(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2010-44488
(P2010-44488A)
(43)公開日　平成22年2月25日(2010.2.25)

(51)Int.Cl.	F I	テーマコード（参考）
G06F 9/50 (2006.01)	G06F 9/46 465A	

審査請求 未請求　請求項の数 1　ＯＬ　（全 16 頁）

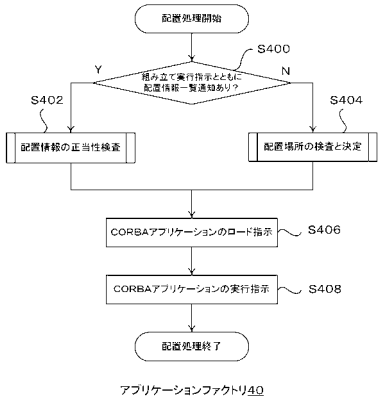
(21)出願番号　特願2008-206796 (P2008-206796)
(22)出願日　平成20年8月11日 (2008.8.11)

(71)出願人　000001122
株式会社日立国際電気
東京都千代田区外神田四丁目１４番１号
(74)代理人　110000039
特許業務法人アイ・ピー・エス
(72)発明者　鈴木　道奉
東京都羽村市神明台二丁目１番１号　株式会社日立国際電気内
(72)発明者　金橋　祐輔
東京都羽村市神明台二丁目１番１号　株式会社日立国際電気内

(54)【発明の名称】分散処理プログラム

(57)【要約】
【課題】複数のアプリケーションソフトウェアそれぞれが、複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報を決定する。
【解決手段】本発明に係る分散処理プログラムは、複数のアプリケーションソフトウェアを、複数のソフトウェア実行環境に配置する分散処理プログラムであって、複数のアプリケーションソフトウェアそれぞれが、複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報の一覧が通知されたときに、配置情報の一覧が正しいか否かを判定する判定ステップと、前記配置情報の一覧が通知されたとき以外に、複数のアプリケーションソフトウェアすべてに対して、複数のソフトウェア実行環境のいずれかに配置可能か否かを判定する判定ステップと、判定ステップの結果に基づいて、配置可能な複数のアプリケーションソフトウェアの配置情報を決定する決定ステップとをコンピュータに実行させる。
【選択図】図３

(A)



アプリケーションファクトリ40

(B)

	実行対象CORBAアプリケーションのID	配置場所に常駐するGPPDeviceのID
1	ID = AAAA	ID = 1111
2	ID = BBBB	ID = 2222
3	ID = CCCC	ID = 3333
4	ID = DDDD	ID = 4444
5	ID = EEEE	ID = 5555
6	ID = FFFF	ID = 6666
7	ID = GGGG	ID = 8888

【特許請求の範囲】
【請求項１】
複数のアプリケーションソフトウェアを、複数のソフトウェア実行環境に配置する分散処理プログラムであって、
前記複数のアプリケーションソフトウェアそれぞれが、前記複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報の一覧が通知されたときに、前記複数のアプリケーションソフトウェアの有無および前記複数のソフトウェア実行環境の有無またはこれらのいずれかを示す情報に基づいて、前記配置情報の一覧が正しいか否かを判定する判定ステップと、
前記配置情報の一覧が通知されたとき以外に、前記複数のアプリケーションソフトウェアすべてに対して、前記複数のソフトウェア実行環境のいずれかに配置可能か否かを判定する判定ステップと、
前記判定ステップの結果に基づいて、配置可能な複数のアプリケーションソフトウェアの配置情報を決定する決定ステップと
をコンピュータに実行させる分散処理プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、複数のアプリケーションソフトウェアを、複数のソフトウェア実行環境に配置する分散処理プログラムに関する。
【背景技術】
【０００２】
たとえば、特許文献１は、アプリケーションを複数のオブジェクトの組合せにより構築する分散処理プログラムを開示する。
【特許文献１】特開２００６－３０９５３３号公報
【発明の開示】
【発明が解決しようとする課題】
【０００３】
本発明は、複数のアプリケーションソフトウェアそれぞれが、複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報を決定することができるように改良された分散処理プログラムを提供する。
【課題を解決するための手段】
【０００４】
上記目的を達成するために、本発明に係る分散処理プログラムは、複数のアプリケーションソフトウェアを、複数のソフトウェア実行環境に配置する分散処理プログラムであって、前記複数のアプリケーションソフトウェアそれぞれが、前記複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報の一覧が通知されたときに、前記複数のアプリケーションソフトウェアの有無および前記複数のソフトウェア実行環境の有無またはこれらのいずれかを示す情報に基づいて、前記配置情報の一覧が正しいか否かを判定する判定ステップと、前記配置情報の一覧が通知されたとき以外に、前記複数のアプリケーションソフトウェアすべてに対して、前記複数のソフトウェア実行環境のいずれかに配置可能か否かを判定する判定ステップと、前記判定ステップの結果に基づいて、配置可能な複数のアプリケーションソフトウェアの配置情報を決定する決定ステップとをコンピュータに実行させる。
【発明の効果】
【０００５】
本発明に係る分散処理プログラムによれば、複数のアプリケーションソフトウェアそれぞれが、複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報を決定することができる。
【発明を実施するための最良の形態】

【０００６】
［本発明の背景］
本発明の理解を助けるために、まず、本発明がなされるに至った背景を説明する。
アプリケーションソフトウェアを実行するためには、アプリケーションソフトウェアのプロパティに応じて、アプリケーションソフトウェアを、適切なソフトウェア実行環境に配置する必要がある。
アプリケーションソフトウェアのプロパティとは、そのアプリケーションソフトウェアが、どのＯＳ上で、どのプロセッサ上で実行可能であるかを示す情報などである。
適切なソフトウェア実行環境とは、その実行環境のＯＳやプロセッサ情報などを示すプロパティと、アプリケーションソフトウェアのプロパティとが合致するような実行環境である。
アプリケーションソフトウェアを適切なソフトウェア実行環境に配置するために、従来は、アプリケーションソフトウェアを配置するソフトウェア実行環境が、事前に固定的に決められていた。
以下に説明するアプリケーションソフトウェア配置システム１は、複数のアプリケーションソフトウェアそれぞれが、複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報を動的に決定するように構成されている。
【０００７】
［アプリケーションソフトウェア配置システム１］
以下、本発明の実施形態として、アプリケーションソフトウェア配置システム１を説明する。
図１（Ａ）は、本発明に係る分散処理プログラムが実行されるアプリケーションソフトウェア配置システム１の構成を例示する図である。
図１（Ａ）に示すように、アプリケーションソフトウェア配置システム１は、ソフトウェア無線装置２－１～２－ｎおよび制御装置３が、ＬＡＮなどのネットワーク１００を介して相互に接続されている。
ただし、ｎは１以上の整数であって、ｎが常に同じ数を示すとは限らない。
以下、「ソフトウェア無線装置２－１～２－ｎ」など、複数存在しうる構成部分のいずれかが、特定されずに示されるときには、単に「ソフトウェア無線装置２」などと略記されることがある。
また、以下の各図において、実質的に同じ構成部分には、同じ符号が付される。
【０００８】
［ソフトウェア無線装置２］
図１（Ｂ）は、図１（Ａ）に示したソフトウェア無線装置２のハードウェア構成を例示する図である。
図１（Ｂ）に示すように、ソフトウェア無線装置２は、ＲＦ部２０２と、ＲＦ部２０２に接続されたアンテナ２００と、ＣｏｍｐａｃｔＰＣＩ(Peripheral Component Interconnect)などのバス２０４を介して接続されたモデム部２２、コーデック部２４および制御部２６から構成される。
また、モデム部２２、コーデック部２４および制御部２６は、それぞれ、メモリ２０６およびＣＰＵ２０８などを含み、アプリケーションソフトウェアがメモリ２０６にロードされ、モデム部２２、コーデック部２４および制御部２６のＯＳ（図示せず）上で、モデム部２２、コーデック部２４および制御部２６のハードウェア資源を具体的に利用して実行される。
なお、これらの構成部分においては、ＣＰＵ２０８とともに、あるいは、ＣＰＵの代わりに、ＤＳＰ(Digital Signal Processor)が用いられてもよい。
【０００９】
ＲＦ部２０２は、アンテナ２００を介して所定の無線方式の信号を受信し、モデム部２２に対して出力する。
また、ＲＦ部２０２は、モデム部２２から入力される信号を、アンテナ２００を介して

