世界知的所有権機関
国際事務局

PCT

特許協力条約に基づいて公開された国際出願

(51) 国際特許分類7 **G06F 9/44, B25J 9/00**	**A1**	(11) 国際公開番号	**WO00/42504**
		(43) 国際公開日	2000年7月20日(20.07.00)

(21) 国際出願番号 PCT/JP00/00123

(22) 国際出願日 2000年1月13日(13.01.00)

(30) 優先権データ

特願平11/6491	1999年1月13日(13.01.99)	JP
特願平11/6757	1999年1月13日(13.01.99)	JP
特願2000/6032	2000年1月11日(11.01.00)	JP

(71) 出願人（米国を除くすべての指定国について）
ソニー株式会社(SONY CORPORATION)[JP/JP]
〒141-0001 東京都品川区北品川6丁目7番35号 Tokyo, (JP)
(72) 発明者；および
(75) 発明者／出願人（米国についてのみ）
藤田雅博(FUJITA, Masahiro)[JP/JP]
坂本隆之(SAKAMOTO, Takayuki)[JP/JP]
佐部浩太郎(SABE, Kotaro)[JP/JP]
高木　剛(TAKAGI, Tsuyoshi)[JP/JP]
〒141-0001 東京都品川区北品川6丁目7番35号
ソニー株式会社内 Tokyo, (JP)

(74) 代理人
弁理士　田辺恵基(TANABE, Shigemoto)
〒150-0001 東京都渋谷区神宮前1丁目11番11-508号
グリーンフアンタジアビル5階 Tokyo, (JP)

(81) 指定国　CN, KR, US, 欧州特許 (DE, FR, GB)

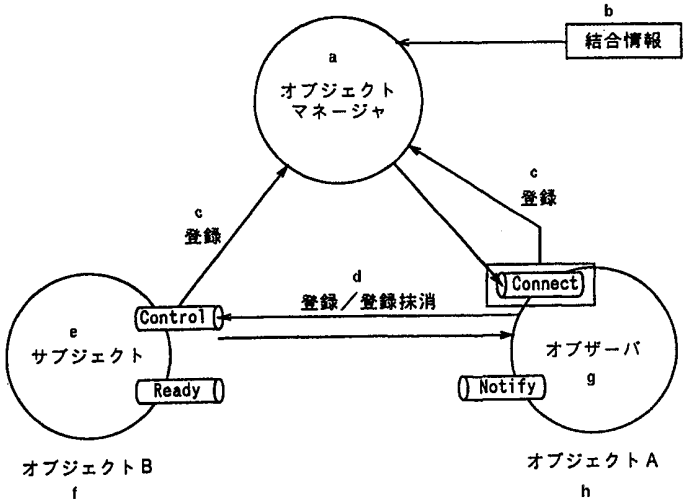
添付公開書類
国際調査報告書
請求の範囲の補正の期限前の公開；補正書受領の際には再公開される。

(54)Title: PROCESSOR, METHOD FOR COMMUNICATION BETWEEN OBJECTS, AND ROBOT

(54)発明の名称　演算処理装置、オブジェクト間通信方法及びロボット

(57) Abstract

A processor, method for communication between objects, and robot are disclosed and applied to a pet robot, enabling simple setting of adequate software adaptative to a current mode. The independence of an object is improved compared with the prior art. Based ont the results of comparison between information corresponding to the results of detection of a part held in each mode and the results of detection of a part concerning the current mode, the software is changed according to the mode. Further, based on coupling information, objects are so coupled that data is exchangeable.



a...OBJECT MANAGER
b...COUPLING INFORMATION
c...REGISTER
d...REGISTRATION/ELIMINATION FROM REGISTER
e...SUBJECT
f...OBJECT B
g...OBSERVER
h...OBJECT A

(57)要約

本発明は、演算処理装置、オブジェクト間通信方法及びロボットに関し、例えばペット型のロボットに適用して、現在の形態に対応する適切なソフトウエアを簡易に設定することができるようにする。また従来に比してオブジェクトの独立性を向上することができるようにする。本発明は、形態毎に保持された部品検出結果に対応する情報と、現在の形態に係る部品検出結果との比較結果に基づいて、形態に応じてソフトウエアを変更する。また本発明は、結合情報に基づいて、データ交換可能にオブジェクトを結合する。

PCTに基づいて公開される国際出願のパンフレット第一頁に掲載されたPCT加盟国を同定するために使用されるコード(参考情報)

AE アラブ首長国連邦
AG アンティグア・バーブーダ
AL アルバニア
AM アルメニア
AT オーストリア
AU オーストラリア
AZ アゼルバイジャン
BA ボズニア・ヘルツェゴビナ
BB バルバドス
BE ベルギー
BF ブルギナ・ファソ
BG ブルガリア
BJ ベナン
BR ブラジル
BY ベラルーシ
CA カナダ
CF 中央アフリカ
CG コンゴー
CH スイス
CI コートジボアール
CM カメルーン
CN 中国
CR コスタ・リカ
CU キューバ
CY キプロス
CZ チェッコ
DE ドイツ
DK デンマーク
DM ドミニカ
DZ アルジェリア
EE エストニア
ES スペイン
FI フィンランド
FR フランス
GA ガボン
GB 英国
GD グレナダ
GE グルジア
GH ガーナ
GM ガンビア
GN ギニア
GR ギリシャ
GW ギニア・ビサオ
HR クロアチア
HU ハンガリー
ID インドネシア
IE アイルランド
IL イスラエル
IN インド
IS アイスランド
IT イタリア
JP 日本
KE ケニア
KG キルギスタン
KP 北朝鮮
KR 韓国
KZ カザフスタン
LC セントルシア
LI リヒテンシュタイン
LK スリ・ランカ
LR リベリア
LS レソト
LT リトアニア
LU ルクセンブルグ
LV ラトヴィア
MA モロッコ
MC モナコ
MD モルドヴァ
MG マダガスカル
MK マケドニア旧ユーゴスラヴィア共和国
ML マリ
MN モンゴル
MR モーリタニア
MW マラウイ
MX メキシコ
MZ モザンビーク
NE ニジェール
NL オランダ
NO ノールウェー
NZ ニュー・ジーランド
PL ポーランド
PT ポルトガル
RO ルーマニア
RU ロシア
SD スーダン
SE スウェーデン
SG シンガポール
SI スロヴェニア
SK スロヴァキア
SL シエラ・レオネ
SN セネガル
SZ スワジランド
TD チャード
TG トーゴー
TJ タジキスタン
TM トルクメニスタン
TR トルコ
TT トリニダッド・トバゴ
TZ タンザニア
UA ウクライナ
UG ウガンダ
US 米国
UZ ウズベキスタン
VN ヴェトナム
YU ユーゴースラヴィア
ZA 南アフリカ共和国
ZW ジンバブエ

明　細　書

演算処理装置、オブジェクト間通信方法及びロボット

技術分野

本発明は、演算処理装置、オブジェクト間通信方法及びロボットに関し、例えばペット型のロボットに適用することができる。本発明は、形態毎に保持された部品検出結果に対応する情報と、現在の形態に係る部品検出結果との比較結果に基づいて、形態に応じてソフトウエアを変更することにより、現在の形態に対応する適切なソフトウエアを簡易に設定することができるようにする。また本発明は、結合情報に基づいて、データ交換可能にオブジェクトを結合することにより、従来に比してオブジェクトの独立性を向上することができるようにする。

背景技術

従来、例えばペット型のロボット等においては、オブジェクト指向によりソフトウエアが構成されるようになされ、オブジェクト間通信により種々のデータを処理して例えば動物に似た仕草等を実行できるようになされている。

ところでこの種のペット型のロボットにおいて、例えば脚等の構成部品を必要に応じて交換してユーザーの所望する形態に設定することができれば、この種のペット型のロボットをさらに一段と身近なものにできると考えられる。

この場合、形態に対応してソフトウエアを簡易に交換することが望まれる。

発明の開示

本発明は以上の点を考慮してなされたもので、第１に、現在の形態に対応する適切なソフトウエアを簡易に設定することができるロボットを提案しようとするものである。

またこのように脚等の構成部品を必要に応じて交換してユーザーの所望する形

態に設定する場合、全体の動作を制御するソフトウエアにおいては、オブジェクトを構築し直して、形態の変化に対応するようにソフトウエアを変更することが考えられる。

しかしながら、従来のオブジェクト指向のソフトウエアにおいては、オブジェクト間の明示的な関係を前提として設計されるようになされていることにより、オブジェクト間の依存性が高い。これによりこのように比較的自由な形態の変化に対応するようにソフトウェアを変更するようにしても、結局、各形態に対応した専用のソフトウエアの開発が必要となる。

このようなオブジェクト間の依存性を低減する１つの方法として、例えばデザインパターン（オブザーバーパターン）による方法によりソフトウエアを開発することが考えられる（Ｅｒｉｃｈ　Ｇａｍｍａ他著，オブジェクト指向における再利用のためのデザインパターン，ｐｐ３１３－３２４）。すなわちこのデザインパターンによる方法においては、オブジェクト（Ｏｂｊｅｃｔ）をサブジェクト（Ｓｕｂｊｅｃｔ）とオブザーバー（Ｏｂｓｅｒｖｅｒ）とに分類すると共に、サブジェクトにオブザーバーを登録することにより、この登録に基づいてオブジェクト間を接続し、オブザーバーからサブジェクトに各種通知、データを送出するものである。

この場合、サブジェクトは、登録したオブザーバーに対して、順次指定されたメソド（ｍｅｔｈｏｄ（又はｕｐｄａｔｅ））を呼び出し、オブザーバーにデータの処理を委ねる。これによりこのデザインパターンによる方法において、サブジェクトは、設計時、特定のオブジェクトに対してデータを送出すると言うコーディングをすることなく、独立性の高い設計が可能となる。

ところがこの方法の場合でも、オブザーバーにおいては、特定のサブジェクトに登録することが必要であり、このオブジェクトを事前に記述する必要がある。これによりこのデザインパターンによる方法においては、各オブジェクトの独立性を確保する点において実用上未だ不十分な問題がある。

これに対してこの手法を用いてビジュアルなツールを作成することによりオブ

ジェクトの独立性を図る方法も提案されている（ＪＩＰオブジェクト技術研究会著，　ＶｉｓｕａｌＡｇｅ　Ｃ＋＋　入門　ｐｐ１８２－１８５）。しかしながらこの方法においては、ビジュアルなツールによりオブジェクトの接続を指示した後、ソースコードを自動生成することにより、ソフトウェア構成の変更するためにコンパイルが必要になる問題がある。

また例えば特開平１０－２８３９８号公報に提案されているように、いわゆるプロキシを用いることによりオブジェクト間通信する方法も考えられるが、この方法はオブジェクトの独立性を高めるものではない。

また例えば特開平１０－１７１４６５号公報に開示されているようなクライアントサーバーシステムの命名サービスの方法を利用する方法も考えられる。すなわちこのクライアントサーバーシステムには、オブザーバーパターンと対比されるソフトウェア部品化の方法が適用されており、この方法はＣＯＲＢＡ（Ｃｏｍｍｏｎ　Ｏｂｊｅｃｔ　Ｒｅｑｕｅｓｔ　Ｂｒｏｋｅｒ　Ａｒｃｈｉ－ｔｅｃｔｕｒｅ）においても採用されているものである。

