては動的に変更することが出来ず、Ｗｅｂサービスの変更に対して、Ｗｅｂアプリケーションプログラムそのものを修正する必要があるという問題がある。
【０００６】
本発明は、上記の問題点を解決するためになされたもので、本発明の目的は、Ｗｅｂサービスならびにプロキシクラスが変更された際に、Ｗｅｂアプリケーションプログラムを変更することなくＷｅｂサービスを利用するために、Ｗｅｂサービスの入出力データとＷｅｂアプリケーションプログラムの入出力データモデルの関係を記述するマッピングファイルを変更する際に既存のマッピングファイルのユーザ入力項目を反映することで、新たにユーザの入力が必要な項目を減らした新たなマッピングファイルを生成することを目的とする。
【課題を解決するための手段】
【０００７】
本発明は、Ｗｅｂサービス記述言語ファイルを有するＷｅｂサービスを提供するＷｅｂサービス提供装置、及び前記Ｗｅｂサービスの処理の代理として呼び出されるプロキシクラスファイルを介して前記Ｗｅｂサービスを利用してＷｅｂアプリケーションプログラムを実行するＷｅｂアプリサーバ装置と通信可能なプログラム生成装置であって、前記Ｗｅｂサービス提供装置の前記Ｗｅｂサービス記述言語ファイルを読み込むＷｅｂサービス記述言語ファイル読込手段（図１の１１６）と、前記Ｗｅｂサービス記述言語ファイル読込手段により読み込まれたＷｅｂサービス記述言語ファイルを解析してＷｅｂサービス記述言語ファイルの要素を分類し、前記プロキシクラスファイルの入出力に必要な要素を解析するＷｅｂサービス記述言語ファイル解析手段（図１の１０１）と、前記Ｗｅｂサービス記述言語ファイル解析手段により解析された要素を抽出し、抽出した情報を基に前記プロキシクラスファイルと前記Ｗｅｂアプリケーションプログラムとの入出力データモデルとの対応を記憶したマッピングファイルを作成する際のテンプレートとなるマッピングファイルテンプレートを生成するマッピングファイルテンプレート生成手段（図１の１０３）と、前記Ｗｅｂアプリサーバ装置から既存の第１のマッピングファイルを読み込むマッピングファイル読込手段（図１の１１７）と、前記マッピングファイル読込手段により読み

込まれた第１のマッピングファイルのユーザ入力項目を抽出して、該抽出されたユーザ入力項目を前記マッピングファイルテンプレート生成手段で生成されたマッピングファイルテンプレートに適応させて第２のマッピングファイルを生成するマッピングファイル生成手段（図１の１１９）とを有することを特徴とする。

【発明の効果】

【０００８】

本発明は、Ｗｅｂサービスならびにプロキシクラスが変更された際に、Ｗｅｂアプリケーションプログラムを変更することなくＷｅｂサービスを利用するために、Ｗｅｂサービスの入出力データとＷｅｂアプリケーションプログラムの入出力データモデルの関係を記述するマッピングファイルを変更する際に既存のマッピングファイルのユーザ入力項目を反映することで、新たにユーザの入力が必要な項目を減らした新たなマッピングファイルを生成することを可能とする効果を有する。

【図面の簡単な説明】

【０００９】

【図１】本発明の実施形態を示すプログラム生成システムの構成を示すブロック図である。

【図２】本発明のプログラム生成装置における第１の制御処理手順の一例を示すフローチャートである。

【図３】本発明のＷｅｂアプリサーバ装置における第２の制御処理手順の一例を示すフローチャートである。

【図４】本発明のプログラム生成装置およびＷｅｂサービス提供装置における第１の制御処理手順の一例を示すシーケンスチャートである。

【図５】本発明のＷｅｂサービス提供装置におけるＷＳＤＬファイルの一例である。

【図６】本発明のプログラム生成装置により生成されるマッピングファイルテンプレートにユーザがアプリケーション入出力データモデルを入力した一例である。

【図７】本発明のＷｅｂサービス提供装置におけるＷｅｂサービス変更後のＷＳＤＬファイルの一例である。

【図８】本発明のプログラム生成装置によりＷｅｂサービス変更後に生成されたマッピングファイルテンプレートにユーザがアプリケーション入出力データモデルを入力した一例である。