所定の無線方式で送信する。
モデム部２２は、ＲＦ部２０２から入力された信号を復調し、コーデック部２４に対して出力する。
また、モデム部２２は、コーデック部２４から入力された信号を変調し、ＲＦ部２０２に対して出力する。
【００１０】
コーデック部２４は、モデム部２２から入力される信号をデコードし、制御部２６に対して出力する。
また、コーデック部２４は、制御部２６から入力される信号をエンコードし、モデム部２２に対して出力する。
また、コーデック部２４は、ＬＡＮインタフェース（ＩＦ）２１０を含み、ネットワーク１００に接続されている。
制御部２６は、モデム部２２およびコーデック部２４などのソフトウェア無線装置２の各部の制御を行う。
また、制御部２６は、ＬＡＮインタフェース２１０を含み、ネットワーク１００を介して制御装置３に制御される。
【００１１】
［制御装置３］
図１（Ｃ）は、図１（Ａ）に示した制御装置３のハードウェア構成を例示する図である。
図１（Ｃ）に示すように、制御装置３は、メモリ２０６およびＣＰＵ２０８などを含む情報処理装置３００、キーボードおよび表示装置などを含む外部入出力装置３０２、データ通信を行うための通信装置３０４、および、ハードディスクなどの記録媒体３０８に対してデータ記録を行う記録装置３０６などから構成される。
つまり、制御装置３は、外部からの操作に応じて、ソフトウェア無線装置２を制御することができるコンピュータとしての構成部分を有している。
たとえば、制御装置３は、外部からの操作に応じて、ソフトウェア無線装置２に対して、実行対象となるアプリケーションソフトウェアの配置、アプリケーションソフトウェア間のコネクションの確立およびアプリケーションソフトウェアのコンフィギュレーションといったアプリケーションソフトウェアの組み立て実行を指示する。
【００１２】
［分散処理プログラム２０］
以下、アプリケーションソフトウェア配置システム１のソフトウェア無線装置２において動作するソフトウェアを説明する。
図２は、図１（Ｂ）のモデム部２２、コーデック部２４および制御部２６において実行される分散処理プログラム２０の構成を示す図である。
分散処理プログラム２０は、モデム部のメモリ２２０、コーデック部のメモリ２４０および制御部のメモリ２６０に記憶されるソフトウェアがバス２０４内のソフトウェアバスであるＯＲＢ(Object Request Broker)を介して接続されることにより、ＣＯＲＢＡ(Common Object Request Broker Architecture)を構成する。
ＣＯＲＢＡを構成するＣＯＲＢＡアプリケーションは、オブジェクトを含むソフトウェアコンポーネントをインスタンス化した実行形式のプログラムを意味し、スケルトンおよびスタブを介してＯＲＢと通信することができる。
なお、以下、説明の具体化、明確化のために、分散処理プログラム２０によって配置処理される「アプリケーションソフトウェア」が、「ＣＯＲＢＡアプリケーション」である場合を具体例とする。
【００１３】
モデム部メモリ２２０は、モデム部ＣＯＲＢＡアプリケーション２２２およびＧＰＰＤｅｖｉｃｅ２２４を記憶する。
モデム部ＣＯＲＢＡアプリケーション２２２は、モデム部２２が信号の変復調を行う。

ＧＰＰＤｅｖｉｃｅ２２４は、ＣＯＲＢＡアプリケーションであって、たとえば、ＤＳＰ、シリアルデバイス、オーディオデバイスなどのＣＯＲＢＡ非対応デバイスにアクセス可能とする論理デバイスである。

また、ＧＰＰＤｅｖｉｃｅ２２４は、後述するアプリケーションファクトリ２６４から、ＣＯＲＢＡアプリケーションのロードおよび実行指示を受けると、ＧＰＰＤｅｖｉｃｅ２２４が記憶されるメモリ上に、ＣＯＲＢＡアプリケーションをロードし、実行する。

コーデック部メモリ２４０は、コーデック部２４が、信号をエンコードおよびデコードするためのコーデック部ＣＯＲＢＡアプリケーション２４２を記憶する。

【００１４】

制御部メモリ２６０は、ＨＣＩ(Host Controller Interface)２６２、アプリケーションファクトリ２６４、ドメインマネージャ２６６およびドメインプロファイル２６８を記憶する。

ＨＣＩ２６２は、ＣＯＲＢＡアプリケーションであって、ＬＡＮインタフェース２１０を介して、図１（Ａ）の制御装置３からの組み立て実行指示を受け取り、受け取った組み立て実行指示をアプリケーションファクトリ２６４に出力する。

アプリケーションファクトリ２６４は、ＣＯＲＢＡアプリケーションであって、ＨＣＩ２６２から組み立て実行指示を受け取ると、ドメインマネージャ２６６またはドメインプロファイル２６８から読み出した情報に基づいて、実行対象ＣＯＲＢＡアプリケーションを、実行可能なソフトウェア無線装置２に配置し、実行させる。

【００１５】

ドメインマネージャ２６６は、ＣＯＲＢＡアプリケーションであって、ソフトウェア無線装置２に実行対象ＣＯＲＢＡアプリケーションを配置するための情報であるオブジェクトリファレンスを管理する。

ドメインプロファイル２６８には、ＣＯＲＢＡアプリケーションの組み立て情報が記述されたＳＡＤ(Software Assembly Descriptor)、ＣＯＲＢＡアプリケーションのプロパティ情報が記述されたＳＰＤ(Software Property Descriptor)およびＣＯＲＢＡアプリケーションのプロパティ情報に関する詳細な情報が記述されたＰＲＦ(Property Descriptor)などが含まれる。

なお、ドメインプロファイル２６８は、ＸＭＬ(eXtensible Markup Language)形式で記述される。

【００１６】

［アプリケーションファクトリ４０の処理］

図３（Ａ）は、図２に示したアプリケーションファクトリ４０の処理を示すフローチャートである。

以下、図３（Ａ）を参照して、アプリケーションファクトリ４０の処理をさらに説明する。

図１（Ａ）の制御装置３から、アプリケーションソフトウェアの組み立て実行指示を受け取ることによって、アプリケーションファクトリ４０の処理が開始する。

処理が開始すると、図３（Ａ）に示すように、ステップ４００（Ｓ４００）において、アプリケーションファクトリ４０は、組み立て実行指示とともに、配置情報の一覧が通知されているか否かを判断する。