しかしながらこの方法においても、オブザーバーパターンと同様にしてオブザーバーがサブジェクトを指定することにより、オブジェクトの独立性を図る点で実用上未だ不十分な問題があり、また再コンパイル、再リンキングも必要になる問題がある。

本発明は以上の点を考慮してなされたもので、第２に、従来に比してオブジェクトの独立性を向上することができる演算処理装置、オブジェクト間通信方法及びロボットを提案しようとするものである。

かかる課題を解決するため請求の範囲第１項に係る発明においては、ロボットに適用して、取り付けられた部品を検出して部品検出結果を出力する部品検出手段と、部品を交換して得られる形態毎に、部品検出結果に対応する情報を保持する情報保持手段と、部品検出結果と情報保持手段に保持された情報との比較結果に基づいて、形態に応じてソフトウエアを変更するソフトウエア変更手段とを備えるようにする。

かかる課題を解決するため請求の範囲第５項に係る発明においては、演算処理装置に適用して、データ交換可能にオブジェクトを結合するオブジェクトマネージャーと、このオブジェクトマネージャーにオブジェクトの結合に必要な結合情報を提供する結合情報提供手段とを備えるようにする。

また請求の範囲第１１項に係る発明においては、オブジェクト間通信方法に適用して、所定の情報提供手段より提供される結合情報に基づいて、データ交換可能にオブジェクトを結合する。

また請求の範囲第１７項に係る発明においては、ロボットに適用して、データ交換可能にオブジェクトを結合するオブジェクトマネージャーと、オブジェクトマネージャーにオブジェクトの結合に必要な結合情報を提供する結合情報提供手段とを備えるようにする。

また請求の範囲第１項に係る構成によれば、取り付けられた部品を検出して得られる部品検出結果と、部品を交換して得られる形態毎に、部品検出結果に対応する情報との比較結果に基づいて、形態に応じてソフトウエアを変更することにより、部品の変更に伴う形態の変更に対応してソフトウエアを変更することができ、これにより現在の形態に対応する適切なソフトウエアを簡易に設定することができる。

請求の範囲第５項に係る構成によれば、データ交換可能にオブジェクトを結合するオブジェクトマネージャーと、このオブジェクトマネージャーにオブジェクトの結合に必要な結合情報を提供する結合情報提供手段とを備えることにより、オブジェクトにおいては、具体的な結合対象を何ら考慮することなく設計でき、またこのようにして設計されたオブジェクトについて、結合情報を種々に設定、変更して結合を種々に変更することができ、これによりオブジェクトの独立性を従来に比して格段的に向上することができる。

また請求の範囲第１１項に係る構成によれば、所定の情報提供手段より提供される結合情報に基づいて、データ交換可能にオブジェクトを結合すことにより、オブジェクトにおいては、具体的な結合対象を何ら考慮することなく設計でき、

またこのようにして設計されたオブジェクトについて、結合情報を種々に設定、変更して結合を種々に変更することができ、これによりオブジェクトの独立性を従来に比して格段的に向上することができる。

また請求の範囲第１７項に係る構成によれば、データ交換可能にオブジェクトを結合するオブジェクトマネージャーと、オブジェクトマネージャーにオブジェクトの結合に必要な結合情報を提供する結合情報提供手段とを備えるようにすることにより、具体的な結合対象を何ら考慮することなく各オブジェクトを設計でき、またこのようにして設計されたオブジェクトについて、結合情報を種々に設定、変更して結合を種々に変更することができ、これによりオブジェクトの独立性を従来に比して格段的に向上することができる。

図面の簡単な説明

図１は、本発明の実施の形態に係るロボットを示す斜視図である。

図２は、図１のロボットの構成を示すブロック図である。

図３は、４足型におけるソフトウエア構造を示す略線図である。

図４は、タイヤ型におけるソフトウエア構造を示す略線図である。

図５は、４足型における姿勢の変化を示す状態遷移図である。

図６は、入力と状態の関係を示す図表である。

図７は、デザインロボットの処理手順を示すフローチャートである。

図８は、デザインファイルの記述を示す図表である。

図９は、結合情報を示す図表である。

図１０は、ブート後におけるオブジェクトマネージャー、バーチャルロボット、デザインロボットの動作を示すタイムチャートである。

図１１は、プラグイン、プラグアウト等におけるオブジェクトマネージャー、バーチャルロボット、デザインロボットの動作を示すタイムチャートである。

図１２は、オブザーバーとサブジェクトとの関係を示す略線図である。

図１３は、マルチオブザーバーにおけるオブザーバーとサブジェクトとの関係

を示す略線図である。

図１４は、マルチサブジェクトにおけるオブザーバーとサブジェクトとの関係を示す略線図である。

図１５は、オブジェクトマネージャーとオブザーバー、サブジェクトの関係を示す略線図である。

図１６は、結合情報を受け取った場合におけるオブジェクトマネージャーの動作を示すタイムチャートである。

図１７は、ドウスタート以降のシーケンスを示すタイムチャートである。

図１８は、プラグイン、プラグアウト、状態変更におけるシーケンスを示すタイムチャートである。

図１９は、結合情報とオブザーバー、サブジェクトの関係の説明に供する略線図である。

図２０は、連想記憶１の記述を示す図表である。

図２１は、連想記憶２の記述を示す図表である。

図２２は、本発明の第２の実施の形態に係るロボットのタイヤ型におけるソフトウエア構造を示す略線図である。

図２３は、オブジェクトの結合を示す略線図である。

図２４は、形態の変更に係る結合状態の変化を示す略線図である。

図２５は、図２２のロボットにおける起動時の処理手順を示すタイムチャートである。

図２６は、図２５の処理における初期化の説明に供する略線図である。

図２７は、図２５の処理におけるスタートの説明に供する略線図である。

図２８は、図２５の続きを示すタイムチャートである。

図２９は、図２８の処理におけるオブジェクトのロードの説明に供する略線図である。

図３０は、図２８の続きを示すタイムチャートである。

図３１は、図３０の処理におけるコネクトの説明に供する略線図である。

図３２は、サービスマネージャーにおけるデータを示す図表である。

図３３は、図３１の続きの説明に供する略線図である。

図３４は、サービスマネージャーにおける処理の説明に供する略線図である。

図３５は、オブジェクトの結合の説明に供する略線図である。

図３６は、図１９との対比により結合情報とオブザーバー、サブジェクトの関係の説明に供する略線図である。

図３７は、図２８との対比により各オブジェクトの結合を示す略線図である。

図３８は、図３０におけるスタートの説明に供する略線図である。

図３９は、図２２のロボットにおける形態変更時の処理を示すタイムチャートである。

図４０は、図３９の処理における形態変更の通知を示す略線図である。

図４１は、図３９の処理におけるストップの処理を示す略線図である。

図４２は、図３９の続きの処理を示すタイムチャートである。

図４３は、図４２の処理におけるオブジェクトの切り離しを示す略線図である。

図４４は、図４３の続きの処理を示す略線図である。

図４５は、図４２の続きの処理を示すタイムチャートである。

図４６は、形態の変更に係るオブジェクトの変更を示す略線図である。

図４７は、形態変更後の各オブジェクトの関係を示す略線図である。

図４８は、サービスマネージャーの処理を示す略線図である。

図４９は、コピーによるオブジェクト間通信の説明に供する略線図である。

図５０は、共有メモリによるオブジェクト間通信の説明に供する略線図である。

図５１は、共有メモリの監視の説明に供する略線図である。

図５２は、複数のデータをまとめたデータ通信の説明に供する略線図である。

発明を実施するための最良の形態

以下、図面について本発明の一実施の形態を詳述する。

（１）第１の実施の形態の構成

（１－１）全体構成

図１は、本発明の実施の形態に係るロボットを示す斜視図である。このロボット１は、いわゆるペット型のロボットであり、全体が犬に似た形状であり、４本脚により歩行できるようになされている。すなわちロボット１は、本体２に移動ユニット３が取り付けられ、この本体２が頭部２Ａと胴体部２Ｂとにより構成される。

頭部２Ａは、マイク４、タッチセンサー５、立体視可能なテレビジョンカメラ６、遠隔制御に係る赤外線の受信部７等が配置され、これらにより各種情報を取得できるようになされている。また頭部２Ａは、画像表示部９、スピーカー１０が配置され、これらにより種々の情報を出力できるようになされている。

胴体部２Ｂは、背中の部分に操作子１１が配置され、また尻尾に相当する箇所にアンテナ１２が配置されるようになされ、これらを介してユーザーの操作を検出できるようになされている。また胴体２Ｂは、スロットを有し、このスロットにＩＣカードを装着してソフトウエアのバージョンアップ等を実行できるようになされている。

さらに胴体部２Ｂは、頭部２Ａを介して入出力する各種情報、操作子１１等の操作情報を処理して全体の動作を制御するコントローラ１５、各部に電源を供給する電源ユニット１４、アンテナ１２を介して各種情報を送受する通信ユニット１６等が配置され、下部にバッテリー１７が配置されるようになされている。

これに対して移動ユニット３は、それぞれ関節に相当する箇所にアクチュエータ、角度検出センサーを配置してなる４つの足を配置して形成され、本体２とシリアルバスにより接続され、本体２の制御により動作するようになされている。これによりロボット１は、４足歩行により移動できるようになされている。

さらにこの移動ユニット３は、後足が本体２より取り外すことができるようになされている。ロボット１は、この後足に代えて、タイヤを備えた移動ユニット

を取り付けることができるようになされている。これによりこのロボット１は、ユーザーの好みに応じて４足により移動する形態（以下４足型と呼ぶ）と、タイヤにより移動する形態（以下タイヤ型と呼ぶ）とで形態を変更できるようになされている。

図２は、このロボット１の構成を示すブロック図である。ロボット１においては、処理プログラムを記録したプログラムメディア２０がフェリペラル２１を介して中央処理ユニット２２に接続され、この中央処理ユニット２２によりプログラムメディア２０に記録された処理手順を実行するようになされている。また中央処理ユニット２２は、フェリペラル２１を介して移動ユニット３の各アクチュエーター、センサー、さらにはテレビジョンカメラ６、操作子１１等であるロボットコンポーネント２４、２５、２６と接続され、これにより全体の動作を制御できるようになされている。

また中央処理ユニット２２は、同様に、フェリペラル２１を介して電源ユニット１４を構成する電源制御用の１チップマイコン２７と接続され、この１チップマイコン２７の動作を制御してバッテリー１７より全体に電源を供給し、さらには省電力モードにおいてバッテリー１７に代えてボタン電池２９より全体の電源を供給するようになされている。

（１－２）ソフトウエア構成

図３は、ロボット１の４足型におけるソフトウエア構成をレイヤーにより示す略線図であり、図４は図３との対比によりタイヤ型におけるソフトウエア構成を示す略線図である。このソフトウエア構成において、最も下位層はデバイスドライバレイヤーであり、各種デバイスドライバが配置される。ここでデバイスドライバは、移動ユニット３のセンサー、アクチュエータ、さらには頭部２Ａ等に配置した各種ロボットコンポーネントとの間で各種情報を入出力し、これらの処理のためのソフトウェアオブジェクトを備えるようになされている。