【図９】本発明のＷｅｂアプリサーバ装置におけるＷｅｂアプリケーションプログラムの一部を示す一例である。

【図１０】本発明の実施形態におけるプログラム生成方法を模式的の説明するための模式図である。

【図１１】本発明のＷｅｂアプリサーバ装置で生成されるＷｅｂアプリケーションプログラムの画面の模式図である。

【図１２】本発明のＷｅｂサービス提供装置の変更後にＷｅｂアプリサーバ装置で生成されるＷｅｂアプリケーションプログラムの画面の模式図である。

【図１３】本発明のプログラム生成装置におけるマッピングファイルテンプレート生成ツールの画面の模式図である。

【図１４】本発明のプログラム生成装置により生成されるプロキシクラス情報ファイルの一例である。

【図１５】本発明のＷｅｂアプリサーバ装置およびＷｅｂサービス提供装置における第２の制御処理手順の一例を示すシーケンスチャートである。

【図１６】本発明の実施形態における各種装置のハードウェア構成を示すブロック図である。

【図１７】本発明のプログラム生成装置により生成されるマッピングファイルテンプレートの一例である。

【図１８】本発明のプログラム生成装置によりＷｅｂサービス変更後に生成されたマッピングファイルテンプレートの一例である。

【図１９】本発明に係るプログラム生成装置で読み込み可能なデータ処理プログラムを格納する記憶媒体のメモリマップを説明する図である。
【図２０】本発明に係るＷｅｂアプリサーバ装置で読み込み可能なデータ処理プログラムを格納する記憶媒体のメモリマップを説明する図である。
【発明を実施するための形態】
【００１０】
以下、図面を用いて本発明のプログラム生成システムの実施形態について説明する。
【００１１】
本実施形態では、Ｗｅｂサービスを構成するＷｅｂアプリサーバ装置側のプログラム生成について説明する。なお、本実施形態では、「プログラム」とはソースプログラムを示すものとする。
【００１２】
また、Ｗｅｂサービス環境の説明は、本発明と直接関係のある部分についてのみ言及する。
【００１３】
図１は、本発明の一実施形態を示すプログラム生成システムのシステム全体を示す構成図である。
【００１４】
図１において、１００はプログラム生成装置であり、１０１はＷＳＤＬ解析部である。ＷＳＤＬ解析部１０１は、Ｗｅｂサービス提供装置１１３にあるＷＳＤＬファイル（Ｗｅｂサービス記述言語ファイル）１１５をＷＳＤＬ読込部１１６で読み込んで解析する。１０２はプロキシクラス生成部であり、ＷＳＤＬ解析部１０１の結果を受けてＷｅｂアプリサーバ装置のプロキシクラスファイル１０７を生成する。１０３はマッピングファイルテンプレート生成部であり、ＷＳＤＬ解析部１０１の結果を受けてマッピングファイルテンプレートを生成する。マッピングファイルテンプレートとは、プロキシクラス生成部１０２で生成したプロキシクラスファイル１０７と、Ｗｅｂアプリサーバ１０５にあるＷｅｂアプリケーションプログラム１０６との間でデータモデルの変換を行うためのマッピングファイルを生成するためのテンプレートである。マッピングファイルテンプレートのＷｅｂアプリケーションプログラム１０６のデータモデル情報をユーザが入力したものが１０９のマッピング済みマッピングファイルである。１０４はプロキシクラス情報ファイル生成部であり、プロキシクラスファイルの所在や名前を示すプロキシクラス情報ファイルを生成する。
【００１５】
１０５はＷｅｂアプリサーバ装置であり、Ｗｅｂサービスを利用する端末である。１０６はＷｅｂアプリケーションプログラムであり、マッピング済みマッピングファイル１０９を介して、プロキシクラスファイル１０７とあたかもＷｅｂアプリサーバ装置１０５側にＷｅｂサービスがあるかのようにＷｅｂサービス提供装置１１３のＷｅｂサービスプログラム１１４を実行する。Ｗｅｂサービス呼出処理部１０８はＷｅｂアプリケーションプログラム１０６で呼び出されるランタイムモジュールを指す。１１１はマッピング処理部であり、プロキシクラスファイル１０７とマッピング済みマッピングファイル１０９とを対応づける処理を行う。
【００１６】
Ｗｅｂサービス提供装置１１３ならびにプログラム生成装置１００、Ｗｅｂアプリサーバ装置１０５は互いにネットワーク１１２で接続されている。
【００１７】
次に、図１６を参照して、図１に示したプログラム生成装置１００およびＷｅｂアプリサーバ装置１０５、ならびにＷｅｂサービス提供装置１１３のハードウェア構成について説明する。
【００１８】
図１６は、本発明の実施形態における各種装置のハードウェア構成を示す図である。

【００１９】
ＣＰＵ１６０１は、システムバス１６０４に接続される各デバイスやコントローラを統括的に制御する。
【００２０】
また、ＲＯＭ１６０２あるいは外部メモリ１６１１には、ＣＰＵ１６０１の制御プログラムであるＢＩＯＳ（Ｂａｓｉｃ Ｉｎｐｕｔ ／ Ｏｕｔｐｕｔ Ｓｙｓｔｅｍ）やオペレーティングシステムプログラム（以下、ＯＳ）や、各サーバ或いは各ＰＣの実行する機能を実現するために必要な後述する各種プログラム等が記憶されている。ＲＡＭ１６０３は、ＣＰＵ１６０１の主メモリ、ワークエリア等として機能する。
【００２１】
ＣＰＵ１６０１は、処理の実行に際して必要なプログラム等をＲＡＭ１６０３にロードして、プログラムを実行することで各種動作を実現するものである。
【００２２】
また、入力コントローラ（入力Ｃ）１６０５は、キーボード１６０９や不図示のマウス等のポインティングデバイスからの入力を制御する。
【００２３】
ビデオコントローラ（ＶＣ）１６０６は、ＣＲＴディスプレイ（ＣＲＴ）１６１０等の表示器への表示を制御する。表示器はＣＲＴだけでなく、液晶ディスプレイでも構わない。これらは必要に応じて管理者が使用するものである。本発明には直接関係があるものではない。
【００２４】
メモリコントローラ（ＭＣ）１６０７は、ブートプログラム、ブラウザソフトウエア、各種のアプリケーション、フォントデータ、ユーザファイル、編集ファイル、各種データ等を記憶するハードディスク（ＨＤ）やフロッピーディスク（登録商標　ＦＤ）或いはＰＣＭＣＩＡカードスロットにアダプタを介して接続されるコンパクトフラッシュメモリ等の外部メモリ１６１１へのアクセスを制御する。
【００２５】
通信Ｉ／Ｆコントローラ（通信Ｉ／ＦＣ）１６０８は、ネットワークを介して、外部機器と接続・通信するものであり、ネットワークでの通信制御処理を実行する。例えば、ＴＣＰ／ＩＰを用いたインターネット通信等が可能である。
【００２６】
なお、ＣＰＵ１６０１は、例えばＲＡＭ１６０３内の表示情報用領域へアウトラインフォントの展開（ラスタライズ）処理を実行することにより、ＣＲＴ１６１０上での表示を可能としている。また、ＣＰＵ１６０１は、ＣＲＴ１６１０上の不図示のマウスカーソル等でのユーザ指示を可能とする。
【００２７】
本発明を実現するためのプログラムは外部メモリ１６１１に記録されており、必要に応じてＲＡＭ１６０３にロードされることによりＣＰＵ１６０１によって実行されるものである。さらに、本発明で生成されるプログラムやプロキシクラスファイル、プログラムが用いる定義ファイル及び各種情報テーブルは外部メモリ１６１１に格納されており、これらについての詳細な説明は後述する。
【００２８】
なお、図１の構成は一例であり、用途や目的に応じて様々な構成例があることは云うまでもない。その他の構成例として、プログラム生成装置が、ネットワークと接続され、通信可能な不図示のパーソナルコンピュータ等の情報処理装置からプログラム生成命令を受けることによって、プログラムの生成を行うことも可能である。
【００２９】
すなわち、本発明の機能が実現されるものであれば、単体の機器であっても、複数の機器からなるシステムであっても、ネットワークを介して処理が行われるシステムであっても本発明を適用することができる。

【００３０】
また、ここで、ハードディスクドライブ（ＨＤＤ）等の外部メモリ１６１１に記憶された情報は、それぞれデータベース（ＤＢ）等に格納されていても良い。
【００３１】
次に、プログラム生成装置１００の基本的な処理フローについて、図２を用いて説明する。
【００３２】
図２は、本発明の実施形態におけるプログラム生成装置１００の基本的な処理フローを示す図である。
【００３３】
上述したように、プログラム生成装置１００はＷｅｂサービス提供装置１１３とネットワーク１１２を介して相互に接続されている。そして、プログラム生成装置１００はＷｅｂサービス提供装置１１３のＷｅｂサービスプログラム１１４を利用するためにＷｅｂサービスを呼ぶプロキシクラスファイルとマッピングファイルテンプレートを生成する。その一例について、以下に説明する。
【００３４】
図２は、ユーザからのＷｅｂサービス利用プログラムを生成する指示により開始される。
【００３５】
まず、ステップＳ２００において、プログラム生成装置１００は、ネットワーク１１２を経由して、Ｗｅｂサービス提供装置１１３にあるＷＳＤＬファイル１１５にアクセスし、読み込む。
【００３６】
次にステップＳ２０１でプログラム生成装置１００は、読み込んだＷＳＤＬファイル１１５の内容を解析する。具体的にはＷＳＤＬで公開されている要素（［ｓｅｒｖｉｃｅ］、［ｐｏｒｔ］、［ｂｉｎｄｉｎｇ］、［ｐｏｒｔＴｙｐｅ］、［ｍｅｓｓａｇｅ］、［ｔｙｐｅｓ］等）を分類する。
【００３７】
ステップＳ２０２では、プログラム生成装置１００は、ＷＳＤＬファイル１１５内をＷＳＤＬに基づいて走査し、分類された要素毎に必要な情報を収集し細分化して、ＷＳＤＬ要素オブジェクトのインスタンス化を図る。そして、その結果からＷｅｂサービスの読み出しに必要なプロキシクラスファイルを生成する。
【００３８】
次にステップＳ２０３でプログラム生成装置１００は、ステップＳ２０２で生成したプロキシクラスファイルの在処やメタ情報を記録したプロキシクラス情報ファイルを生成する。生成したプロキシクラス情報ファイルの具体例を図１４に示す。
【００３９】
図１４はプロキシクラス情報ファイルの一例である。１４００はＳａｍｐｌｅＳｅｒｖｉｃｅクラスの所在を示している。図２のフローに戻る。
【００４０】
ステップＳ２０４では、プログラム生成装置１００は、Ｗｅｂサービス提供装置１１３から読み込んだＷＳＤＬファイル１１５からマッピングファイルテンプレートを生成する。
図５と図１７によりマッピングファイルテンプレート生成の例を示す。
【００４１】
図５は商品検索サービスを提供するＷｅｂサービスのＷＳＤＬファイルの例である。
【００４２】
また、図１７は生成されたマッピングファイルテンプレートの例である。
【００４３】
図５のＷＳＤＬファイルによると、Ｗｅｂサービスで呼び出されるＳａｍｐｌｅＳｅｒ