配置情報は、実行対象ＣＯＲＢＡアプリケーションが、いずれのソフトウェア無線装置２に配置され得るかを示す情報である。

【００１７】

図３（Ｂ）は、配置情報の一覧を例示する図である。

図３（Ｂ）に示すように、配置情報の一覧は、たとえば、実行対象ＣＯＲＢＡアプリケーションのＩＤと配置可能なソフトウェア無線装置２のモデム部メモリ２２０に記憶されるＧＰＰＤｅｖｉｃｅ２２４のＩＤの組み合わせの一覧で示される。

なお、以下、「『配置可能なソフトウェア無線装置２のモデム部メモリ２２０に記憶される』ＧＰＰＤｅｖｉｃｅ２２４」を、「『配置可能なソフトウェア無線装置２に常駐す

る』ＧＰＰＤｅｖｉｃｅ２２４」と記述する。

【００１８】

ステップ４００（Ｓ４００；図３（Ａ））において、アプリケーションファクトリ４０は、組み立て実行指示とともに配置情報の一覧が通知されていると判断するときは、アプリケーションファクトリ４０は、Ｓ４０２の処理に進み（図４を参照して後述）、それ以外のときは、Ｓ４０４の処理に進む（図６を参照して後述）。

ステップ４０６（Ｓ４０６）において、アプリケーションファクトリ４０は、ステップ４０２（Ｓ４０２）またはステップ４０４（Ｓ４０４）の処理により決定した配置情報に基づいて、実行対象ＣＯＲＢＡアプリケーションの配置可能なソフトウェア無線装置２へのロード指示を行う。

ステップ４０８（Ｓ４０８）において、配置可能なソフトウェア無線装置２へロードされた実行対象ＣＯＲＢＡアプリケーションの実行指示を行う。

【００１９】

［配置情報の正当性検査処理］

図４は、図３（Ａ）に示した配置情報の正当性検査処理（Ｓ４０２）を示すフローチャートである。

配置情報の正当性検査処理では、図３（Ｂ）で示した実行対象ＣＯＲＢＡアプリケーションのＩＤと配置可能なソフトウェア無線装置２に常駐するＧＰＰＤｅｖｉｃｅ２２４のＩＤの組み合わせの一覧が、それぞれ正しい組み合わせであるか否かを検査する。

以下、図４を参照して、配置情報の正当性検査処理（Ｓ４０２）をさらに説明する。

【００２０】

図４に示すように、ステップ４１０（Ｓ４１０）において、図２に示したアプリケーションファクトリ４０は、図２に示したドメインマネージャ２６６から、図１（Ａ）に示したアプリケーションソフトウェア配置システム１に存在するすべてのソフトウェア無線装置２に常駐するＧＰＰＤｅｖｉｃｅ２２４（図２）のオブジェクトリファレンスを取得する。

ステップ４１２（Ｓ４１２）において、アプリケーションファクトリ４０は、取得したＧＰＰＤｅｖｉｃｅ２２４のオブジェクトリファレンスを利用して、各ＧＰＰＤｅｖｉｃｅ２２４のＩＤの一覧を取得する。

たとえば、アプリケーションファクトリ４０は、ＧＰＰＤｅｖｉｃｅ２２４の場所情報をオブジェクトリファレンスから取得し、取得した場所情報に基づいて、ＧＰＰＤｅｖｉｃｅ２２４にアクセスし、ＧＰＰＤｅｖｉｃｅ２２４が実装するＩＤ取得用のＣＯＲＢＡインタフェースをコールすることにより、ＧＰＰＤｅｖｉｃｅ２２４のＩＤの一覧を取得する。

【００２１】

ステップ４１４（Ｓ４１４）において、アプリケーションファクトリ４０は、組み立て実行指示とともに通知された配置情報の一覧（図３（Ｂ））に含まれるＧＰＰＤｅｖｉｃｅ２２４のＩＤそれぞれが、ステップ４１０において取得したＧＰＰＤｅｖｉｃｅ２２４のＩＤの一覧のいずれかの中に含まれているか否かを判断する。

組み立て実行指示とともに通知された配置情報の一覧に含まれるＧＰＰＤｅｖｉｃｅ２２４のＩＤが、ステップ４１０において取得したＧＰＰＤｅｖｉｃｅ２２４のＩＤの一覧のいずれかの中に含まれていると判断するときは、Ｓ４１６の処理に進み、それ以外のときは、Ｓ４２０の処理に進む。

【００２２】

ステップ４１６（Ｓ４１６）において、アプリケーションファクトリ４０は、図２で示したドメインプロファイル２６８の１つであるＳＡＤから、実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧を取得する。

たとえば、アプリケーションファクトリ４０は、ＸＭＬパーサ（図示せず）を用いて、ＳＡＤに含まれるタグを解析して、実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧を取得する。

【００２３】
ステップ４１８（Ｓ４１８）において、アプリケーションファクトリ４０は、組み立て実行指示とともに通知された配置情報の一覧（図３（Ｂ））に含まれる実行対象ＣＯＲＢＡアプリケーションのＩＤそれぞれが、ステップ４１６において取得した実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧のいずれかの中に含まれているか否かを判断する。
組み立て実行指示とともに通知された配置情報の一覧に含まれる実行対象ＣＯＲＢＡアプリケーションのＩＤが、ステップ４１６において取得した実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧のいずれかの中に含まれていると判断するときは、処理を終了し、それ以外のときは、Ｓ４２０の処理に進む。
ステップ４２０（Ｓ４２０）において、アプリケーションファクトリ４０は、組み立て実行指示とともに通知された配置情報の一覧に含まれる少なくとも１つ以上のＧＰＰＤｅｖｉｃｅ２２４または実行対象ＣＯＲＢＡアプリケーションが、図１（Ａ）に示したアプリケーションソフトウェア配置システム１内に存在しないとみなして、エラー出力などを行う。
【００２４】
図５は、図４に示した配置情報の正当性検査処理のうち、ステップ４１０（Ｓ４１０）からステップ４１８（Ｓ４１８）における処理を例示する図である。
図５には、組み立て指示とともに通知された配置情報の一覧６００、ステップ４１２（Ｓ４１２）において取得した配置場所に常駐するＧＰＰＤｅｖｉｃｅＩＤの一覧６０２およびステップ４１６（Ｓ４１６）において取得した実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧６０４が含まれる。
たとえば、ステップ４１４（Ｓ４１４）において、図５に示した配置情報の一覧６００のうち、配置場所に常駐するＧＰＰＤｅｖｉｃｅのＩＤの一覧（列（Ｂ））と、ステップ４１２（Ｓ４１２）において取得した配置場所に常駐するＧＰＰＤｅｖｉｃｅのＩＤの一覧６０２（列（Ｂ'））を比較する。
その結果、列（Ｂ'）のＩＤ一覧には、列（Ｂ）の一覧に存在する「ＩＤ＝８８８８」が存在しないため、ＧＰＰＤｅｖｉｃｅの「ＩＤ＝８８８８」に対応する「ＩＤ＝ＨＨＨＨ」の実行対象ＣＯＲＢＡアプリケーションは、アプリケーションソフトウェア配置システム１に存在するいずれのソフトウェア無線装置２にも配置できないと判断される。
また、たとえば、ステップ４１８（Ｓ４１８）において、配置情報の一覧６００のうち、実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧（列（Ａ））と、ステップ４１６（Ｓ４１６）において取得した実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧６０４（列（Ａ'）を比較する。
その結果、列（Ａ'）のＩＤ一覧には、列（Ａ）の一覧に存在する「ＩＤ＝ＨＨＨＨ」が存在しないため、「ＩＤ＝ＨＨＨＨ」のＣＯＲＢＡアプリケーションは、
実行対象ではないと判断される。
【００２５】
［配置場所の検査と決定］
図６は、図３（Ａ）に示した配置場所の検査と決定処理（Ｓ４０４）を示すフローチャートである。
以下、図６を参照して、配置場所の検査と決定処理（Ｓ４０４）をさらに説明する。
図６に示すように、ステップ４２２（Ｓ４２２）において、アプリケーションファクトリ４０は、図２で示したドメインプロファイル２６８の１つであるＳＰＤから、実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧と、それぞれのプロパティ情報を取得する。
図７は、ＳＰＤを例示する図である。たとえば、アプリケーションファクトリ４０は、ＸＭＬパーサ（図示せず）を用いて、ＳＰＤに含まれるタグを解析して、実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧と、それぞれのプロパティ情報を取得する。
ＳＰＤから取得するＣＯＲＢＡアプリケーションのプロパティ情報とは、たとえば、図７で示された「<os>」タグの属性「name」の値に対応するＣＯＲＢＡアプリケーションが実行可能なＯＳの名称や、「<processor>」タグの属性「name」の値に対応する実行可能