このうちバッテリーマネージャーデバイスドライバ（バッテリーマネージャーＤＤ）は、電源ユニット１４の１チップマイコン２７との間のデータ通信により

バッテリー１７の残量を検出して定期的に上位のロボットシステムソフトウエアに通知する。また中央処理ユニット２２等のクロックを管理し、ロボットシステムソフトウエアの指示によりこのクロックの周波数を低減すると共に、不必要なデバイスの動作を停止し、これにより全体の消費電力を低減する。

またロボットシリアルバスデバイスドライバ（ロボットシリアルバスＤＤ）は、本体２に配置されたシリアルバスのマスター制御用のハードウェアとの間のデータ通信により、シリアルバスに接続された移動ユニット３のセンサー情報等を取り込んで上位のロボットシステムソフトウエアに通知し、またこれとは逆にロボットシステムソフトウエアより出力されるアクチュエータの制御データ、オーディオ信号等を各デバイスに送出する。

またロボットシリアルバスデバイスドライバは、システムの起動時、シリアルバスに接続されたデバイスの変更（追加、削除である）を検出する。さらにこの変更に係るデバイス固有の情報を取り込むことにより、ロボット１の現在の形態を示す情報をロボットシステムソフトウエアに出力する。

このデバイスドライバレイヤーの上位には、ロボットシステムソフトウエアが配置される。ここでロボットシステムソフトウエアは、バーチャルロボット、デザインロボット、パワーマネージャー、オブジェクトマネージャー等のソフトウエアオブジェクトより構成される。

このうちバーチャルロボットは、ロボットシリアルバスデバイスドライバとの間のデータ通信において、各デバイス固有のフォーマットによるデータをロボット一般的なフォーマットに変換して送受する。具体的に、例えばモータ制御用のデータについて、デバイス固有のフォーマットにおいてはポテンションメーターであるセンサー出力をアナログディジタル変換して得られる１０ビットにより表現するのに対し、バーチャルロボットは、この表現によるデータを最下位の１ビットが０．００１度を示すデータに変換してロボット１における一般的なフォーマットとして出力する。またバーチャルロボットは、このようなフォーマット変換によるデータの送受に加えて、例えばテレビジョンカメラ６より取得したイメ

ージデータを上位のソフトウエアに出力する。

またバーチャルロボットは、ロボットシリアルバスデバイスドライバよりロボット１の現在の形態を示す情報を受け取り、この情報を取りまとめる。これによりバーチャルロボットは、ロボット１全体でどの様なロボットコンポーネントがどの様な順番で接続されているのかを示す結合情報（ＣＰＣ（Ｃｏｎｆｉｇｕｒａｂｌｅ Ｐｈｙｓｉｃａｌ Ｃｏｍｐｏｎｅｎｔ）Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）を管理し、この結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）をデザインロボットに通知する。

またバーチャルロボットは、シリアルバスに接続されたデバイスに変更があった場合、デバイスドライバのロボットシリアルバスデバイスドライバより通知を受け、デザインロボットに通知する。

デザインロボットは、バーチャルロボットより結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）を受け取り、事前に記録した結合テンプレート情報と順次比較することにより、現在のロボット１の形態に適したテンプレートを選択する。さらにデザインロボットは、この選択したテンプレートに応じて、上位の形態依存ソフトウエアを現在の形態に適した形態依存ソフトウエアに更新するように、オブジェクトマネージャーに指示する。

なおこの実施の形態において、結合テンプレート情報は、デザインファイルにデザインデータとして記述されるようになされている。

オブジェクトマネージャーは、デザインロボットの指示により、デザインロボットより通知されるオブジェクト、各オブジェクトの結合に関する情報を用いて形態依存ソフトウエアを現在の形態に適した形態依存ソフトウエアに更新する。すなわちオブジェクトマネージャーは、形態依存ソフトウエアを構成する全てのオブジェクトについて、動作の停止、オブジェクト間通信の結合解除、オブジェクトの破壊、リソースの開放を指示する。さらに新たに必要なオブジェクトのロード、初期化し、さらにはオブジェクト間通信のための結合構築、これらオブジェクトの起動を指示する。なおここでオブジェクト間の接合とは、オブジェクト

間でデータを送受信するための結合を意味する。

パワーマネージャーは、バッテリーマネージャーデバイスドライバとの間でデータ通信し、上位のソフトウエア（形態独立ソフトウエア又は形態依存ソフトウエア）の指示に従って、バッテリーマネージャーデバイスドライバにクロックの切り換えを指示し、またオブジェクトの動作の停止等を指示する。

このロボットシステムソフトウエアの上位には、形態依存ソフトウエアが配置され、さらにその上位に形態独立ソフトウエアが配置される。ここで形態依存ソフトウエアは、このロボット１の形態に応じて変更されるソフトウエアであり、形態独立ソフトウエアは、これとは逆に、ロボットの形態に係わらずに固定されたこのロボット１の上位のソフトウエアである。このため形態依存ソフトウエアは、ロボット１の形態に依存した各種オブジェクトが存在する。この実施の形態においては、形態依存ソフトウエアの変更により４足型とタイヤ型とでそれぞれ形態に適したソフトウエアを全体として簡易に構成できるようになされている。

すなわちここで４足型において、この形態依存ソフトウエアは（図３）、移動ユニット３に関連するものとしてコマンドコンバータ、モーションネットワーク、モーションリプレイ、ウォーキングパターンジェネレータ、ビジョン等のオブジェクトにより構成される。

このうちコマンドコンバータは、形態に依存しない形態独立ソフトウエアからのコマンドをロボット１の形態に適したコマンドに変換する。すなわち形態独立ソフトウエアから移動ユニット３に関するコマンドとして、それぞれ睡眠（Ｓｌｅｅｐ）、休憩（Ｒｅｓｔ）、起床（Ｗａｋｅｕｐ）、移動（Ｍｏｖｅ）の姿勢を指示するコマンドが入力されると、コマンドコンバータは、この４足型において、これらのコマンドをそれぞれ睡眠中（Ｓｌｅｅｐｉｎｇ）、お座り（Ｓｉｔｔｉｎｇ）、起立（Ｓｔａｎｄｉｎｇ）、歩行（Ｗａｌｋｉｎｇ）の各姿勢を指示するコマンドに変換する。

モーションネットワークは、コマンドコンバータにより出力されるこれら姿勢を示すコマンドより、図５に示す遷移に従って、この遷移図の矢印に対応するオ

ブジェクトのメソド（エントリ）を起動する。すなわち睡眠中の状態で歩行のコマンドが入力された場合、お座り、起立、歩行と姿勢を順次変化させるように、モーションリプレイオブジェクトの対応するエントリを順次起動する。なおこの場合、最後の歩行の状態に辿り着いたときには、この状態を示す自己ループに対応するウォーキングパターンジェネレータのエントリを起動する。

モーションリプレイは、モーションネットワークによるエントリの起動により、対応する姿勢変化用の関節角度指令値を出力する。具体的に、モーションリプレイは、登録された離散的な関節角度指令値をキーデータとして保持し、このキーデータを用いた補間演算処理により一連の角度指令値を出力する。

ウォーキングパターンジェネレータは、形態独立ソフトウエアにより指定される方向に移動するように、各関節角度指令値を計算して出力する。

ビジョンは、例えば画像処理用のオブジェクトであり、テレビジョンカメラ６で取得したイメージデータをバーチャルロボットオブジェクトより受け取り処理する。ビジョンは、これにより例えば赤色の物体を識別し、このロボット１を基準にしたこの赤色の物体の位置、見かけの大きさを検出して出力する。

これに対してタイヤ型において、形態依存ソフトウエアは（図４）、移動ユニット３に関するオブジェクトとして、４足型と共通のビジョンのオブジェクト、このタイヤ型に特有のコマンドコンバータ、モーションコマンダ、モーションリプレイ２、ホイールパターンジェネレータのオブジェクトを有する。

このうちコマンドコンバータは、４足型におけるコマンドコンバータと同様に、形態に依存しない形態独立ソフトウエアからのコマンドをロボット１の形態に適したコマンドに変換する。この場合、ロボット１においてはタイヤ型であることにより、コマンドコンバータは、それぞれ睡眠（Ｓｌｅｅｐ）、休憩（Ｒｅｓｔ）、起床（Ｗａｋｅｕｐ）、移動（Ｍｏｖｅ）の姿勢を指示するコマンドを、睡眠中（Ｓｌｅｅｐｉｎｇ）、休憩（Ｒｅｓｔ）、待機（Ｒｅａｄｙ）、移動（Ｇｏ）のコマンドに変換する。

モーションコマンダは、コマンドコンバータの出力を受け、タイヤ用の制御コ

マンドと関連する関節角度指令値を生成する。すなわちモーションコマンダは、睡眠中（Ｓｌｅｅｐｉｎｇ）のコマンドが入力された場合、関節角度指令値を中立（ｎｅｕｔｒａｌ）に設定する。ここで中立（ｎｅｕｔｒａｌ）においては、何らモータを駆動、制動しない状態であり、これにより移動ユニットを構成する両腕にあっては、垂れ下がった状態に保持され、またタイヤにあっては停止状態に保持される。

これに対して休憩（Ｒｅｓｔ）のコマンドが入力された場合、モーションコマンダは、両腕にあっては肘の部分で９０度折り曲げて前に差し出すように、また頭部２Ａにあっては顔を正面に向けるように、関節角度指令値を生成する。なおタイヤは停止したままである。

また待機（Ｒｅａｄｙ）のコマンドが入力された場合、モーションコマンダは、両腕にあっては肘を伸ばして前に差し出すように、また頭部２Ａにあっては顔を正面に向けるように関節角度指令値を生成する。なおこの場合もタイヤは停止したままである。さらに移動（Ｇｏ）のコマンドにあっては、待機（Ｒｅａｄｙ）の状態と同様に両腕を保持したまま、タイヤが前進方向に回転するように、指令値を出力する。

モーションリプレイ２は、モーションコマンダの指示によりタイヤ型に固有の動作に関して、タイヤ以外の動作（例えば両腕の動作である）を制御する。ホイールパターンジェネレータは、モーションコマンダの制御によりタイヤの制御情報を生成する。

形態独立ソフトウエアは、ロボット１の形態に依存しないソフトウエアであり、形態設定オブジェクト等を有する。ここで形態設定オブジェクトは、形態依存ソフトウエアよりロボット１の形態に依存しない情報を受け取り、この情報によりロボット１の形態に依存しない上述したようなロボット１の姿勢を示すコマンド（睡眠等である）を形態依存ソフトウエアに出力する。