ｖｉｃｅサービスのエンドポイント（Ｗｅｂサービスを提供しているＵＲＬ）は５０１にあるように“ｈｔｔｐ：／／ｌｏｃａｌｈｏｓｔ：８０８２／ｓａｍｐｌｅ”なので、図１７の１７００のようにマッピングファイルテンプレートに出力する。

【００４４】

また、ＳａｍｐｌｅＳｅｒｖｉｃｅサービスのｓｅａｒｃｈＳｈｏｈｉｎオペレーションへの入力データモデルとしては５０２にあるように、ｎａｍｅ＝“ｃｏｄｅ”、ｔｙｐｅ＝“ｘｓ：ｓｔｒｉｎｇ”と指定されているので、図１７の１７０１のように、ｎａｍｅ＝“ｃｏｄｅ”、ｔｙｐｅ＝“ｓｔｒｉｎｇ”と出力する。

【００４５】

同様にｓｅａｒｃｈＳｈｏｈｉｎオペレーションからの出力データモデルは５０３にあるように、ｎａｍｅ＝“ｓｈｏｈｉｎ”、ｔｙｐｅ＝“ｔｎｓ：ｓｈｏｈｉｎ”と指定されており、“ｓｈｏｈｉｎ”というユーザ定義のデータモデルとして、“ｃｏｄｅ”、“ｄｅｓｃｒｉｐｔｉｏｎ１”、“ｎａｍｅ”、“ｐｒｉｃｅ”のそれぞれの名前を持つ要素を持つ。そのため、マッピングファイルテンプレートには、図１７の１７０２のように出力する。

【００４６】

なお、図１７のマッピングファイルテンプレートに大文字で記載されているＤＭＣＯＤＥやＤＭＩＴＥＭ、ＯＵＴＰＡＲＡＭの属性値は、Ｗｅｂアプリケーションプログラムの入出力のデータモデルなので、マッピングファイルテンプレート生成時には値が設定されていない。図２のフローに戻る。

【００４７】

ステップＳ２０５において、プログラム生成装置１００は、このフロー以前に記憶されたマッピングファイルがあるかを判断する。以前に記憶されたマッピングファイルがない場合はこのフローを終了する。以前に記憶されたマッピングファイルがある場合は、ステップＳ２０６へと処理を進める。

【００４８】

以前に記憶されたマッピングファイルがある場合、ステップＳ２０６において、プログラム生成装置１００は、以前に作成したマッピングファイルを読み込んで、ステップＳ２０４で作成したマッピングファイルテンプレートに対応するデータモデルの属性値をマッピングする。なお、新しいマッピングファイルテンプレートのファイル名は以前に記憶されたマッピングファイルと同じ（即ち、上書き）でも構わないし、新たに名前を変えて保存しても構わない。新たに名前を変えて保存した場合は、Ｗｅｂアプリケーションプログラムを変更しないために、マッピングしたマッピングファイルのファイル名を以前に記憶されたマッピングファイルと同じにする。

【００４９】

具体的に以前に記憶されたマッピングファイルがある場合の例を次に示す。

【００５０】

以下は、図５のＷＳＤＬファイルが図７のＷＳＤＬファイルに変わった場合の例である。

【００５１】

Ｓ２０４で作成した図１７のマッピングファイルテンプレートに開発者がＷｅｂアプリケーションプログラムのデータモデルを元にマッピングした例が図６のマッピングファイルである。

【００５２】

図６では、６０３の欄に開発者により“Ｓｈｏｈｉｎ”、“Ｃｏｄｅ”、“Ｄｅｓｃｒｉｐｔｉｏｎ１”、“Ｎａｍｅ”、“Ｐｒｉｃｅ”などが入力される。

【００５３】

図６のマッピングファイルをＷｅｂアプリケーションプログラムで実装したユーザインタフェイスの例が図１１である。図１１の場合、将来的にＷｅｂサービスの商品説明（６０３の“Ｄｅｓｃｒｉｐｔｉｏｎ”属性）が追加されることが分かっているため、商品説

明１フィールド１１００（“Ｄｅｓｃｒｉｐｔｉｏｎ１”属性）の他に商品説明２フィールド１１０１も予め作成されている。但し、Ｗｅｂサービスによる出力はまだないため、データそのものは取得できない状態である。

【００５４】

図７は図５に示す商品検索サービスが変更された結果、ＷＳＤＬファイルが変更された例である。

【００５５】

図７の７００に示すように、変更前には図５の５００の“ｓｈｏｈｉｎ”であった出力データの名前（ｎａｍｅ属性）は商品検索サービスの変更に伴い、“ｐｒｏｄｕｃｔ”に変更されている。

【００５６】

また、出力データを構成する要素には７０１のように“ｄｅｓｃｒｉｐｔｉｏｎ２”が追記されている。

【００５７】

さらに、当初、Ｗｅｂサービスのエンドポイント５０１は“ｈｔｔｐ：／／ｌｏｃａｌｈｏｓｔ：８０８２／ｓａｍｐｌｅ”を指しており、Ｗｅｂアプリサーバ装置自身にＷｅｂサービスを実装して開発していたが、図７のＷＳＤＬを持つＷｅｂサービスに変更されると、Ｗｅｂサービスのエンドポイント７０２は“ｈｔｔｐ：／／ｒｅａｌｓｅｒｖｅｒ／ｓａｍｐｌｅ”に変わり、実際のＷｅｂサービス提供装置を指している。

【００５８】

図１８は商品検索サービスが図５のＷＳＤＬから図７のように変更されたＷＳＤＬを読み込み生成されたマッピングファイルテンプレートの例である。マッピングファイルテンプレートを作成する際に前回作成したマッピングファイル（この場合、図６）を読み込む。