なプロセッサの名称の情報などである。
図７に示されるタグ、属性または子要素、およびその意味は、以下の表のように対応する。
【００２６】
【表１】

タグ名称	属性または子要素	意味
os	name	このＣＯＲＢＡアプリケーションが実行可能なＯＳの名称についての情報。
	version	このＣＯＲＢＡアプリケーションが実行可能なバージョンについての情報。
processor	name	このＣＯＲＢＡアプリケーションが実行可能なプロセッサの名称の情報。
dependency	type	このＣＯＲＢＡアプリケーションが実行上、依存する種別の名称。プロパティ情報としては特に使用しない。
propertyref	refid	「dependency」におけるプロパティのＩＤ情報。
	value	「dependency」におけるプロパティの値情報。
softpkgref	localfile	他のＳＰＤファイルの名称。他のＳＰＤファイルに記述されるプロパティ情報も使用する場合に指定する。
	implref	他のＳＰＤのimplementationタグのＩＤ。他のＳＰＤファイルに記述されるプロパティ情報も使用する場合に指定する。
usesdevice	type	このＣＯＲＢＡアプリケーションが実行上、使用するデバイス種別のタイプおよびＩＤ情報。 プロパティ情報としては特に使用しない。
	id	
propertyref	refid	「usesdevice」におけるプロパティＩＤ情報。
	value	「usesdevice」におけるプロパティの値情報。

【００２７】
アプリケーションファクトリ４０は、ステップ４２２において取得したプロパティ情報のそれぞれを、情報の種類を一意に定めるＩＤとその値を組にして保持する。
ステップ４２４（Ｓ４２４；図６）において、アプリケーションファクトリ４０は、図２で示したドメインプロファイル２６８の１つであるＰＲＦから、ステップ４２２において取得した実行対象ＣＯＲＢＡアプリケーションのプロパティ情報に関する詳細な情報を取得する。
プロパティ情報に関する詳細な情報とは、たとえば、ＳＰＤから取得した各プロパティの値のデータ型の情報などである。
データ型の値には、「boolean」、「char」、「double」、「float」、「short」、「long」、「object」、「octet」、「string」、「ulong」、「ushort」などを使用する。

図８は、ＰＲＦを例示する図である。たとえば、アプリケーションファクトリ４０は、ＸＭＬパーサ（図示せず）を用いて、ＰＲＦに含まれるタグを解析して、ステップ４２２において取得した実行対象となるＣＯＲＢＡアプリケーションのプロパティの値のデータ型の情報などを取得する。

具体的には、まず、図８で示された「<simple>」タグの属性「id」の値と、ＳＰＤから取得した、情報の種類を一意に定めるＩＤの値とが同一である「<simple>」タグを検索する。

次に、その「<simple>」タグの子要素である「<kind>」タグの属性「kindtype」の値を取得する。

「kindtype」の値が、"allocation"であるときは、同じ「<simple>」タグの属性「type」の値からデータ型を取得する。「kindtype」の値が、"allocation"以外のときは、データ型のデフォルト値として"ulong"を取得する。

【００２８】

ステップ４２６（Ｓ４２６；図６）において、アプリケーションファクトリ４０は、ステップ４２２（Ｓ４２２）およびステップ４２４（Ｓ４２４）で取得したプロパティ情報に基づいて、図１（Ａ）に示したアプリケーションソフトウェア配置システム１に存在するすべてのソフトウェア無線装置２に常駐するＧＰＰＤｅｖｉｃｅ２２４に対して、実行対象ＣＯＲＢＡアプリケーションが配置可能であるか否かを問い合わせる。

たとえば、アプリケーションファクトリ４０は、ステップ４２２（Ｓ４２０）において保持したＩＤとその値の組（プロパティ情報）を引数として、ＧＰＰＤｅｖｉｃｅ２２４が実装する「allocateCapacity」というＣＯＲＢＡインタフェースをコールする。

なお、「allocateCapacity」ＣＯＲＢＡインタフェースをコールされたときのＧＰＰＤｅｖｉｃｅ２２４での処理については、図９を参照して後述する。

【００２９】

ステップ４２８（Ｓ４２６）において、アプリケーションファクトリ４０は、ステップ４２６（Ｓ４２４）においてコールした「allocateCapacity」ＣＯＲＢＡインタフェースの戻り値を受け取り、戻り値がTRUEであるか否かを判断する。

戻り値がTRUEであると判断するときは、Ｓ４３０の処理に進み、それ以外のときは、Ｓ４３２の処理に進む。

【００３０】

ステップ４３０（Ｓ４３０）において、アプリケーションファクトリ４０は、ステップ４２６（Ｓ４２６）において問い合わせたプロパティ情報を持つＣＯＲＢＡアプリケーションの配置場所を決定して、処理を終了する。

配置場所は、ステップ４２６（Ｓ４２６）においてコールした「allocateCapacity」ＣＯＲＢＡインタフェースを実装するＧＰＰＤｅｖｉｃｅを含むソフトウェア無線装置２である。

アプリケーションファクトリ４０は、ステップ４２６（Ｓ４２６）において問い合わせたプロパティ情報を持つＣＯＲＢＡアプリケーションのＩＤと、ステップ４２６においてコールした「allocateCapacity」を実装するＧＰＰＤｅｖｉｃｅのＩＤの組み合わせを配置情報として保持する。

【００３１】

ステップ４３２（Ｓ４３２）において、アプリケーションファクトリ４０は、ステップ４２６（Ｓ４２６）においてすでに問い合わせたＧＰＰＤｅｖｉｃｅ以外のＧＰＰＤｅｖｉｃｅが存在するか否かを判断する。