すなわち例えば、ビジョンオブジェクトより通知される赤色の物体の有無とその位置情報に基づいて、１分以上赤色の物体の存在が検出されなかった場合、図

６に示すように、睡眠（Ｓｌｅｅｐ）の状態にあってはその状態を維持するように、休憩（Ｒｅｓｔ）の状態にあっては睡眠（Ｓｌｅｅｐ）に、起床（Ｗａｋｅｕｐ）の状態にあっては休憩（Ｒｅｓｔ）に、移動（Ｍｏｖｅ）にあっては起床（Ｗａｋｅｕｐ）の状態に遷移するようにコマンドを発行する。

また赤色物体が所定値より小さな大きさで検出され（ロボットの設定による値１０以下)、これが１０秒以上続いた場合、睡眠（Ｓｌｅｅｐ）の状態にあっては休憩（Ｒｅｓｔ）の状態に、休憩（Ｒｅｓｔ）の状態にあっては起床（Ｗａｋｅｕｐ）に、起床（Ｗａｋｅｕｐ）の状態にあっては移動（Ｍｏｖｅ）に遷移するようにコマンドを発行し、また移動（Ｍｏｖｅ）にあっては移動（Ｍｏｖｅ）を継続するようにコマンドを発行する。

さらに赤色物体が所定値以上の大きさで検出された場合、睡眠（Ｓｌｅｅｐ）の状態にあっては（Ｗａｋｅｕｐ）の状態に遷移するように、休憩（Ｒｅｓｔ）及び起床（Ｗａｋｅｕｐ）の状態にあっては移動（Ｍｏｖｅ）に遷移するように、また移動（Ｍｏｖｅ）にあっては移動（Ｍｏｖｅ）を継続するようにコマンドを発行する。

これらによりロボット１では、最も上位のソフトウエアが、形態に依存しないように形成され、この形態に依存しないソフトウエアとの間では形態に依存しない形式によりデータを送受するようになされている。これによりこのロボット１では、形態に応じて、この形態に依存するソフトウエアである形態依存ソフトウエアだけを変更して、形態に応じてソフトウエアを簡易に変更することができるようになされている。

（１－３）形態依存ソフトウエアの変更

ロボット１において、この形態依存ソフトウエアは、デザインロボットの指示によるオブジェクトマネージャーの処理により更新される。このときこの実施の形態においては、形態依存ソフトウエアを構成するオブジェクトのロード、アンロード、オブジェクト間通信のための結合再構築により形態依存ソフトウエアを更新する。

このためデザインロボットは、バーチャルロボットよりロボット１の形態を特定する結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）の通知を受け、この結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）に基づいてオブジェクトマネージャーに形態依存ソフトウエアの更新を指示すると共に必要な情報を提供する。

またバーチャルロボットにおいては、下位層であるロボットシリアルバスデバイスドライバを介してシリアルバスに接続された各デバイスの情報を取得し、この情報より結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）を作成してデザインロボットに通知する。

このためこのロボット１においては、各ロボットコンポーネントのメモリに所定の情報が記録され、ロボットシリアルバスデバイスドライバを介して実行されるデータ通信により、バーチャルロボット１においてシリアルバスに接続されたデバイスの情報をその位置情報と共に取得できるようになされている。

ここでロボット１においては、シリアルバスに接続されるテレビジョンカメラ、スピーカー、マイク、各種アクチュエータ、センサー等が基本的な構成要素（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ）であり、リンク状態に設定されたこれら基本的な構成要素の複数の組み合わせにより各部品（ＣＰＣ　Ｍｏｄｅｌ）が構成される。すなわち例えば部品である１つの足は、３個のモータ、１個のスイッチによる複数の基本的な構成要素の特定なリンク状態により構成される。

ロボット１においては、これら各部品に固有の識別データが割り当てられ、この識別データによりシリアルバスに接続された各構成要素を特定できるようになされている。なおこの識別データは、部品を作成した製造工場のＩＤと製品番号に対応するＩＤとにより構成される。またこの識別データに対応するように、各構成要素の情報、部品における各構成要素の位置情報が部品中における各構成要素のリンク情報、各構成要素の属性情報等により設定され、これらの情報が各部品の識別データと共に各部品のメモリに保持されるようになされている。なおこの実施の形態では、部品側にてメモリ容量が少ない場合には、各部品側に識別デ

ータのみ記録され、構成要素の情報、部品における構成要素の位置情報は、本体2側に保持するようになされている。

バーチャルロボットは、システム起動時、デバイスの変更時、ロボットシリアルバスデバイスドライバより通知されるこれら識別データ、位置情報等により、ツリー構造によるシリアルバスを順次辿って、各基本的な構成要素（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ）と各構成要素の結合の順番を示すデータ構造とにより結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）を作成してデザインロボットに通知する。かくするにつきバーチャルロボットより通知される結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）にあっては、形態に対応する構成要素情報（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ Ｌｏｃａｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）により構成される。

デザインロボットは、バーチャルロボットより通知される結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）を基準にしてデザインファイルを参照することにより現在のロボット１の形態に対応する結合テンプレート（後述するラベルである）を選択する。さらにデザインロボットは、この選択したテンプレートに応じて、現在の形態に適した形態依存ソフトウエアへの更新をオブジェクトマネージャーに指示する。

ここでデザインファイルは、テキストデータ形成の記述であり、ロボット１の形態毎に、各部品の構成要素情報（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ Ｌｏｃａｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）群に対してラベルを付加して記述される。また構成要素情報（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ Ｌｏｃａｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）は、各部品の基本的な構成要素（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ）と各構成要素の位置情報（ＣＰＣ Ｃｏｏｄｉｎａｔｅ Ｌｏｃａｔｏｒ）により構成される。このうち位置情報（ＣＰＣ Ｃｏｏｄｉｎａｔｅ Ｌｏｃａｔｏｒ）は、ロボット１に設定された基準位置からの座標であり、一般的なリンク座標系により表記され、座標系変換のための回転行列と位置ベクトルの縦続接続により構成されるようになされている。

またラベルは、対応するオブジェクトの特定、このオブジェクトのロード、オブジェクト間通信の構築に必要なデータを特定するものであり、デザインラベル（Ｄｅｓｉｇｎ　Ｌａｂｅｌ)、又はデザインラベルに比して汎用的な理解容易な記述によるバーチャルラベル（Ｖｉｒｔｕａｌ　Ｌａｂｅｌ)、デザインラベル（Ｄｅｓｉｇｎ　Ｌａｂｅｌ）とバーチャルラベル（Ｖｉｒｔｕａｌ　Ｌａｂｅｌ）との複合型のラベル（Ｃｏｍｐｏｓｉｔｅ　Ｌａｂｅｌ）が適用される。なおビジュアルラベルにあっては、デザインロボットによりデザインラベルに置き換えられて利用に供されるようになされている。

次式は、このうちのデザインラベルの記述を示すものである。デザインラベルは、１つ以上の構成要素情報（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ　Ｌｏｃａｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ)、又はデザインラベル（Ｄｅｓｉｇｎ　Ｌａｂｅｌ）により構成される。

☆

Design Label (

CPC Primitive Location Information 又は Design Label ;

.....

) ..... (1)

★

これに対して次式は、バーチャルラベル（Ｖｉｒｔｕａｌ　Ｌａｂｅｌ）の記述を示すものである。バーチャルラベルは、デザインラベル（Ｄｅｓｉｇｎ　Ｌａｂｅｌ）、構成要素（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ）、構成要素（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ）の位置情報（ＣＰＣ　Ｃｏｏｄｉｎａｔｅ　Ｌｏｃａｔｏｒ）により構成される。

☆

Virtual Label (

Design Label :

CPC Primitive :

CPC Coordinate Locator :

) ……（２）

★

これに対して次式は、複合型のラベル（Ｃｏｍｐｏｓｉｔｅ　Ｌａｂｅｌ）の記述を示すものである。

☆

Composite label（

Composite Label ;

or/and Design Label :

or/and Virtual Label ;

）　……（3）

★

かくするにつきデザインロボットは、通知された結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）により特定される構成要素情報（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ　Ｌｏｃａｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）において、特定される位置情報（ＣＰＣ　Ｃｏｏｄｉｎａｔｅ　Ｌｏｃａｔｏｒ）よりツリー構造の先端側にバーチャルラベル（Ｖｉｒｔｕａｌ　Ｌａｂｅｌ）内で定義された位置情報（ＣＰＣ　Ｃｏｏｄｉｎａｔｅ　Ｌｏｃａｔｏｒ）が存在する場合、このバーチャルラベル（Ｖｉｒｔｕａｌ　Ｌａｂｅｌ）をデザインラベル（Ｄｅｓｉｇｎ　Ｌａｂｅｌ）に置き換え、またこのようにして定義したデザ

インラベル（Ｄｅｓｉｇｎ　Ｌａｂｅｌ）を有効なものと設定する。

これらによりこの実施の形態では、結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）によりこのデザインファイルをアクセスして現在のロボットを形態（４足型、タイヤ型等の形態）を特定できるようになされ、また形態依存ソフトウエアの更新に必要な各種データを取得できるようになされている。

図７は、このデザインロボットの処理手順を具体的に示すフローチャートである。デザインロボットは、バーチャルロボットより結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）が通知されると、ステップＳＰ１からステップＳＰ２に移り、この結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）によりデザインファイルをアクセスし、これにより現在のロボット１の形態を特定する。

続いてデザインロボットは、ステップＳＰ３に移り、バーチャルロボットより通知された結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）による構成要素情報（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ　Ｌｏｃａｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）とデザインファイルに保持された結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）による構成要素情報（ＣＰＣ　Ｐｒｉｍｉｔｉｖｅ　Ｌｏｃａｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）とを比較する。

続いてデザインロボットは、ステップＳＰ４に移り、この比較結果に基づいて、デザインファイルをアクセスし、現在の形態に対応するオブジェクト、オブジェクト間通信の再構築に必要なデータを特定するデザインラベルを検出する。

続いてデザインロボットは、ステップＳＰ５に移り、検出したデザインラベルより結合ファイル（Ｃｏｎｎｅｃｔｉｏｎ　Ｆｉｌｅ）をアクセスし、ラベルに対応する結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を検出することにより、対応するオブジェクトの特定、オブジェクト間通信の再構築に必要なデータを取得する。かくするにつき結合ファイル（Ｃｏｎｎｅｃｔｉｏｎ　Ｆｉｌｅ）は、

ラベルとの対応により結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を記録したファイルであり、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）は、対応するオブジェクトの特定、オブジェクト間通信の再構築に必要なデータである。

続いてデザインロボットは、ステップＳＰ６に移り、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）をオブジェクトマネージャーに送出し、これにより形態依存ソフトウエアの更新を指示した後、ステップＳＰ７に移ってこの処理手順を終了する。

図８は、デザインファイルの一部の記述を示す図表である。このデザインファイルにおいて、ＤＲＸは、このロボット１の形態を示し、４Ｌｅｇｇｅｄは、４足型の形態であることを示す記述である。またＨｅａｄ、ＲｉｇｈｔＦｒｏｎｔＬｅｇ、ＬｅｆｔＦｒｏｎｔＬｅｇ、ＲｉｇｈｔＲｅａｒＬｅｇ、ＬｅｆｔＲｅａｒＬｅｇは、それぞれ頭部、右前足、左前足、右後足、左後足を示し、これにより第１文において４足型が頭部、右前足、左前足、右後足、左後足により構成されることが特定される。