【００５９】

図１８の１８０４には、図７の７０２を参照してＷｅｂサービスのエンドポイントを“ｈｔｔｐ：／／ｒｅａｌｓｅｒｖｅｒ／ｓａｍｐｌｅ”に設定している。また、１８０１には図７の７００に示すように出力データの属性を“ｐｒｏｄｕｃｔ”と設定している。

【００６０】

さらに、マッピングファイルテンプレート１８００を作成する際に読み込んだマッピングファイル６０２と比較し、ユーザにより入力された属性値である６０３を抽出し、該当する属性に入力する（１８０２）。６０３を抽出する判断は、例えば６０５の属性名（ｎａｍｅ）と属性型（ｔｙｐｅ）が一致している要素の場合や、属性型（ｔｙｐｅ）が一致している要素の場合などがある。１８０３には追加された出力データを構成する要素として、図７の７０１を反映した要素が追加される。

【００６１】

ユーザは図１８の追加された出力データの要素１８０３の属性値として、“Ｓｈｏｈｉｎ”や“Ｄｅｓｃｒｉｐｔｉｏｎ２”を入力するのみで、Ｗｅｂサービスの変更に対応できる。ユーザの入力後のマッピングファイルの例が図８である。図８の８０１がユーザにより入力された属性値である。

【００６２】

図８のマッピングファイルをＷｅｂアプリケーションプログラムで実装したユーザインタフェイスの例が図１２である。図１２は、商品説明２フィールドを予め作っていたため、Ｗｅｂサービスの変更に伴い追加された商品説明（１８０３の要素）を１２００のように表示可能になる。

図１０により、本発明を模式的に説明する。

【００６３】

Ｗｅｂサービスの更新により、図１０のようにＷｅｂサービス１（１１４－１）から、Ｗｅｂサービス２（１１４－２）へ更新された場合、それに伴いＷＳＤＬファイルもＷＳＤＬファイル１（１１５－１）からＷＳＤＬファイル（１１５－２）に更新される。

【００６４】
ＷＳＤＬファイル１（１１５－１）が図５に該当し、ＷＳＤＬファイル２（１１５－２）が図７に該当する。図５と図７の場合、５００にある出力データの名前の“ｓｈｏｈｉｎ”が、７００にある“ｐｒｏｄｕｃｔ”に変更される。また、Ｗｅｂサービスのエンドポイント５０１は“ｈｔｔｐ：／／ｌｏｃａｌｈｏｓｔ：８０８２／ｓａｍｐｌｅ”を指しており、当初はＷｅｂアプリサーバ装置自身にＷｅｂサービスを実装して開発しているが、図７のＷＳＤＬを持つＷｅｂサービスに変更されると、Ｗｅｂサービスのエンドポイント７０２は“ｈｔｔｐ：／／ｒｅａｌｓｅｒｖｅｒ／ｓａｍｐｌｅ”に変わり、実際のＷｅｂサービス提供装置を指す。また、当初はなかった“ｄｅｓｃｒｉｐｔｉｏｎ２”という名前の項目が図７のＷＳＤＬを持つＷｅｂサービスには追加されている（７０１）。
【００６５】
ＷＳＤＬファイルが更新されていると、マッピングファイル生成部１９０２内にあるマッピングファイルテンプレート生成部が、Ｗｅｂサービス１（１１４－１）に対応したマッピングファイル１（１０９－１）を読み込み、ＷＳＤＬファイル２（１１５－２）と共にマッピングファイルテンプレートを生成する。その後、ユーザの入力などによりマッピングファイル２（１０９－２）を作成し、Ｗｅｂアプリサーバ装置１０５に送信する。
【００６６】
図２のフローチャート終了後、ユーザによってマッピングファイルテンプレートが修正され、プログラム生成装置１００はＷｅｂアプリサーバ装置１０５にプロキシクラスファイル１０７やプロキシクラス情報ファイル１１０、マッピングファイル１０９、別途生成したＷｅｂアプリケーションプログラム１０６を送信し、Ｗｅｂアプリサーバ装置の外部から利用可能な状態にする（デプロイする）。
【００６７】
図２のフローチャートをシーケンスチャートで示したのが図１５である。各生成部やＷＳＤＬ解析部はプログラム生成装置内１００に配置され、ＷＳＤＬはＷｅｂサービス提供装置１１３に配置する。なお、図２の各ステップと同じ処理をしているステップに同じステップ番号を振っている。
【００６８】
次に、Ｗｅｂアプリサーバ装置１０５の基本的な処理フローについて、図３と図４を用いて説明する。
【００６９】
図３は、本発明の実施形態におけるＷｅｂアプリサーバ装置１０５の基本的な処理フローを示す図である。
【００７０】
また、図４は、本発明の実施形態におけるＷｅｂアプリサーバ装置１０５とＷｅｂサービス提供装置１１３との基本的なシーケンスチャートを示す図である。なお、図番号やステップ番号が共通の処理、構成に関しては図１や図３、図４と同じ図番号やステップ番号を振っている。
【００７１】
図４のクライアント部１０５、マッピングファイル１０９、プロキシクラス情報ファイル１１０、Ｗｅｂサービス呼出処理部１０８はＷｅｂアプリサーバ装置に配置され、Ｗｅｂサービス部１１４はＷｅｂサービス提供装置に配置する
【００７２】
図３、図４は、図示しないユーザ端末からＷｅｂアプリサーバ装置１０５にＷｅｂアプリケーションプログラム１０６を実行する指示があり、Ｗｅｂアプリケーションプログラム１０６がＷｅｂサービスプログラム１１４を実行する指示があった際に開始される。
【００７３】
まずステップＳ３００において、Ｗｅｂアプリサーバ装置１０５は、Ｗｅｂアプリケーションプログラム１０６からＷｅｂサービス呼出命令を受ける。Ｗｅｂアプリケーションプログラム１０６の一部分の一例を図９に示す。