存在すると判断するときは、Ｓ４２６の処理に進み、それ以外のときは、Ｓ４３４の処理に進む。

ステップ４３４（Ｓ４３４）において、アプリケーションファクトリ４０は、ステップ４２６（Ｓ４２６）において問い合わせたプロパティ情報を持つＣＯＲＢＡアプリケーションは、アプリケーションソフトウェア配置システム１に存在するいずれのソフトウェア無線装置２にも配置できないとみなして、エラー出力などを行う。

【００３２】
［ＧＰＰＤｅｖｉｃｅ５０］
以下、実行対象ＣＯＲＢＡアプリケーションのソフトウェア無線装置２への配置可否検査処理、および、配置場所が決定したＣＯＲＢＡアプリケーションのロード、実行処理を行うＧＰＰＤｅｖｉｃｅ５０の動作について説明する。
ＧＰＰＤｅｖｉｃｅ５０の配置可否検査処理は、図６で示したステップ４２６（Ｓ４２６）において、アプリケーションファクトリ４０によって、「allocateCapacity」ＣＯＲＢＡインタフェースがコールされたときに行われる処理である。
図９（Ａ）は、ＧＰＰＤｅｖｉｃｅ５０の配置可否検査処理を示すフローチャートである。
図９（Ｂ）は、ステップ４２６（Ｓ４２６）と、図９（Ａ）で示したＧＰＰＤｅｖｉｃｅ５０の配置可否検査処理との関係を例示する図である。
以下、図９（Ａ）、（Ｂ）を参照して、ＧＰＰＤｅｖｉｃｅ５０の配置可否検査処理をさらに説明する。
【００３３】
図９（Ａ）に示すように、ステップ５０２（Ｓ５０２）において、ＧＰＰＤｅｖｉｃｅ５０は、各検査項目に基づいて、ＧＰＰＤｅｖｉｃｅ５０が常駐するソフトウェア無線装置２の実行環境の情報と、「allocateCapacity」ＣＯＲＢＡインタフェースを介してアプリケーションファクトリ４０から受け取ったプロパティ情報を比較して、一致するか否かを検査する。
各検査項目とは、たとえば、図９（Ｂ）に示すように、ＯＳ名称やＰｒｏｃｅｓｓｏｒ名称などの情報である。
実行対象ＣＯＲＢＡアプリケーションを実行するために必要な各検査項目の情報と、検査対象となるソフトウェア無線装置２の各検査項目の情報とを比較することで、実行対象ＣＯＲＢＡアプリケーションを実行するために必要な実行環境が、検査対象となるソフトウェア無線装置２の実行環境と合致するか否かを判断する。
【００３４】
ステップ５０４（Ｓ５０４）において、ＧＰＰＤｅｖｉｃｅ５０は、ステップ５０２（Ｓ５０２）において検査したすべての項目について、ＧＰＰＤｅｖｉｃｅ５０が常駐するソフトウェア無線装置２の実行環境の情報と、アプリケーションファクトリ４０から受け取ったプロパティ情報が一致しているか否かを判断する。
すべてが一致していると判断するときは、Ｓ５０６の処理に進み、それ以外のときは、Ｓ５０８の処理に進む。
ステップ５０６（Ｓ５０６）において、ＧＰＰＤｅｖｉｃｅ５０は、「allocateCapacity」ＣＯＲＢＡインタフェースの返り値として、"TRUE"を返して、処理を終了する。
ステップ５０８（Ｓ５０８）において、ＧＰＰＤｅｖｉｃｅ５０は、「allocateCapacity」ＣＯＲＢＡインタフェースの返り値として、"FALSE" を返して、処理を終了する。
【００３５】
図１０（Ａ）は、図３（Ａ）に示したステップ４０６（Ｓ４０６）における、実行対象ＣＯＲＢＡアプリケーションのソフトウェア無線装置２へのロード指示と、ＧＰＰＤｅｖｉｃｅ５０のロード処理との関係を例示する図である。
図１０（Ａ）に示すように、たとえば、ＧＰＰＤｅｖｉｃｅ５０－１～５０－ｎは、それぞれ、ステップ４０６（Ｓ４０６）におけるアプリケーションファクトリ４０から、「load」ＣＯＲＢＡインタフェース２２６を介して、実行対象ＣＯＲＢＡアプリケーションのソフトウェア無線装置２－１～２－ｎへのロード指示を受ける。
ロード指示を受けたＧＰＰＤｅｖｉｃｅ５０－１～５０－ｎは、それぞれ、ソフトウェア無線装置２－１～２－ｎに、実行対象ＣＯＲＢＡアプリケーション２２２～２２２－ｎをロードする（Ｓ５１０）。
【００３６】
図１０（Ｂ）は、図３（Ａ）に示したステップ４０８（Ｓ４０８）における、ロードさ

れた実行対象ＣＯＲＢＡアプリケーションの実行指示と、ＧＰＰＤｅｖｉｃｅ５０の実行処理との関係を例示する図である。

図１０（Ｂ）に示すように、たとえば、ＧＰＰＤｅｖｉｃｅ５０－１～５０－ｎは、それぞれ、ステップ４０８（Ｓ４０８）におけるアプリケーションファクトリ４０から、「execute」ＣＯＲＢＡインタフェース２２８を介して、ロードされた実行対象ＣＯＲＢＡアプリケーション２２２－１～２２２－ｎの実行指示を受ける。

実行指示を受けたＧＰＰＤｅｖｉｃｅ５０－１～５０－ｎは、それぞれ、ステップ５１０（Ｓ５１０）においてロードされた実行対象ＣＯＲＢＡアプリケーション２２２－１～２２２－ｎを実行する（Ｓ５１２）。

【００３７】

［配置場所の検査と決定の動作例］

図１１（Ａ）、（Ｂ）は、配置場所の検査と決定の動作例を示す通信シーケンス図である。

図１１（Ａ）、（Ｂ）に示すように、たとえば、アプリケーションソフトウェア配置システム１内には、２つのソフトウェア無線装置２にそれぞれ常駐するＧＰＰＤｅｖｉｃｅ５０－１、５０－２が存在するとする。

なお、図１１（Ａ）、（Ｂ）において、ＧＰＰＤｅｖｉｃｅ５０－１、５０－２としたのは例示の１つであり、ＧＰＰＤｅｖｉｃｅ５０がいくつ存在しても、実質的に同様の動作となる。

【００３８】

図１１（Ａ）に示すように、ステップ４２６（Ｓ４２６）において、アプリケーションファクトリ４０は、ＧＰＰＤｅｖｉｃｅ５０－１が実装する「allocateCapacity」ＣＯＲＢＡインタフェースを介して、プロパティ情報をＧＰＰＤｅｖｉｃｅ５０－１に対して送信する。

ステップ５０４（Ｓ５０４）において、ＧＰＰＤｅｖｉｃｅ５０－１は、受信したプロパティ情報に基づいて、ＧＰＰＤｅｖｉｃｅ５０－１が常駐するソフトウェア無線装置２－１に配置可能か否かを判断した結果、配置可能であると判断する。