また第２文におけるＷｈｅｅｌはタイヤ型を示し、第１文との対比によりこの第２文においてタイヤ型が頭部、右前足、左前足、右後輪、左後輪により構成されることが特定される。また第３文においては、頭部にカメラリンクの配置が記述され、第４文にはこのカメラリンクのバーチャルラベル（Ｖｉｒｔｕａｌ　Ｌａｂｅｌ）が記述されるようになされ、第５文以降においては、右前足、左前足等について、位置情報（ＣＰＣ　Ｃｏｏｄｉｎａｔｅ　Ｌｏｃａｔｏｒ）がツリー形式により記述されるようになされている。

これに対して図９は、結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）の一部の記述を示す図表である。この実施の形態においては結合情報もデザインファイルと同様にテキストデータ形成により記述される。この記述において、ｊｏｉｎｔはインターフェースを示し、この部分が他の構成要素との結合箇所であることを示すものである。この場合第２行目から第５行目までの記載により、Ｃ０、Ｃ０、Ｃ０、Ｃ０、Ｃ０による５つのハブ等を介して

テレビジョンカメラが接続されていることが判る。

これによりこの図９の例による結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）にあっては、図８について上述した記述である４足型と一致することになる。

これに対して次式は、結合ファイル（Ｃｏｎｎｅｃｔｉｏｎ　Ｆｉｌｅ）の一部の記述を示す図表である。結合ファイルは、テキストデータ形成により記述され、この記述において、第１文及び第２文は、それぞれ４足型及びタイヤ型における足及びタイヤに対応するオブジェクトと、このオブジェクトにおけるオブジェクト間通信の構築に必要なデータが記述される。

すなわち第１文にあっては、サービス名としてオブジェクト名であるＭｏＮｅｔとデータ形式ＭｏＮｅｔＯｕｔＤａｔａ、サブジェクトを示すＳの記述が記述される。さらに同様のサービス名としてオブジェクト名であるＭｏＮｅｔＲｅｐｌａｙ、データ形式ＭｏＮｅｔＯｕｔＤａｔａ等が記述され、最後にオブザーバーを示すＯの記述が割り当てられる。また続く行において、同様のサブジェクト及びオブザーバー、データ形式等が記述される。

☆

DRX 4Legged（

Monet.MoNetOutData..S,MoNetReplay.MoNetOutData...O

MoNet.MoNetOutData..S,WalkingPatternGenerator.MoNetOutData...O

)

DRX Whell (

MotionConverter.MCData..S,MotionReplay2.MCData...O

MotionConverter.MCData..S,Whell.MCData..O

)

…… (4)

★

これによりこの図８の例の場合、デザインロボットは、このＤＲＸ　４Ｌｅ－ｇｇｅｄによる第１文の記述より結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）（ＤＲＸ　４Ｌｅｇｇｅｄ以降の括弧書の記述）を抽出してオブジェクトマネージャーに通知し、これによりこの実施の形態では形態依存ソフトウエアを更新するようになされている。

図１０は、オブジェクトマネージャー、バーチャルロボット、デザインロボットにおけるブート時の処理を示すタイムチャートである。ロボット１においては、電源が投入されると、又はリセットがされると、ブートの処理を開始する。ここでロボット１においては、各オブジェクトがファイルよりロードされて形態依存ソフトウエアが構築され、このとき各オブジェクトマネージャーの内部変数が

初期化される。

続いてオブジェクトマネージャーよりドウイニシャライズ（ＤｏＩｎｉｔ）が各オブジェクトに送出され、このドウイニシャライズ（ＤｏＩｎｉｔ）により各オブジェクトの問い合わせ用エントリ（Ｅｎｔｒｙ）がオブジェクトマネージャー等に登録される。これによりロボットシステムソフトウエアレイヤーにおいて、各オブジェク間通信が構築される。

続いてオブジェクトマネージャーよりドウスタート（ＤｏＳｔａｒｔ）がバーチャルロボット、デザインロボットに順次送出され、バーチャルロボット、デザインロボットが動作を開始する。これによりバーチャルロボットにより結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）が生成され、デザインロボットの要求によりこの結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）がデザインロボットに渡され、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）が生成される。

さらにこの結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）がオブジェクトマネージャーに渡され、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）により形態依存ソフトウエアが構築される。

これに対してシャットダウンにおいては、オブジェクトマネージャーよりドウストップ（ＤｏＳｔｏｐ）が各オブジェクトに送出され、このドウストップ（ＤｏＳｔｏｐ）により各オブジェクトにおいては、全てのリクエストに対してエラーを通知して動作を停止する。続いてオブジェクトマネージャーよりドウデストロイ（ＤｏＤｅｓｔｒｏｙ）が各オブジェクトに送出され、これにより各オブジェクトにおいては、使用していたソースを開放し、またオブジェクトマネージャーに登録したエントリを抹消して待機する。

これによりロボット１においては、形態が変更されて立ち上げられた場合でも、この変更された形態に適したソフトウエアにより動作できるようになされている。

これに対していわゆるプラグイン（Ｐｌｕｇ－Ｉｎ）、プラグアウト（

Ｐｌｕｇ－Ｏｕｔ）においては、図１１に示すように、オブジェクトマネージャーよりドウストップ（ＤｏＳｔｏｐ）が各オブジェクトに送出され、このドウストップ（ＤｏＳｔｏｐ）により各オブジェクトにおいては、全てのリクエストに対してエラーを通知して動作を停止する。

続いてオブジェクトマネージャーよりブート時と同様にして順次ドウスタート（ＤｏＳｔａｒｔ）がバーチャルロボット、デザインロボットに順次送出され、バーチャルロボットにより結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）が生成され、この結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）がデザインロボットに渡されて結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）が生成され、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）により形態依存ソフトウエアが構築される。

これによりロボット１においては、プラグイン（Ｐｌｕｇ－Ｉｎ）、プラグアウト（Ｐｌｕｇ－Ｏｕｔ）により動的に形態が変更された場合でも、この変更された形態に適したソフトウエアにより動作できるようになされている。

また、バッテリー１７が消耗した場合等にあっては、バッテリーマネージャーからの状態変更要求により、プラグイン（Ｐｌｕｇ－Ｉｎ）、プラグアウト（Ｐｌｕｇ－Ｏｕｔ）の場合と同様に、ドウストップ（ＤｏＳｔｏｐ）、ドウスタート（ＤｏＳｔａｒｔ）の処理が繰り返され、これによりバッテリー１７が消耗した場合にあっては、クロック周波数を低減すると共に不必要なデバイスの動作を停止してボタン電池２９により動作するように状態が変更され、これとは逆に充電されたバッテリー１７が装着された場合にあっては、クロック周波数を増大すると共に各デバイスの動作が開始されてバッテリー１７により動作するようになされている。

（１－４）オブジェクトマネージャーの処理

オブジェクトマネージャーは、このようにして実行されるデザインロボットからの通知により形態依存ソフトウエアを構成するオブジェクトを再構成し、これにより形態依存ソフトウエアを更新する。具体的に、オブジェクトマネージャー

は、デザインロボットからの通知によりオブジェクトをロード、アンロードすると共に、これらアンロード及びロードに対応するようにオブジェクト間通信を再構築し、これにより形態依存ソフトウエアを変更する。

このオブジェクト間通信を構築する際に、対応するオブジェクト名等がオブジェクトに登録されていなければならないとすると、各オブジェクトの独立性が害され、自由な形態に対応するようにオブジェクトを複数種類用意することが必要になる。このためこの実施の形態において、オブジェクトマネージャーは、デザインロボットより出力される結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に基づいてオブジェクト間通信を構築し、これによりオブジェクトの独立性を担保する。

なお以下の説明においては、ロボット１における具体的なオブジェクト名の特定を避けて説明を簡略化する。またオブジェクト間通信は、非同期により実行するものとする。

すなわち図１２は、この実施の形態に係る２つのオブジェクト間におけるオブジェクト間通信の基本構成を示す略線図である。この実施の形態においては、レディー（Ｒｅａｄｙ）とノーティファイ（Ｎｏｔｉｆｙ）とを用いてオブジェクト間通信を実行し、これによりオブジェクト（この例ではＯｂｊｅｃｔＢ）の処理能力以上のデータを送出しないようになされている。

すなわち図１２において、オブジェクトＡは、オブジェクトＡのメンバーであるクラス０のサブジェクトからオブジェクトＢのメンバーである０オブザーバークラスに対してデータを送出し、これによりオブジェクトＢのメソドがインボウク（ｉｎｖｏｋｅ）される。

この関係において、サブジェクトは、オブザーバーよりレディー（Ｒｅａｄｙ）の信号が送出されている場合にのみデータを送出する。なおレディー（Ｒｅａｄｙ）の信号は、１回のデータにつき１回送る必要がある。これによりオブザーバー側であるオブジェクトＢにおいては、受け取ったデータの処理の完了を待って続くデータを受信することができるようになされている。

これに対して図１３は、マルチオブザーバーにおけるオブジェクト間通信の基本構成を示す略線図である。この場合、サブジェクトであるオブジェクトＡは、全てのオブザーバーに対してデータを配信することができ、またオブジェクトＩＤにより特定される特定のオブザーバーに対してのみにデータを配信することもできる。なおこの実施の形態においては、このようなマルチオブザーバーの状態においても、レディー（Ｒｅａｄｙ）の信号により対応するオブザーバーに対してデータを送出する。

各オブジェクトにおいては、オブジェクトを特定するオブジェクトＩＤ、メソド（ｍｅｔｈｏｄ）を特定するセレクタ番号(ｓｅｌｅｃｔｏｒ)（メソド番号である）を指定することにより、対応するメソドを起動（エントリ）し、オブザーバーを特定して所望のデータを送出する。

図１４は、いわゆるマルチサブジェクトにおけるオブジェクト間通信の基本構成を示す略線図である。この場合、オブジェクトＡ及びＢがサブジェクトであり、オブジェクトＣがオブザーバーである。オブザーバーにおいては、このように複数のサブジェクトよりそれぞれデータを受信することができ、またデータを受信する毎にデータを処理するメソドがインボウク（ｉｎｖｏｋｅ）されるようになされている。またオブザーバーは、サブジェクトＩＤにより特定される特定のサブジェクトに対してのみにレディー（Ｒｅａｄｙ）の信号を送出して所望のデータを選択的に入力できるようになされている。

この実施の形態において、少なくとも形態依存ソフトウエアレイヤーの属するオブジェクトにおいては、これら図１２～図１４を用いて説明したように、レディー（Ｒｅａｄｙ）とノーティファイ（Ｎｏｔｉｆｙ）とを用いてオブジェクト間通信を実行し、また必要に応じてマルチオブザーバー、マルチサブジェクトの結合形態により他のオブジェクトを結合できるようになされている。すなわち複数のオブザーバーを有するオブジェクトにあっては、これら複数のオブザーバーに対応する数のコネクトエントリーを有するようになされている。