【００７４】
図９はＷｅｂアプリケーションプログラム１０６内のＷｅｂサービスプログラム１１４を呼ぶプログラムの一例である。
【００７５】
図９の例では、９００や９０１に記載されているように、ＳａｍｐｌｅＳｅｒｖｉｃｅクラスのｓｅａｒｃｈＳｈｏｈｉｎオペレーションが呼び出される。図３、図４の説明に戻る。
【００７６】
図４のステップＳ４０１では、Ｗｅｂアプリサーバ装置１０５は、マッピングファイルをオブジェクトとして読み込むためにＣｒｅａｔｅしている。
【００７７】
同様にステップＳ４０２では、Ｗｅｂアプリサーバ装置１０５は、プロキシクラス情報ファイルをオブジェクトとして読み込むためにＣｒｅａｔｅしている。
【００７８】
ステップＳ３０１において、Ｗｅｂアプリサーバ装置１０５は図２のフローの結果デプロイされたマッピングファイル１０９を読み込む。
【００７９】
ステップＳ３０２において、Ｗｅｂアプリサーバ装置１０５は図２のフローの結果デプロイされたプロキシクラス情報ファイル１１０を読み込んでステップＳ３０３に処理を進める。プロキシクラス情報ファイルには、プロキシファイルの位置や名前などのメタ情報が記憶されており、ステップＳ３０３の処理でプロキシクラスファイルの在処を指定する。
【００８０】
ステップＳ３０３において、Ｗｅｂアプリサーバ装置１０５はＷｅｂアプリケーションプログラム１０６から指定されたＷｅｂサービスを実行するプロキシクラスファイルを呼ぶためにＷｅｂサービス呼出処理部１０８にステップＳ３０２で指定されたプロキシクラスファイル１０７が読み込まれる。
【００８１】
Ｗｅｂアプリサーバ装置１０５は、プロキシクラスファイル１０７を呼ぶ際の引数のデータモデルの詳細をステップＳ３０１で読み込んだマッピングファイル１０９から取得する。
【００８２】
その後、Ｗｅｂアプリサーバ装置１０５は、Ｗｅｂアプリケーションプログラム１０６がユーザもしくはシステムから入力された引数をプロキシクラスファイル１０７の入力パラメータとしてＷｅｂサービス呼出処理部に渡す。
【００８３】
ステップＳ４０３において、Ｗｅｂアプリサーバ装置１０５のＷｅｂサービス呼出処理部１０８は、プロキシクラスファイル１０７を実行して、Ｗｅｂサービスプログラム１１４を呼び出す。Ｗｅｂサービス呼出処理部１０８は、Ｓ３０３で得られた入力パラメータをＳＯＡＰメッセージに変換してＷｅｂサービスプログラムへ送信する。Ｗｅｂサービス呼出処理部１０８はＷｅｂサービスプログラムを呼び出した結果、返ってくる戻り値をステップＳ３０３の戻り値とする。
【００８４】
ステップＳ４０４において、Ｗｅｂアプリサーバ装置１０５は、Ｗｅｂサービス呼出処理部１０８からの戻り値のデータモデルの詳細をマッピングファイル１０９から取得する。
【００８５】
ステップＳ３０４において、Ｗｅｂアプリサーバ装置１０５は、ステップＳ３０３で呼び出したＷｅｂサービス呼出処理部１０８の結果を受け取り、ステップＳ４０４で取得したマッピングファイル１０９のデータモデルに適応させて、Ｗｅｂアプリケーションプロ

グラム１０６にデータを返す。
【００８６】
これらの構成により、Ｗｅｂサービスの仕様が変更になった場合にＷｅｂアプリケーションプログラムを変更することなく、マッピングファイルを変更するだけで、Ｗｅｂサービスの仕様変更に対応できる。
【００８７】
図１３はマッピングファイルテンプレート生成のためのグラフィカルユーザインタフェースの一例である。ＷＳＤＬ　ＵＲＩ入力フィールド１３００は、開発者がＷＳＤＬのＵＲＩやＷＳＤＬのファイルのパスを入力するためのフィールドである。ＷＳＤＬ読込ボタン１３０１を押すと、ＷＳＤＬ　ＵＲＩ入力フィールド１３００で指定されたＷＳＤＬを読み込み、ＷＳＤＬに記述されているサービス名およびポート名のリストをそれぞれ、サービス名選択フィールド１３０２、ポート名選択フィールド１３０３に選択肢として表示する。マッピングファイルテンプレート実行ボタン１３０４を押すことで、マッピングファイルテンプレートの生成を行う。
【００８８】
図１０は本実施例を模式的に説明した図である。なお、図１と共通の構成には、同じ符号を付している。
【００８９】
図１０のプロキシクラスファイル１（１０７－１）はＷＳＤＬファイル１（１１５－１）から生成されるＷｅｂアプリサーバ１０５上のＷｅｂサービス１（１１４－１）のプロキシクラスファイルである。Ｗｅｂアプリプログラム１０６がプロキシクラスファイルを呼ぶ際に、Ｗｅｂアプリサーバ１０５は、同じくＷＳＤＬファイル１（１１５－１）から作成したマッピングファイル１（１０９－１）を読み込んで、Ｗｅｂサービスのエンドポイントを指定し、Ｗｅｂサービス側（１１４－１）の入出力データ名およびデータ型などをＷｅｂアプリプログラム１０６の入出力データモデルに対応させる。
【００９０】
図１０において、Ｗｅｂサービス１（１１４－１）の仕様が変更になった場合、変更後のＷｅｂサービス２（１１４－２）のＷＳＤＬファイル２（１１５－２）により、プロキシクラス生成部１０２がプロキシクラスファイル２（１０７－２）を生成する。また、マッピングファイル生成部１９０２は、ＷＳＤＬファイル２（１１５－２）とマッピングファイル１（１０９－１）から、マッピングファイルテンプレートを生成し、プログラム生成装置でのユーザ入力により、マッピングファイル２（１０９－２）を生成する。
【００９１】
ＷｅｂアプリサーバはＷｅｂアプリプログラム１０６がプロキシクラスファイルを呼ぶ際に、マッピングファイル２（１０９－２）を読み込んで、Ｗｅｂサービスのエンドポイントを指定し、Ｗｅｂサービス側（１１４－２）の入出力データ名およびデータ型などをＷｅｂアプリプログラム１０６の入出力データモデルに対応させる。
【００９２】
以上のように、本発明ではマッピングファイルテンプレートの変更とプロキシクラスファイルの更新のみで、Ｗｅｂアプリケーションプログラムの修正やコンパイルを必要とせず、Ｗｅｂサービスの変更に対応するという効果を有する。
【００９３】
なお、マッピングファイルテンプレートに修正の必要がなければ、そのままマッピングファイルとして適応することもできる。
【００９４】
また、前記実施形態ではプログラム生成装置にて任意の時点でマッピングファイルを生成しているが、ＷＳＤＬファイルの変更をネットワーク１１２経由で定期的に確認し、自動的にマッピングファイルを生成しても良い。
【００９５】
さらに、マッピングファイルはファイルでなくても良く、マッピングファイルに記述さ

れているＸＭＬデータをリレーショナルデータベースやＸＭＬデータベースに格納し、データアクセスコマンドからアクセスしても良い。

【００９６】

なお、上述した各種データの構成及びその内容はこれに限定されるものではなく、用途や目的に応じて、様々な構成や内容で構成されることは言うまでもない。

【００９７】

以上、一実施形態について示したが、本発明は、例えば、システム、装置、方法、プログラムもしくは記録媒体等としての実施態様をとることが可能であり、具体的には、複数の機器から構成されるシステムに適用しても良いし、また、一つの機器からなる装置に適用しても良い。

【００９８】

以下、図１９、図２０に示すメモリマップを参照して本発明に係るプログラム生成装置およびＷｅｂアプリサーバ装置（共にコンピュータ）で読み取り可能なデータ処理プログラムの構成について説明する。

【００９９】

図１９は、本発明に係るプログラム生成装置（コンピュータ）で読み取り可能な各種データ処理プログラムを格納する記録媒体（記憶媒体）のメモリマップを説明する図である。

【０１００】

また、図２０は、本発明に係るＷｅｂアプリサーバ装置（コンピュータ）で読み取り可能な各種データ処理プログラムを格納する記録媒体（記憶媒体）のメモリマップを説明する図である。

【０１０１】

なお、特に図示しないが、記録媒体に記憶されるプログラム群を管理する情報、例えばバージョン情報，作成者等も記憶され、かつ、プログラム読み出し側のＯＳ等に依存する情報、例えばプログラムを識別表示するアイコン等も記憶される場合もある。