ステップ５０６（Ｓ５０６）において、ＧＰＰＤｅｖｉｃｅ５０－１は、戻り値"TRUE"を、アプリケーションファクトリ４０に対して送信する。

ステップ４３０（Ｓ４３０）において、アプリケーションファクトリ４０は、ＧＰＰＤｅｖｉｃｅ５０－１が常駐するソフトウェア無線装置２－１を、送信したプロパティ情報を保持するＣＯＲＢＡアプリケーションの配置場所として決定する。

【００３９】

図１１（Ｂ）に示すように、ステップ４２６（Ｓ４２６）において、アプリケーションファクトリ４０は、ＧＰＰＤｅｖｉｃｅ５０－１が実装する「allocateCapacity」ＣＯＲＢＡインタフェースを介して、プロパティ情報をＧＰＰＤｅｖｉｃｅ５０－１に対して送信する。

ステップ５０４（Ｓ５０４）において、ＧＰＰＤｅｖｉｃｅ５０－１は、受信したプロパティ情報に基づいて、ＧＰＰＤｅｖｉｃｅ５０－１が常駐するソフトウェア無線装置２－１に配置可能か否かを判断した結果、配置不可であると判断する。

ステップ５０８（Ｓ５０８）において、ＧＰＰＤｅｖｉｃｅ５０－１は、戻り値"FALSE"を、アプリケーションファクトリ４０に対して送信する。

【００４０】

ステップ４２６（Ｓ４２６）において、アプリケーションファクトリ４０は、ＧＰＰＤｅｖｉｃｅ５０－２が実装する「allocateCapacity」ＣＯＲＢＡインタフェースを介して、プロパティ情報をＧＰＰＤｅｖｉｃｅ５０－２に対して送信する。

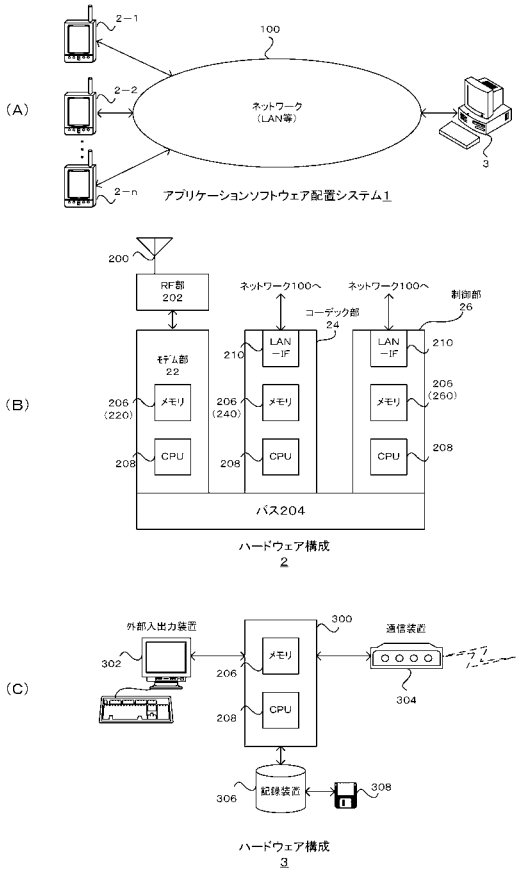
ステップ５０８（Ｓ５０８）において、ＧＰＰＤｅｖｉｃｅ５０－２は、受信したプロパティ情報に基づいて、ＧＰＰＤｅｖｉｃｅ５０－２が常駐するソフトウェア無線装置２－２に配置可能か否かを判断した結果、配置可能であると判断する。

ステップ５０６（Ｓ５０６）において、ＧＰＰＤｅｖｉｃｅ５０－２は、戻り値"TRUE"

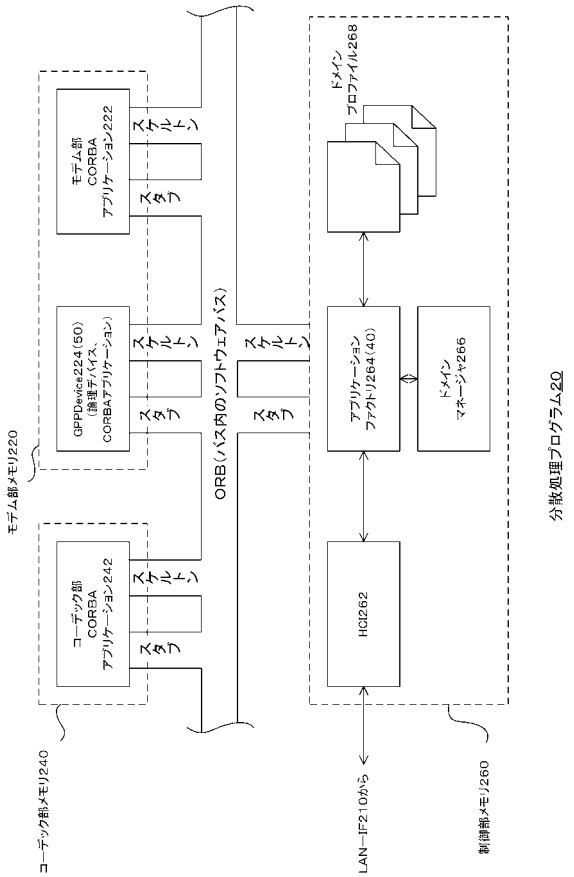
を、アプリケーションファクトリ４０に対して送信する。
ステップ４３０（Ｓ４３０）において、アプリケーションファクトリ４０は、ＧＰＰＤｅｖｉｃｅ５０－２が常駐するソフトウェア無線装置２－２を、送信したプロパティ情報を保持するＣＯＲＢＡアプリケーションの配置場所として決定する。
【００４１】
以上説明したように、本発明に係る分散処理プログラム２０においては、複数のアプリケーションソフトウェアそれぞれが、複数のソフトウェア実行環境のいずれにおいて配置され得るかを示す配置情報を決定することができる。
【図面の簡単な説明】
【００４２】
【図１】図１（Ａ）は、本発明に係る分散処理プログラムを実行するアプリケーションソフトウェア配置システムの構成を例示する図であり、図１（Ｂ）は、ソフトウェア無線装置のハードウェア構成を例示する図であり、図１（Ｃ）は、制御装置のハードウェア構成を例示する図である。
【図２】図２は、図１（Ｂ）のモデム部、コーデック部および制御部が実行する分散処理プログラムの構成を示す図である。
【図３】図３（Ａ）は、図２に示したアプリケーションファクトリの処理を示すフローチャートであり、図３（Ｂ）は、組み立て実行指示とともに通知された配置情報の一覧を例示する図である。
【図４】図４は、図３（Ａ）に示した配置情報の正当性検査処理（Ｓ４０２）を示すフローチャートである。
【図５】図５は、図４に示した配置情報の正当性検査処理のうち、ステップ４１０（Ｓ４１０）からステップ４１８（Ｓ４１８）の処理を例示する図である。
【図６】図６は、図３（Ａ）に示した配置場所の検査と決定処理（Ｓ４０４）を示すフローチャートである。
【図７】図７は、ステップ４２２（Ｓ４２２、図６）において、実行対象ＣＯＲＢＡアプリケーションのプロパティ情報を取得するために用いられた、ドメインプロファイルの１つであるＳＰＤを例示する図である。
【図８】図８は、ステップ４２４（Ｓ４２４、図６）において、実行対象ＣＯＲＢＡアプリケーションのプロパティ情報に関する詳細情報を取得するために用いられた、ドメインプロファイルの１つであるＰＲＦを例示する図である。
【図９】図９（Ａ）は、ＧＰＰＤｅｖｉｃｅの配置可否検査処理を示すフローチャートであり、図９（Ｂ）は、ステップ４２６（Ｓ４２６、図６）と、図９（Ａ）で示したＧＰＰＤｅｖｉｃｅの処理との関係を例示する図である。
【図１０】図１０（Ａ）は、図３（Ａ）に示したステップ４０６（Ｓ４０６）における、実行対象ＣＯＲＢＡアプリケーションのソフトウェア無線装置へのロード指示と、ＧＰＰＤｅｖｉｃｅのロード処理との関係を例示する図であり、図１０（Ｂ）は、図３（Ａ）に示したステップ４０８（Ｓ４０８）における、ロードされた実行対象ＣＯＲＢＡアプリケーションの実行指示と、ＧＰＰＤｅｖｉｃｅの実行処理との関係を例示する図である。
【図１１】図１１（Ａ）および図１１（Ｂ）は、配置場所の検査と決定の動作例を示す通信シーケンス図である。
【符号の説明】
【００４３】
１・・・アプリケーションソフトウェア配置システム
１００・・・ネットワーク
２・・・ソフトウェア無線装置
２２・・・モデム部
２４・・・コーデック部
２６・・・制御部
２００・・・アンテナ