図１５は、オブジェクトマネージャーと各オブジェクトとの関係を示す略線図

である。各オブジェクトにおいては、それぞれオブジェクトＩＤ、セレクタ番号（Ｓｅｌｅｃｔｏｒ）によりメソドを特定してデータ交換し、それぞれ対応するサブジェクト、オブザーバーとは何ら無関係に、基本的なメソドについてセレクタ番号０～３がそれぞれ割り当てられるようになされている。

ここでセレクタ番号０には、ドウイニシャライズ（ＤｏＩｎｉｔ）が割り当てられ、各オブジェクトは、このドウイニシャライズ（ＤｏＩｎｉｔ）の起動により初期化されるようになされている。またセレクタ番号１には、ドウスタート（ＤｏＳｔａｒｔ）が割り当てられ、各オブジェクトは、このドウスタート（ＤｏＳｔａｒｔ）の起動により動作を開始するようになされている。さらにセレクタ番号２には、ドウストップ（ＤｏＳｔｏｐ）が割り当てられ、各オブジェクトは、このドウストップ（ＤｏＳｔｏｐ）の起動により動作を停止するようになされている。またセレクタ番号２には、ドウデストロイ（ＤｏＤｅｓｔｒｏｙ）が割り当てられ、各オブジェクトは、このドウデストロイ（ＤｏＤｅｓｔｒｏｙ）によりリソースを開放するようになされている。

さらに各オブジェクトにおいては、これらオブジェクトマネージャーからの要求、また他のオブジェクトからの要求に対して、適宜、リターン値によりオブザーバーによるサービス名、セレクタ番号等を通知するようになされている。

これによりこの実施の形態において、デザインロボットより通知される結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に基づいて、オブジェクトマネージャーによりオブジェクトをロード、アンロードした後、オブジェクト間通信を再構築できるようになされている。

次式は、このデザインロボットより通知される結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を示すものであり、サブジェクトのサービス名とオブザーバーのサービス名をコロンで区切ってテキストデータ形成により記述される。この場合、サブジェクト側においては、オブジェクト名であるＦｏｏＢ、データ形式Ｉｍａｇｅ、対応する構成要素名による名称ＲｉｇｈｔＣａｍｅｒａ、サブジェクトを示すＳの記述がサービス名として記述される。またオブザーバー側にあ

っては、オブジェクト名であるＦｏｏＡ、データ形式Ｉｍａｇｅ、対応する構成要素名による名称ＲｉｇｈｔＣａｍｅｒａ、オブザーバーを示すＯの記述がサービス名として記述される。

☆

FooB.Image.RightCamera.S,FooA.Image.RightCamera.O ……（５）

★

オブジェクトマネージャーにおいては、オブジェクトをロードする際に、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）の記述よりロードするオブジェクト名を検出する。さらにオブジェクトマネージャーは、スタックメモリ（Ｓｔａｃｋ)、ヒープメモリ（ｈｅａｐ）を確保してこの検出したオブジェクト名のオブジェクトを所定のファイルよりロードする。このときオブジェクトマネージャーは、オペレーションシステム（ＯＳ：Ｏｐｅｒａｔｉｏｎ Ｓｙｓｔｅｍ）よりオブジェクトＩＤを取得し、このオブジェクトＩＤを結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）に記述されたオブジェクト名と共に記録する。

これによりオブジェクトマネージャーにおいては、このようにして登録したオブジェクトＩＤを用いて以下のドウイニシャライズ（ＤｏＩｎｉｔ)、ドウコネクト（ＤｏＣｏｎｎｅｃｔ)、ドウスタート（ＤｏＳｔａｒｔ）等の処理を実行するようになされている。

すなわちドウイニシャライズ（ＤｏＩｎｉｔ）において、取得したオブジェクトＩＤと、上述したセレクタ番号に基づいて、オブジェクトマネージャーは、オ

ブジェクトのロードにより取得した全てのオブジェクトに対してドウイニシャライズ（ＤｏＩｎｉｔ）をコールする。各オブジェクトにおいては、このドウイニシャライズ（ＤｏＩｎｉｔ）により内部変数等を初期化し、これによりオブジェクトマネージャーにおいては、各オブジェクトを初期化する。

この初期化の処理において、オブジェクトマネージャーは、各オブジェクトからの通知により、それぞれ各オブジェクトのサブジェクトとしてのエントリ（Ｃｏｎｔｒｏｌ)、オブジェクトとしてのエントリ（Ｃｏｎｎｅｃｔ）を登録する。なおこの登録は、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）におけるサブジェクト及びオブジェクトの記述に対応して、オブジェクトの名前とサブジェクト、オブジェクトの名前により構成される。

これに対してドウコネクト（ＤｏＣｏｎｎｅｃｔ）において、オブジェクトマネージャーは、ドウイニシャライズ（ＤｏＩｎｉｔ）における登録に基づいて、オブザーバーを有する各オブジェクトのエントリ（Ｃｏｎｎｅｃｔ）に対して、それぞれ登録対象であるサブジェクトのサブジェクト（オブジェクトＩＤ）ＩＤ、エントリ（Ｃｏｎｔｒｏｌ）を通知する。これによりこの通知を受けたオブジェクトにおいて、通知されたサブジェクトＩＤ、エントリ（Ｃｏｎｔｒｏｌ）を用いて対応するサブジェクトを呼び出し、ここにエントリ(Ｎｏｔｉｆｙ)が接続登録される。また接続登録が要求されたサブジェクトよりレディー（Ｒｅａｄｙ）がオブザーバーに返され、これによりオブザーバーと対応するサブジェクトとが結合される。

このようにしてオブザーバー及びサブジェクト間でオブジェクト間通信を構築するにつき、オブジェクトマネージャーにおいては、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）で通知されたオブザーバーに対して対応するサブジェクトのサブジェクトＩＤ（オブジェクトＩＤ)、エントリ（Ｃｏｎｔｒｏｌ）を通知する。これによりこの実施の形態において、各オブジェクトにおいては、何ら結合対象のオブジェクトを明示的に示すことなく開発できるようになされ、また必要に応じてオブジェクトマネージャーの指示により種々のオブジェクトと結合で

きるようになされ、これにより従来に比して高い独立性を確保できるようになされている。

またこのときオブジェクトマネージャーにより結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に従ってオブジェクト間通信を構築することにより、簡易かつ自在に形態依存ソフトウエアを更新して、形態に適してソフトウエアを簡易に変更できるようになされている。

これに対してドウスタート（ＤｏＳｔａｒｔ）において、オブジェクトマネージャーは、上述したセレクタ番号１により各オブジェクトにドウスタート（ＤｏＳｔａｒｔ）を指示する。各オブジェクトにおいては、オブザーバーを有する場合、ドウコネクト（ＤｏＣｏｎｎｅｃｔ）により取得したサブジェクトＩＤ、エントリ（Ｒｅａｄｙ）を用いてサブジェクトにレディー（Ｒｅａｄｙ）を送出し、これにより対応するサブジェクトよりデータ受信可能な状態が形成され、形態依存ソフトウエアにおいては、動作を開始するようになされている。

すなわちマルチオブザーバーによるサブジェクトにおいては、登録したオブザーバーのうちのレディー（Ｒｅａｄｙ）を送出しているオブザーバーに例えばセンサー出力であるデータをノーティファイ（Ｎｏｔｉｆｙ）し、またオブザーバーにおいては、受信したデータの処理を完了して次のデータを受信可能となるとレディー（Ｒｅａｄｙ）を送出する。

これに対して上述したシャットダウン、プラグイン、プラグアウト、状態変更の場合、デザインロボットより通知される結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）が、以前デザインロボットより送出されて登録した内容と異なることにより、この場合オブジェクトマネージャーは、上述したセレクタ番号２により各オブジェクトにドウストップ（ＤｏＳｔｏｐ）を送出する。この場合オブザーバーにあっては、エントリ（Ｒｅａｄｙ）を解除する。

またドウディスコネクト（ＤｏＤｉｓＣｏｎｎｅｃｔ）において、オブジェクトマネージャーは、サブジェクトとオブザーバーとの間の通信接続を遮断する。この場合、オブジェクトマネージャーは、オブザーバーのエントリ（Ｃｏｎ－

ｎｅｃｔ）に対してドウディスコネクト（ＤｏＤｉｓｃｏｎｎｅｃｔ）のメッセージを送出することにより、オブザーバーより対応するサブジェクトにエントリ（Ｃｏｎｔｒｏｌ）の切断要求（Ｒｅｍｏｖｅ　Ｏｂｓｅｒｖｅｒ）を発行させ、これにより通信接続を遮断する。

これに対してドウデストロイ（ＤｏＤｅｓｔｒｏｙ）において、オブジェクトマネージャーは、上述したセレクタ番号３により対応するオブジェクトにドウデストロイ（ＤｏＤｅｓｔｒｏｙ）を送出し、これによりこのオブジェクトを破壊する。このときこのオブジェクトにおいては、ドウイニシャライズ（ＤｏＩｎｉｔ）で実行した登録の解除を実行する。

オブジェクトアンロード（Ｏｂｊｅｃｔ　Ｕｎｌｏａｄ）において、オブジェクトマネージャーは、ドウデストロイ（ＤｏＤｅｓｔｒｏｙ）により破壊したオブジェクトについて、スタックメモリ、ヒープメモリの領域を開放し、このオブジェクトをアンロードする。またロード時に登録したサブジェクトＩＤ、サブジェクト名を抹消する。

オブジェクトマネージャーは、これらの制御により、例えば結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）をデザインロボットより受け取ると、図１６に示すシーケンスによりドウスタート（ＤｏＳｔａｒｔ）までの制御を実行する。

すなわちオブジェクトマネージャーは、メッセージにより結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）が送られると起動し、例えばこの結合情報に記述されたオブジェクトＡ及びオブジェクトＢをロードする。なおここではオペレーションシステムのシステム命令によりオブジェクトＡ及びオブジェクトＢをロードする。このときオブジェクトマネージャーは、オブジェクトＡ及びＢのオブジェクトＩＤを取得して登録する。

続いてオブジェクトマネージャーは、上述したようにしてこの取得したオブザーバーＩＤ、セレクタ番号０によりオブジェクトＡ及びＢのドウイニシャライズ（ＤｏＩｎｉｔ）を起動し、これによりオブジェクトＡ及びＢより、サブジェク

トとしてのエントリ(Ｃｏｎｔｒｏｌ)、オブジェクトとしてのエントリ（Ｃｏｎｎｅｃｔ）を取得して登録する。

続いて登録したエントリ（Ｃｏｎｎｅｃｔ）を用いてオブザーバーであるオブジェクトＡのドウコネクト（ＤｏＣｏｎｎｅｃｔ）を起動し、サブジェクトであるオブジェクトＢとこのオブジェクトＡとを結合する。このようにして結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に基づいてオブジェクト間通信が構築され、オブジェクトマネージャーは、続いてオブジェクトＡ及びＢのドウスタート（ＤｏＳｔａｒｔ）を起動する。

図１７は、このドウスタート（ＤｏＳｔａｒｔ）以降のシーケンスを示すタイムチャートである。オブジェクトマネージャーのドウスタート（ＤｏＳｔａｒｔ）によりオブジェクトＡ、オブジェクトＢ、他のオブジェクト等が起動すると、これらのオブジェクトにおいては、上述したレディー（Ｒｅａｄｙ）とノーティファイ（Ｎｏｔｉｆｙ）とを用いてオブジェクト間通信を実行する。