【０１０２】

さらに、各種プログラムに従属するデータも上記ディレクトリに管理されている。また、インストールするプログラムやデータが圧縮されている場合に、解凍するプログラム等も記憶される場合もある。

【０１０３】

本実施形態における機能が外部からインストールされるプログラムによって、ホストコンピュータにより遂行されていてもよい。そして、その場合、ＣＤ－ＲＯＭやフラッシュメモリやＦＤ等の記録媒体により、あるいはネットワークを介して外部の記録媒体から、プログラムを含む情報群を出力装置に供給される場合でも本発明は適用されるものである。

【０１０４】

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

【０１０５】

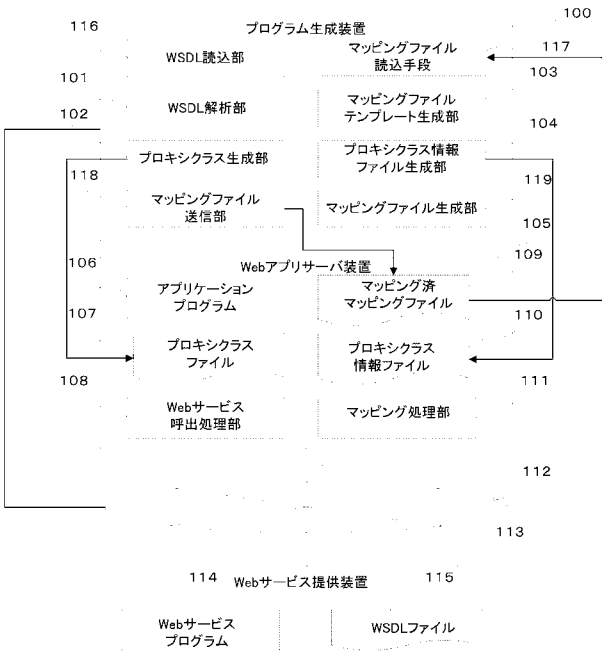
この場合、記録媒体から読み出されたプログラムコード自体が本発明の新規な機能を実現することになり、そのプログラムコードを記憶した記録媒体は本発明を構成することになる。

【０１０６】

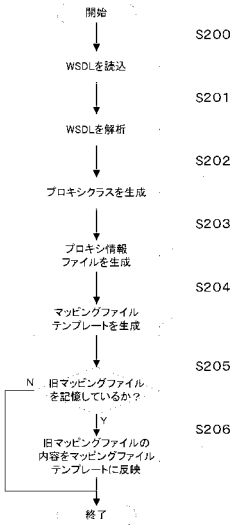
プログラムコードを供給するための記録媒体としては、例えば、フレキシブルディスク，ハードディスク，光ディスク，光磁気ディスク，ＣＤ－ＲＯＭ，ＣＤ－Ｒ，ＤＶＤ－ＲＯＭ，磁気テープ，不揮発性のメモリカード，ＲＯＭ，ＥＥＰＲＯＭ，シリコンディスク等を用いることができる。

【０１０７】
また、コンピュータが読み出したプログラムコードを実行することにより、前述した実施形態の機能が実現されるだけでなく、そのプログラムコードの指示に基づき、コンピュータ上で稼働しているＯＳ（オペレーティングシステム）等が実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。
【０１０８】
さらに、記録媒体から読み出されたプログラムコードが、コンピュータに挿入された機能拡張ボードやコンピュータに接続された機能拡張ユニットに備わるメモリに書き込まれた後、そのプログラムコードの指示に基づき、その機能拡張ボードや機能拡張ユニットに備わるＣＰＵ等が実際の処理の一部または全部を行い、その処理によって前述した実施形態の機能が実現される場合も含まれることは言うまでもない。
【０１０９】
また、本発明は、複数の機器から構成されるシステムに適用しても、１つの機器からなる装置に適用してもよい。また、本発明は、システムあるいは装置にプログラムを供給することによって達成される場合にも適応できることは言うまでもない。この場合、本発明を達成するためのソフトウェアによって表されるプログラムを格納した記録媒体を該システムあるいは装置に読み出すことによって、そのシステムあるいは装置が、本発明の効果を享受することが可能となる。
【０１１０】
さらに、本発明を達成するためのソフトウェアによって表されるプログラムをネットワーク上のサーバ，データベース等から通信プログラムによりダウンロードして読み出すことによって、そのシステムあるいは装置が、本発明の効果を享受することが可能となる。
【０１１１】
なお、上述した各実施形態およびその変形例を組み合わせた構成も全て本発明に含まれるものである。
【符号の説明】
【０１１２】
１００　プログラム生成装置
１０１　ＷＳＤＬ解析部
１０２　プロキシクラス生成部
１０３　マッピングファイルテンプレート生成部
１０４　プロキシクラス情報ファイル
１０５　Ｗｅｂアプリサーバ装置
１０６　Ｗｅｂアプリケーションプログラム
１０７　プロキシクラス
１０８　Ｗｅｂサービス呼出処理部
１０９　マッピング済マッピングファイル
１１０　プロキシクラス情報ファイル
１１１　マッピング処理部
１１２　ネットワーク
１１３　Ｗｅｂサービス提供装置
１１４　Ｗｅｂサービスプログラム
１１５　ＷＳＤＬファイル
１１６　ＷＳＤＬ読込部
１１７　マッピングファイル読込部
１１８　マッピングファイル送信部

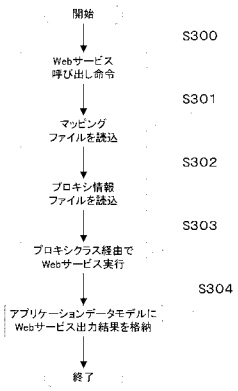
【図１】


100
116
プログラム生成装置
WSDL読込部
マッピングファイル
読込手段
117
101
103
WSDL解析部
マッピングファイル
テンプレート生成部
102
104
プロキシクラス生成部
プロキシクラス情報
ファイル生成部
118
119
マッピングファイル
送信部
マッピングファイル生成部
105
106
109
Webアプリサーバ装置
アプリケーション
プログラム
マッピング済
マッピングファイル
107
110
プロキシクラス
ファイル
プロキシクラス
情報ファイル
108
111
Webサービス
呼出処理部
マッピング処理部
112
113
114
Webサービス提供装置
115
Webサービス
プログラム
WSDLファイル