２０２・・・ＲＦ部
２０４・・・バス
２０６・・・メモリ
２０８・・・ＣＰＵ
２１０・・・ＬＡＮ－ＩＦ
３・・・制御装置
３００・・・情報処理装置
３０２・・・外部入出力装置
３０４・・・通信装置
３０６・・・記録装置
３０８・・・記録媒体
２０・・・分散処理プログラム
２２０・・・モデム部メモリ
２２２・・・モデム部ＣＯＲＢＡアプリケーション
２２４・・・ＧＰＰＤｅｖｉｃｅ
２２６・・・load　ＣＯＲＢＡインタフェース
２２８・・・execute　ＣＯＲＢＡインタフェース
２４０・・・コーデック部メモリ
２４２・・・コーデック部ＣＯＲＢＡアプリケーション
２６０・・・制御部メモリ
２６２・・・ＨＣＩ
２６４・・・アプリケーションファクトリ
２６６・・・ドメインマネージャ
２６８・・・ドメインプロファイル
６００・・・組み立て情報とともに通知された配置情報の一覧
６０２・・・配置場所に常駐するＧＰＰＤｅｖｉｃｅのＩＤの一覧
６０４・・・実行対象ＣＯＲＢＡアプリケーションのＩＤの一覧

【図１】

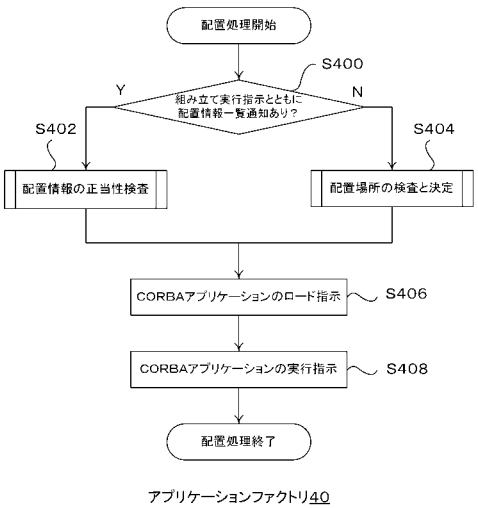


【図２】



【図３】

(A)

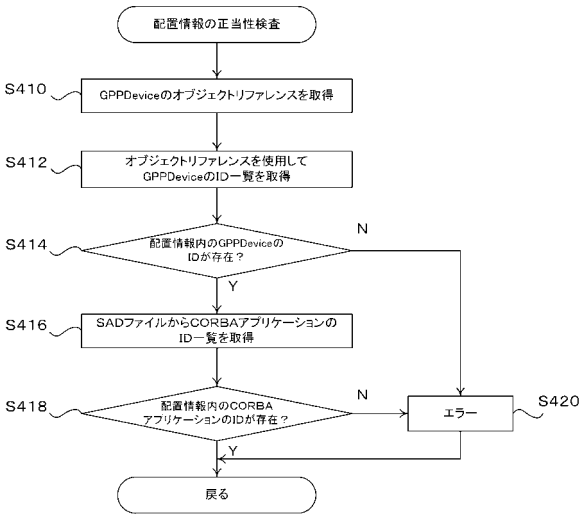


アプリケーションファクトリ40

(B)

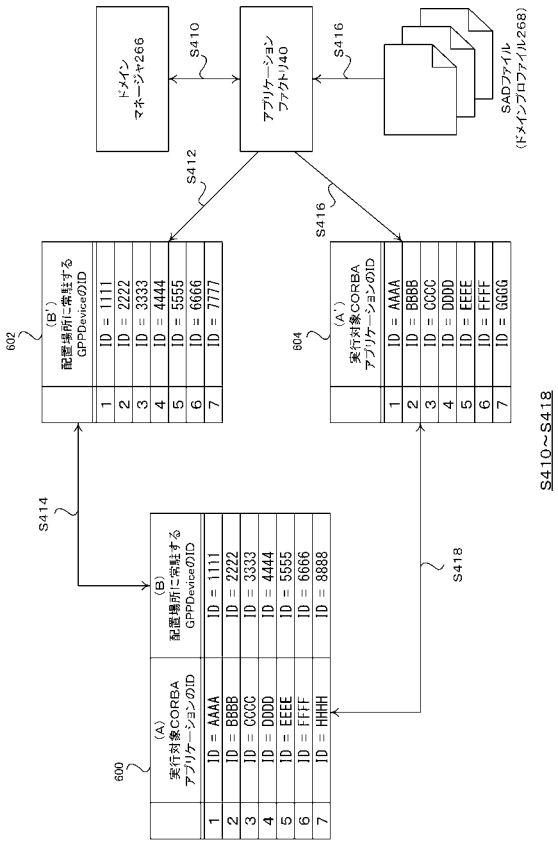
	実行対象CORBA アプリケーションのID	配置場所に常駐する GPPDeviceのID
1	ID = AAAA	ID = 1111
2	ID = BBBB	ID = 2222
3	ID = CCCC	ID = 3333
4	ID = DDDD	ID = 4444
5	ID = EEEE	ID = 5555
6	ID = FFFF	ID = 6666
7	ID = GGGG	ID = 8888