すなわちこの場合、起動によりオブジェクトＡがオブジェクトＢのレディー（Ｒｅａｄｙ）エントリにレディーメッセージを送出し、オブジェクトＢによるオブジェクトＡのノーティファイ（Ｎｏｔｉｆｙ）エントリによりオブジェクトＢからオブジェクトＡにデータ等が送出される。またこのデータの処理中に、オブジェクトＡよりオブジェクトＢのレディー（Ｒｅａｄｙ）エントリに未だメッセージが送出されていない場合には、オブジェクトＢによるオブジェクトＡのノーティファイ（Ｎｏｔｉｆｙ）エントリが登録され、オブジェクトＡによるオブジェクトＢのレディー（Ｒｅａｄｙ）エントリによりデータ等が送出される。これによりこのオブジェクト間通信においては、オブジェクトＡのデータ処理能力を越えたデータの伝送が防止される。

図１８は、プラグイン、プラグアウト、状態変更におけるシーケンスを示すタイムチャートである。オブジェクトマネージャーは、デザインロボットより通知される結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）が、以前デザインロボットより送出されて登録した内容と異なっていると、全てのオブジェクトにドウスト

ップ（ＤｏＳｔｏｐ）を送出し、全てのオブジェクトの動作を停止する。この場合各オブジェクトＡ及びＢにおいては、エントリ（Ｒｅａｄｙ）を解除することにより、それ以後、自己のノーティファイエントリが起動されないようにする。

全てのオブジェクトのドウストップ（ＤｏＳｔｏｐ）が完了すると、オブジェクトマネージャーは、この場合、オブジェクトＡにドウディスコネクト（ＤｏＤｉｓＣｏｎｎｅｃｔ）を通知してこのオブジェクトＡを他のオブジェクトと切り離した後、このオブジェクトＡのドウデストロイ（ＤｏＤｅｓｔｒｏｙ）を起動する。これによりオブジェクトＡのリソースを開放し、またドウイニシャライズ（ＤｏＩｎｉｔ）で実行したオブジェクトＢの登録を解除する。

必要なオブジェクトに対するドウデストロイ（ＤｏＤｅｓｔｒｏｙ）が終了するとオブジェクトマネージャーは、アンロードをオペレーションシステムのシステム命令により実行する。なおここで必要なオブジェクトのｄｅｓｔｒｕｃｔｏｒが呼び出され、この場合はオブジェクトＡのｄｅｓｔｒｕｃｔｏｒが呼び出されることにより、ロード時に実行したオブジェクトＡの登録を解除する。さらにオブジェクトマネージャーは、スタックメモリ、ヒープメモリを開放し、これによりオブジェクトＡのアンロードを完了する。

続いてオブジェクトマネージャーは、結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）に従ってオブジェクトＣのロードを指示し、図１６について上述したと同様にして、順次ドウイニシャライズ（ＤｏＩｎｉｔ）、ドウコネクト（ＤｏＣｏｎｎｅｃｔ）、ドウスタート（ＤｏＳｔａｒｔ）を起動する。これによりこの場合、動作中に、コンパイルすることなくオブジェクトＡ及びＢによる構成がオブジェクトＢ及びＣによる構成に変更されるようになされている。

（１－５）連想記憶によるオブジェクトの登録

図１９は、（５）式により示した結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）に従ってオブジェクトＡ及びオブジェクトＢを示す略線図である。オブジェクトマネージャーにおいては、結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）を連想記憶に保持して記録すると共に、オブジェクトのドウイニシャライズ（ＤｏＩ

ｎｉｔ）を起動して各オブジェクトを登録する際に、結合情報（Ｃｏｎｎｅｃ－ｔｉｏｎ　Ｄａｔａ）の記述を基準にして連想記憶に各オブジェクトを登録する。

すなわちこの場合、オブザーバー側であるオブジェクトＡのオブジェクト名がＦｏｏＡ、サブジェクト側であるオブジェクトＢのオブジェクト名がＦｏｏＢ、データ形式がＩｍａｇｅであり、サブの名称がＲｉｇｈｔＣａｍｅｒａである。これによりこのサブジェクトとオブジェクトとの結合情報（Ｃｏｎｎｅｃ－ｔｉｏｎ　Ｄａｔａ）においては、サブジェクトのサービス名とオブザーバーのサービス名をコロンで区切って（ＦｏｏＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｓ，ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）により記述されることになる。

ブジェクトマネージャーは、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）がデザインロボットより渡されると、図２０に示すように、この結合情報のオブザーバー側の記述（Ｏ．ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ）をキーとして、サブジェクト側の記述（ＦｏｏＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｓ）を連想記憶１に記憶して蓄積する。

これに対して各オブジェクトにおいては、ドウイニシャライズ（ＤｏＩｎｉｔ）によりオブジェクトマネージャーの登録エントリに対してサービス名、セレクタ番号を通知し、オブジェクトマネージャーは、これらの通知を同様に連想記憶２に記憶して各オブジェクトを登録する。

すなわちこの場合オブザーバー側であるオブジェクトＡにおいては、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）におけるオブザーバー側の記述（ＦｏｏＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）と同一の記述によるサービス名を通知し、またコネクションエントリのセレクタ番号を通知する。なおこのオブジェクトＡにおいては、このセレクタ番号を５とする。

オブジェクトマネージャーにおいては、図２１に示すように、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）におけるオブザーバー側の記述（ＦｏｏＡ．

Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）をキーにして、このオブジェクトＡのオブジェクトＩＤ、セレクタ番号（ＯＩＤ，Ｅｎｔｒｙ）＝（２，５）を連想記憶２に記録する。なおここでこのオブジェクトＡのオブジェクトＩＤは、オペレーションシステムより値１として通知されたものである。

またこの場合、サブジェクト側であるオブジェクトＢにおいては、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）におけるサブジェクト側の記述（ＦｏｏＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｓ）と同一の記述によるサービス名を通知し、またコネクションエントリのセレクタ番号を通知する。なおこのオブジェクトＢにおいては、このセレクタ番号を５とする。

オブジェクトマネージャーにおいては、図２１において、オブジェクトＡについての記述と共に示すように、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）におけるサブジェクト側の記述（ＦｏｏｔＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｓ）をキーにして、このオブジェクトＢのオブジェクトＩＤ、セレクタ番号（ＯＩＤ，Ｅｎｔｒｙ）＝（１，５）を連想記憶２に記録する。なおここでこのオブジェクトＢのオブジェクトＩＤは、オペレーションシステムより値２として通知されたものである。

オブジェクトマネージャーにおいては、ドウコネクト（ＤｏＣｏｎｎｅｃｔ）時、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）よりサブジェクト側の記述とオブジェクト側の記述を分離する。これによりオブジェクトマネージャーにおいては、連想記憶１に記憶されたキー（ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）と、このキー（ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）により連想記憶１に蓄積された種々の記述（ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）の１つとにそれぞれ対応するように、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）の記述を分離する。

さらにオブジェクトマネージャーは、この分離した記述によりそれぞれ連想記憶２を検索し、対応するオブザーバーのオブジェクトＩＤ及びセレクタ番号（１，５）、サブジェクトのオブジェクトＩＤ及びセレクタ番号（２，５）を検出す

る。

オブジェクトマネージャーは、これによりオブジェクトＩＤ及びセレクタ番号（１，５）に対して、オブジェクトＩＤ及びセレクタ番号（２，５）のメッセージを送出し、オブザーバーのコネクトエントリを起動し、このコネクトエントリーをオブジェクトＩＤ及びセレクタ番号（２，５）によるサブジェクトの情報を通知する。またこのコネクトエントリーよりオブジェクトＩＤ及びセレクタ番号（２，５）によるサブジェクト側コントロールエントリを起動し、このコントロールエントリにオブザーバー側であるオブジェクトＩＤ及びセレクタ番号（１，６）（この場合はノーティファイ（Ｎｏｔｉｆｙ）である）を登録する。このときオブザーバー側は、戻り値によりレディーのセレクタ番号を得、これを登録する。

これによりオブジェクトマネージャーにおいては、連想記憶１及び２を用いてオブジェクト間の結合を確立するようになされている。

このようにしてオブジェクト間の結合が確立すると、オブジェクトマネージャーは、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）のオブザーバー側の記述（ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）をキーにして、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）のサブジェクト側の記述（ＦｏｏＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｓ）を連想記憶３に記憶し、さらに連想記憶の記録よりオブザーバー側の記述（ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ）をキーにしたサブジェクト側の記述（ＦｏｏＢ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｓ）を削除する。

これによりオブジェクトマネージャーは、結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に対応する記述により、オブジェクト間の結合が確立されてなるオブザーバー及びサブジェクトを連想記憶３に登録して保持するようになされている。

かくするにつきオブジェクトマネージャーは、ドウディスコネクト（ＤｏＤｉｓＣｏｎｎｅｃｔ）時においては、上述したドウコネクト（ＤｏＣｏｎｎｅｃｔ

）時とは逆に、結合を遮断すると共に、連想記憶３に記録された対応する記述を削除して連想記憶１に記録し直し、また連想記憶２の対応する記述を削除すようになされている。

これに対して形態依存ソフトウエアの変更により例えば次式により示される結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）がデザインロボットより通知されたとする。

☆

FooB.Image.RightCamera.S,FooC.Image.RightCamera.O　　……（６）

★

すなわちこの結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）は、オブジェクトＢに代えて、オブジェクトＣにオブザーバーを変更する記述である。オブジェクトマネージャーは、この場合、連想記憶２より　ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏをキーにしてそれまでオブザーバーに設定されているオブジェクトＩＤ、セレクタ番号（１，５）を検出する。さらにこのオブジェクトＩＤ、セレクタ番号（１，５）によりオブザーバーであるオブザーバーＡにドウディスコネクト（ＤｏＤｉｓＣｏｎｎｅｃｔ）を指示すると共に、連想記憶２の対応する記述（オブザーバー及びサブジェクトであるオブジェクトＡ及びＢの記述）を削除し、また連想記憶３に記録された対応する記述を削除して連想記憶１に記録し直す。

すなわちオブジェクトマネージャーは、連想記憶２よりＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．Ｏ、ＦｏｏＡ．Ｉｍａｇｅ．ＲｉｇｈｔＣａｍｅｒａ．

ＳをキーにしてなるオブジェクトｉＤ、セレクタ番号の記述を削除する。これによりオブジェクトマネージャーは、オブジェクトを動的に変更できるようになされている。

（２）第１の実施の形態の動作

以上の構成において、ロボット１においては（図１）、本体２に移動ユニット３として足を取り付けると４足型の形態となり、この足に取り付けたモータの駆動により４足歩行するようになされている。また後足に代えてタイヤを取り付けるとタイヤ型の形態となり、このタイヤにより移動可能とされる。

ロボット１においては（図２）、ロボットコンポーネント２４、２５、２６等である足等により配置されたセンサーの出力、さらには頭部２Ａに配置された各種情報取得手段により取得された情報等に基づいて、中央処理ユニット２２の処理により必要な制御指令が出力されて、これら移動ユニット３等が制御される。