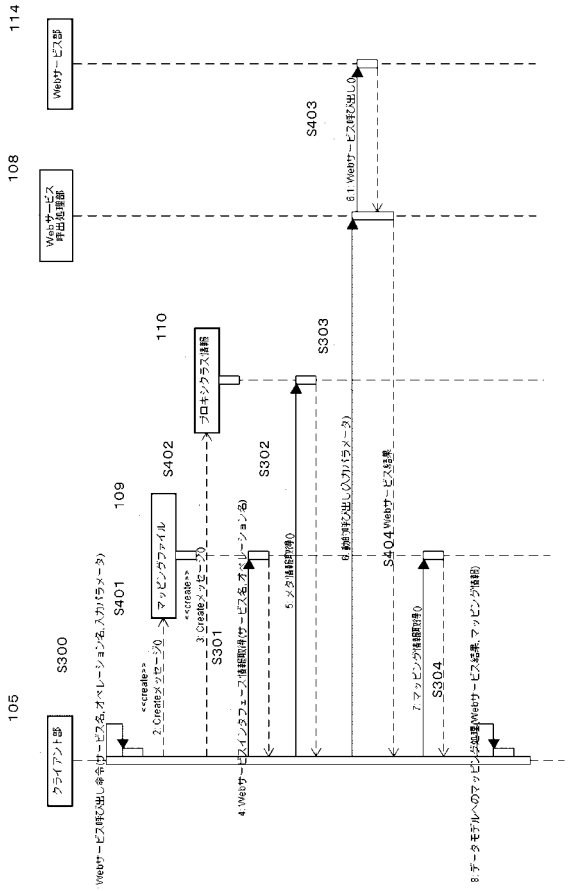
【図２】


開始
S200
WSDLを読込
S201
WSDLを解析
S202
プロキシクラスを生成
S203
プロキシ情報
ファイルを生成
S204
マッピングファイル
テンプレートを生成
S205
N
旧マッピングファイル
を記憶しているか？
Y
S206
旧マッピングファイルの
内容をマッピングファイル
テンプレートに反映
終了

【図３】

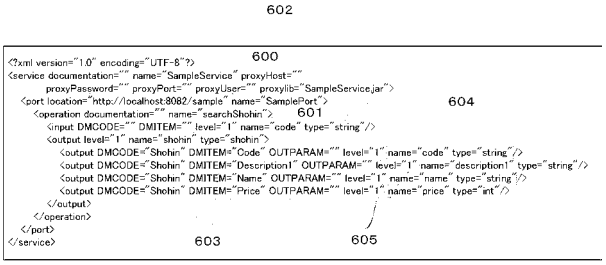

開始
S300
Webサービス
呼び出し命令
S301
マッピング
ファイルを読込
S302
プロキシ情報
ファイルを読込
S303
プロキシクラス経由で
Webサービス実行
S304
アプリケーションデータモデルに
Webサービス出力結果を格納
終了

【図４】


114
Webサービス部
S403
8.1 Webサービス呼び出し
108
Webサービス
呼出処理部
110
プロキシクラス情報
S303
S402
S302
109
マッピングファイル
S401
S301
S300
105
クライアント部
1 Webサービス呼び出し命令(サービス名,オペレーション名,入力パラメータ)
2 Createメッセージ
«create»
«create»
3 Createメッセージ
4 Webサービス(インタフェース情報取得(サービス名,オペレーション名)
5 メタ情報取得()
6 Webサービス呼び出し(入力パラメータ)
S404 Webサービス結果
7 マッピング情報取得()
S304
8 データモデルへのアッピング処理(Webサービス結果,マッピング情報)

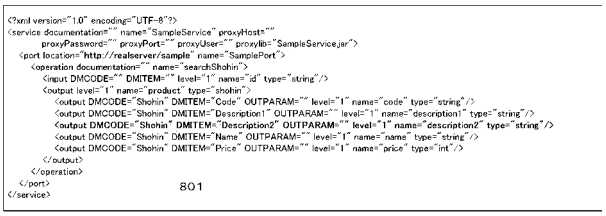
【図５】

【図６】



【図７】

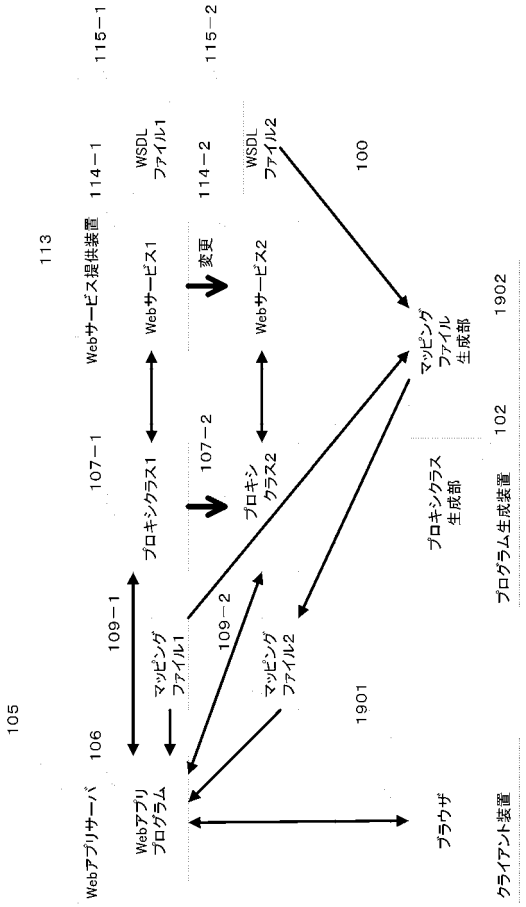
【図８】



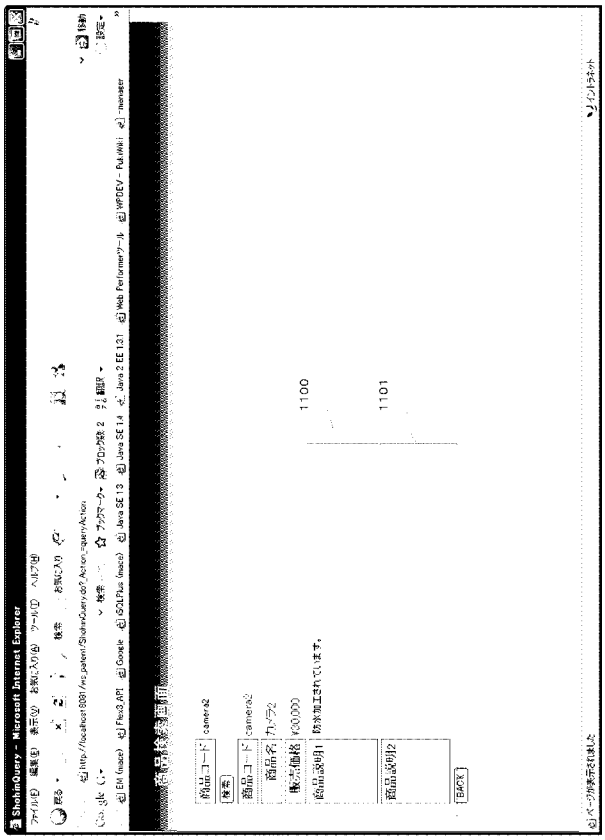
【図９】

```
package service;

import java.util.Collection;
import java.util.List;

import jp.co.canon_soft.wp.runtime.DispDataQuerier;
import jp.co.canon_soft.wp.runtime.webservice.WebService;
import jp.co.canon_soft.wp.runtime.webservice.WebServiceFactory;

public class ShohinService extends DispDataQuerier {                    900    901

    public Collection queryDispData(List params) {
        WebService ws = WebServiceFactory.createWebService("SampleService.xml", "searchShohin");
        ws.setParameters(params);
        ws.exec();
        return ws.getDMList();
    }
}
```