【図４】

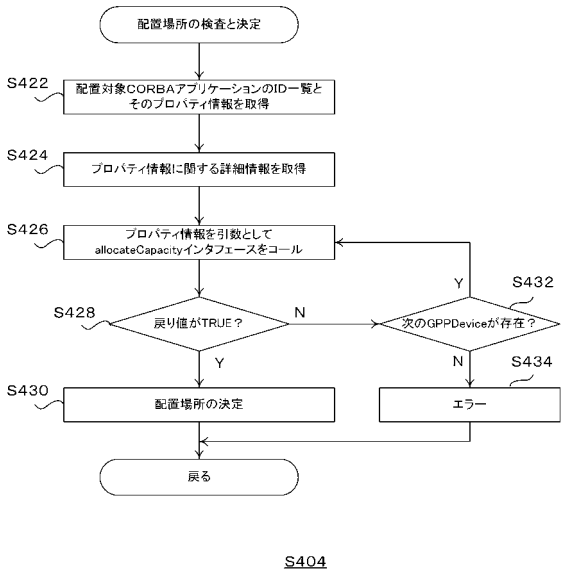


S402

【図５】



【図６】



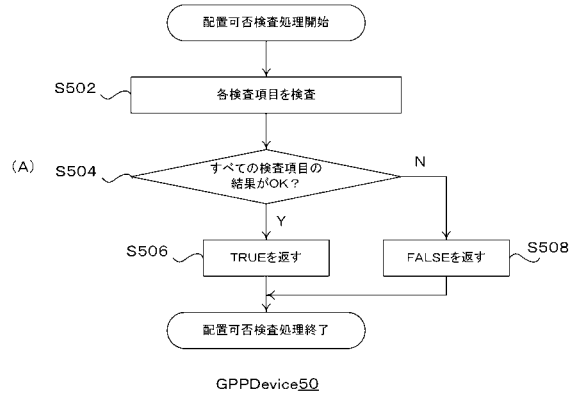
【図７】

```
<softpkg id="sp3" name="IODevice" type="sca_compliant" version="1.0.0">
    <title/>
    <author>
        <name>Hitachi Kokusai</name>
        <company>Hitachi Kokusai</company>
        <webpage>http://www.hitachikokuaielectric.com</webpage>
    </author>
    <description>This is IODevice SPD file</description>
    <descriptor name="">
        <localfile name="/DomainProfile/SCD/io.scd.xml"/>
    </descriptor>
    <implementation id="implementationID_1" aepcompliance="aep_compliant">
        <description>This is IODevice implementation</description>
        <propertyfile type="">
            <localfile name="DomainProfile/PropertyDescriptor/io.prf.xml"/>
        </propertyfile>
        <code type="Executable">
            <localfile name="/hitachikokusaielectric/IODevice"/>
            <entrypoint>XXX</entrypoint>
            <stacksize>0</stacksize>
            <priority>0</priority>
        </code>
        <compiler name="gcc" version="3.2"/>
        <programminglanguage name="c++"/>
        <humanlanguage name="English"/>
        <runtime name="" version=""/>
        <os name="Linux" version="Linux-version-2.4.18-timesys-4.0.346"/>
        <processor name="ppc"/>
        <dependency type="Property Reference Dependency">
            <propertyref refid="DCEce0d02eb-47f6-421e-87a0-30b353cb7c33" value="10" />
        </dependency>
        <dependency type="Property Reference Dependency">
            <softpkgref>
                <localfile name="SPD2.xml"/>
                <implref refid="implementationID_2"/>
            </softpkgref>
        </dependency>
    </implementation>
    <usesdevice type="UsesDevice" id="usesSerialService">
        <propertyref refid="DCE:DCEc8ege285-Z6d0-4687-8936-d0530d6098f0" value="2" />
    </usesdevice>
</softpkg>
```

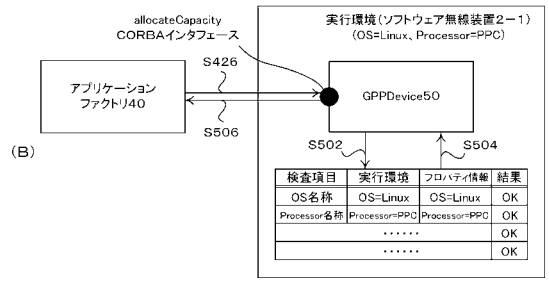
【図８】

```
<!DOCTYPE properties SYSTEM "dtd.2.2/properties.2.2.dtd">
<properties>
  <simple id="DCE80BF17F0-6C7F-11D4-A226-0050DA314CD6" type="string" name="os_name" mode="readonly">
    <description> This property identifies the os_name XML allocation property. </description>
    <!-- Valid values for the os_name element are: -->
    <!-- AIX, BSDi, VMS, DigitalUnix, DOS, HPBLS, HPUX, IRIX, -->
    <!-- Linux, LynxOS, MacOS, OS/2, AS/400, MVS, SCO CMW, -->
    <!-- SCO ODT, Solaris, SunOS, UnixWare, VxWorks, Win95, WinNT -->
    <!-- pSOS, RTXC -->
    <!-- The os_name values are case sensitive. -->
    <value>Linux</value>
    <kind kindtype="allocation"/>
    <action type="eq"/>
  </simple>

  <simple id="CE9B445600-6C7F-11D4-A226-0050DA314CD6" type="string" name="processor_name" mode="readonly">
    <description> This property identifies the processor_name XML allocation property. </description>
    <!-- Valid values for the processor_name element are: -->
    <!-- x86, mips, alpha, ppc, sparc, 680x0, vax, S/390, -->
    <!-- ppcG3, ppcG4, ppcG5, C5x, C6x, ADSP21xx -->
    <!-- The processor_name values are case sensitive. -->
    <value>ppc</value>
    <kind kindtype="allocation"/>
    <action type="eq"/>
  </simple>

  <simple id="XXXXXXXXX" name="" type="ulong" mode="readwrite">
    <value></value>
    <kind kindtype="allocation" /><!-- any, default "configure" -->
    <action type="eq" />
  </simple>
```

【図９】

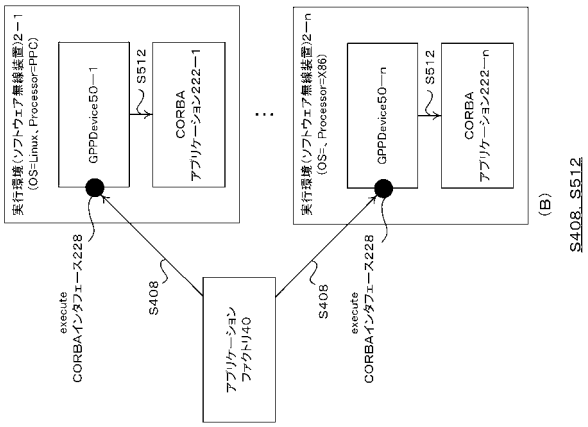


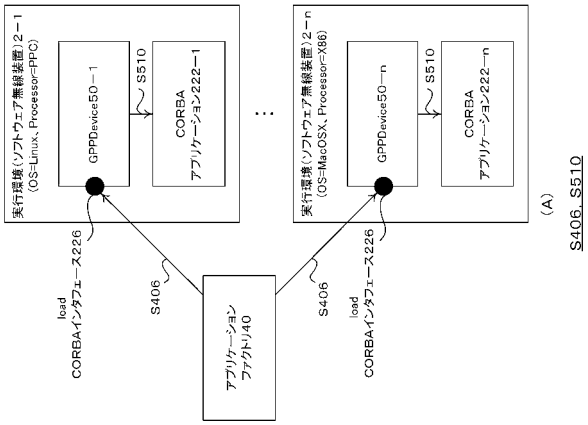
GPPDevice50



S426、S502～S506

【図１０】





【図１１】