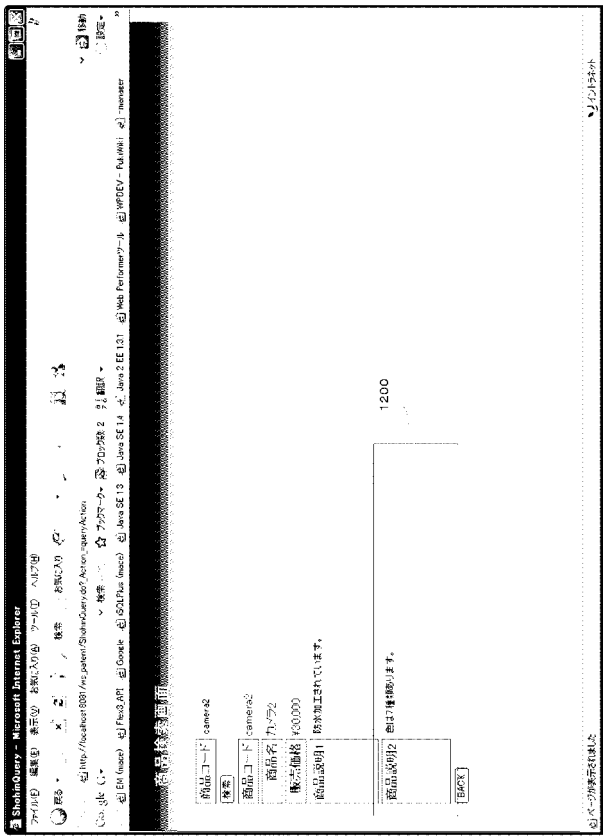
【図１０】



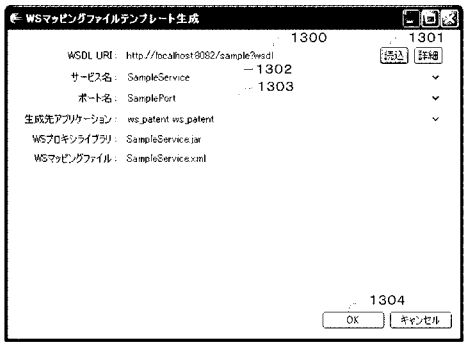
【図１１】



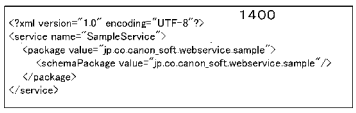
【図１２】



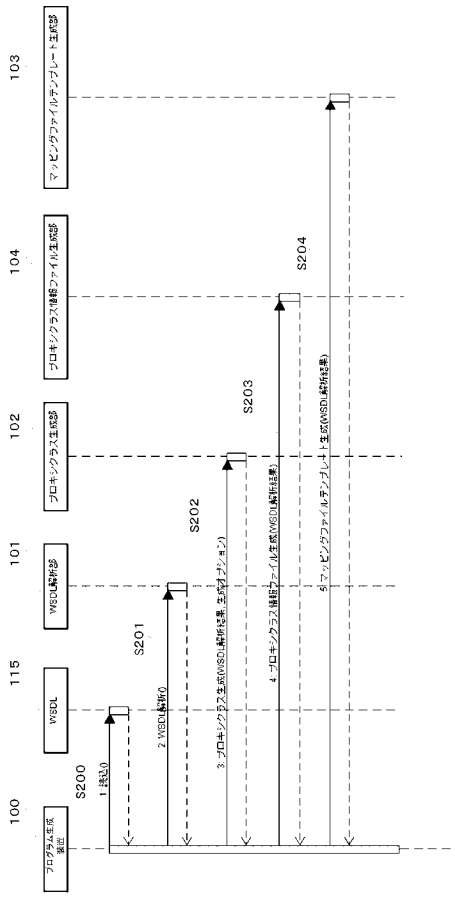
【図１３】



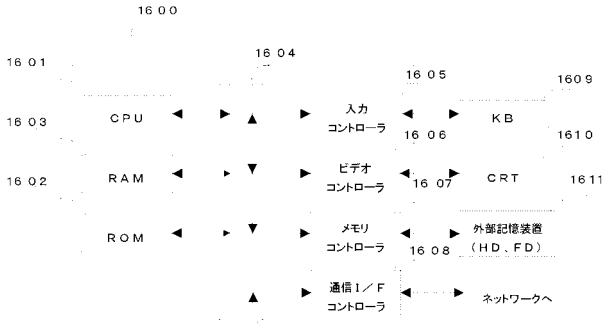
【図１４】



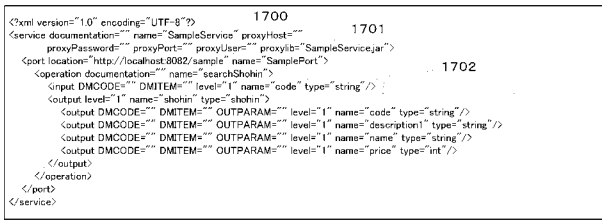
【図１５】



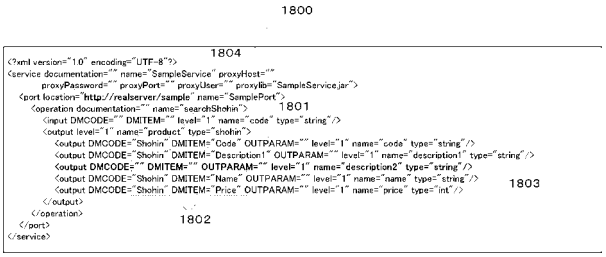
【図１６】



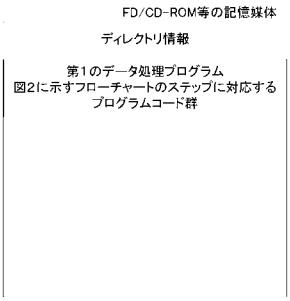
【図１７】



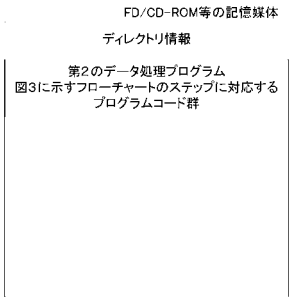
【図１８】



【図１９】



【図２０】



フロントページの続き

(72)発明者　柴本　文洋
東京都港区三田３丁目９番７号　キヤノンソフトウェア株式会社内
(72)発明者　鶴野　貴信
東京都港区三田３丁目９番７号　キヤノンソフトウェア株式会社内
(72)発明者　上田　勲
東京都港区三田３丁目９番７号　キヤノンソフトウェア株式会社内
(72)発明者　山田　真穂
東京都港区三田３丁目９番７号　キヤノンソフトウェア株式会社内

審査官　稲垣　良一

(56)参考文献　特開２００６－１９５５５５（ＪＰ，Ａ）
特開２００８－２８７３６５（ＪＰ，Ａ）
特開平８－３１４７０６（ＪＰ，Ａ）
特開２０００－２２２１９５（ＪＰ，Ａ）
特開２００８－１３４９０６（ＪＰ，Ａ）
特開２００４－３６２１８３（ＪＰ，Ａ）

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ　　９／４４