ロボット１においては（図３及び図４）、オブジェクト指向により構成される上位レイヤーのソフトウエアにより、デバイスドライバレイヤーの各種デバイスドライバを駆動してこれらの制御が実行される。この制御において、デバイスドライバレイヤーの上位層であるロボットシステムソフトウエアにおいては、バーチャルロボットによりシリアルバスに接続された各部品が検出され、ロボット１全体でどの様なロボットコンポーネントがどの様な順番で接続されているのかを示す結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）がデザインロボットに通知される。これによりロボット１においては、シルアルバスに接続されたデバイスにより現在の形態が特定され、さらにこの現在の形態を構成する制御対象等が具体的に特定される。

ロボット１においては、結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）に応じてこのロボットシステムソフトウエアの上位層のソフトウエアが更新され、これにより形態に適した適切なソフトウエアにより全体の動作が制御される。

このソフトウエアの更新において、ロボット１は、ロボットシステムソフトウ

エアのデザインロボットにより、各形態毎に、各部品の構成要素情報（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ Ｌｏｃａｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）群に対してラベルを付加して記述されてなるデザインファイルと、バーチャルロボットより通知される結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）とを比較して、形態に適した適切なオブジェクトを選択し、このオブジェクトによる形態依存ソフトウエアが変更される（図７、図８）。これによりロボット１においては、形態に適した適切なソフトウエアに簡易に変更して全体の動作を制御することができる。

また、上位層のソフトウエアがロボットの形態に依存しない形態独立ソフトウエアと、形態に依存する形態依存ソフトウエアに分離されて、さらに形態独立ソフトウエアにより形態に依存しない形式により取り扱われるデータが形態依存ソフトウエアにより形態に対応した形式により変換されてロボットシステムソフトウエアとの間で送受されるように構成された状態で、この形態依存ソフトウエアのみが更新される。これにより簡易な処理によりソフトウエアを更新することが可能となる。

さらにこのときバーチャルロボットにおいては、テキストデータ形式の記述により構成要素情報（ＣＰＣ Ｐｒｉｍｉｔｉｖｅ Ｌｏｃａｔｉｏｎ Ｉｎ－ｆｏｒｍａｔｉｏｎ）を生成するのに対し（図８及び図９）、デザインファイルにおいては、同様のテキストデータ形式により記述されていることにより、変化する形態が将来増大したような場合にも、簡易に対応することが可能となる。

ロボット１においては、このソフトウエアの更新がオブジェクトマネージャーにより形態依存ソフトウエアのオブジェクトの再構築により実行される。すなわちオブジェクトマネージャーにおいては、起動時等において、デザインロボットより現在の形態に適したオブジェクトの結合を指示する結合情報（Ｃｏｎ－ｎｅｃｔｉｏｎ Ｄａｔａ）が通知され、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）により指定されるオブジェクトがロードされる（図１６）。

さらにロードしたオブジェクトが初期化された後、結合情報（Ｃｏｎｎｅｃ－

ｔｉｏｎ　Ｄａｔａ）の指定によりオブジェクトマネージャーでデータ交換可能にサブジェクト及びオブザーバーが結合される。これによりロボット１においては、形態に適したオブジェクトにより形態依存ソフトウエアが構築される。

従ってロボット１においては、具体的な結合対象を何ら考慮することなく各オブジェクトを設計して、ソフトウエアを構築することができ、また形態に応じて結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を生成することにより、形態に適したソフトウエアを簡易に構築することができる。さらに再コンパイル、再リンキングすることなくソフトウエアを変更することができる。これによりロボット１においては、オブジェクトの独立性を従来に比して格段的に向上することができ、その分自由な形態の変更に対して適切なソフトウエアを簡易かつ迅速に提供することが可能となる。またこの結合情報の設定により１つのオブジェクトと複数のオブジェクトとの間でオブジェクト間通信を構築することもできる。

この結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）がサブジェクトと通信するデータの形式とを特定する情報であることにより、オブジェクトマネージャーにおいては、この結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）により対応するオブジェクトＩＤ、セレクタ番号を特定して簡易にオブジェクト通信を構築することができる。

またデザインロボットにより、複数の結合情報が記録されてなるデザインファイルから結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を選択してオブジェクトマネージャーに提供することにより、このデザインの設定により必要に応じて各種形態に対応するソフトウエアを簡易に構築することが可能となる。

またこの結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）においても、テキストデータ形式により記述されていることにより、簡易に種々に変更することができる。

これに対して形態が変更されると、ロボット１においては、バーチャルロボットにより形態の変更が検出され、この変更された形態に係る結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）がデザインロボットに通知さ

れる。さらにこのデザインロボットにおいてデザインファイルとの比較により、変更された形態に係る結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）が生成される。またこの結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に従ってオブジェクトマネージャーによって形態の変更により不要となったオブジェクトがアンロードされると共に、形態の変更により必要なオブジェクトがロードされ、形態依存ソフトウエアを構成するオブジェクトが再構築される。さらにこれらのオブジェクトが結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）によりオブジェクト間通信可能に結合され、これにより形態依存ソフトウエアが更新される。

これによりロボット１においては、形態独立ソフトウエア等を動作させたままの状態で、必要に応じて形態依存ソフトウエアを更新して形態に適したソフトウエアを構築することができる。またこのソフトウエアの変更を電力消費モードの切り換えに適用することも可能となる。

（３）第１の実施の形態の効果

以上の構成によれば、部品検出結果である結合情報と、各形態毎の部品検出結果に対応する情報の集合によるデザインファイルとを比較し、この比較結果に基づいて形態に応じてソフトウエアを変更することにより、部品の変更に伴う形態の変更に対応してソフトウエアを変更することができ、これにより現在の形態に対応する適切なソフトウエアを簡易に設定することができる。

このときオブジェクトの変更によりソフトウエアを変更することにより、簡易な処理によりソフトウエアを変更することができる。

また上位のソフトウエアを形態依存ソフトウエアと形態独立ソフトウエアにより構成し、形態依存のソフトウエアの変更により形態に応じてソフトウエアを変更することにより、形態に適したソフトウエアを簡易に提供することができる。

また部品検出結果に対応する情報がテキストデータによる記述であることにより、変化する形態が将来増大したような場合にも、簡易に対応することが可能となる。

また以上の構成によれば、部品を交換して形態を変更できるようになされたロ

ボットにおいて、結合情報に基づいて、オブジェクトマネージャーによりデータ交換可能にオブジェクトを結合することにより、従来に比してオブジェクトの独立性を向上することができ、その分簡易に形態に適してソフトウエアを変更することができる。また再コンパイル、再リンキングすることなく、簡易にソフトウエアを変更することができる。またこの結合情報の設定により１つのオブジェクトと複数のオブジェクトとの間でオブジェクト間通信を構築することもできる。

このとき結合情報が、オブジェクトと、通信するデータの形式とを特定する情報であることにより、簡易にこれらオブジェクトを結合することができる。

また複数の結合情報が記録されてなるデザインファイルより結合情報を選択してオブジェクトを結合することにより、このデザインファイルの設定により形態に適したソフトウエアを簡易に提供することができる。

さらに結合情報に従って、対応するオブジェクト間の結合を一旦切り離した後、オブジェクトを結合してオブジェクト間通信を構築することにより、ソフトウエアを動作させたままの状態で、必要に応じてソフトウエアを更新して形態に適したソフトウエアを構成することができる。

またすなわちバーチャルロボットにより部品を検出して結合情報を提供することにより、形態に応じてソフトウエアを変更することができる。

また結合情報をテキストデータにより記述したことにより、形態に応じた適切なソフトウエアを簡易かつ迅速に作成することができる。

またこのようにしてデータ通信するオブジェクト間において、レディー、ノーティファイによりデータ交換することにより、処理に能力を越えるようなデータの供給を防止することができる。

（４）第２の実施の形態

ところで上述したドウイニシャライズ（ＤｏＩｎｉｔ）等の命令をＣａｌｌ命令により処理するオペレーションシステムにおいては、命令の送出により送出先に処理を渡すことになり、オブジェクトマネージャーにおいては、送出先よりＲｅｐｌｙが返されるまで各オブジェクトから送出されるメッセージをブロックす

ることになる。従って図１６について上述したように、ドウイニシャライズ（ＤｏＩｎｉｔ）を送出した後、サブジェクトとしてのエントリ(Ｃｏｎｔｒｏｌ)、オブジェクトとしてのエントリ（Ｃｏｎｎｅｃｔ）をオブジェクトマネージャーで登録することが困難になる。

このためこの実施の形態に係るロボットおいては、図４との対比により図２２に示すように、これらエントリ(Ｃｏｎｔｒｏｌ、Ｃｏｎｎｅｃｔ)をサービスマネージャーに登録し、サービスマネージャーによりオブジェクト間通信を管理する。これに対してオブジェクトマネージャーにおいては、オブジェクトのロード、アンロード等の処理のみ実行する。すなわちこの実施の形態において、オブジェクトマネージャーは、形態依存ソフトウエアを構成する全てのオブジェクトについて、動作の停止、オブジェクト間通信のための結合解除、オブジェクトの破壊、リソースの開放を指示し、さらに新たに必要なオブジェクトのロード、初期化、オブジェクトの起動を指示する。これに対してサービスマネージャーは、オブジェクト間通信における通信相手の情報を管理する。

なおこの実施の形態においては、このオブジェクトマネージャー、サービスマネージャーの構成、これらの構成に関連する各部の構成が異なる点を除いて、上述した第１の実施に形態と同一であることにより、この実施の形態では、オブジェクトマネージャー、サービスマネージャーの構成を中心に説明し、重複した説明は省略する。

なお図２３に示すように、この実施の形態において、オブジェクト間の結合は、オブジェクト間でデータを送受信するための結合を意味し、通信を行う双方のサービス間の結合として表現する。また結合は、（５）式について上述したように、データ出力側のサービス、データ入力側のサービス、通信するデータ型、データフロー方向で表現される。また図２４に示すように、結合状態とは、ロボットシステム内にロードされているオブジェクトの一群においてオブジェクト同士に張られている結合の集合を意味し、接続情報とは、起動時、形態の変更時に新たにオブジェクト間に張られる結合の情報を意味し、現在の結合状態と新しい結

合状態の差分情報である。また切断情報とは、形態の変更時に、不要となった結合情報を意味し、新しい結合状態と現在の結合状態の差分情報である。

図２５は、図１０との対比により、この実施の形態に係るロボットシステムソフトウエアの起動時の処理を示すタイムチャートである。電源が投入されると、又はリセットがされると、始めにロボットシステムソフトウエアを構成する各オブジェクトに対して初期化等の処理が実行される。すなわちロボットにおいては、図２６に示すように、パワーマネージャー、サービスマネージャー、バーチャルロボット、デザインロボットの順に、オブジェクトマネージャーよりドウイニシャライズ（ＤｏＩｎｉｔ）の送出、リプライ（Ｒｅｐｌｙ）の受信が繰り返され、これによりロボットシステムソフトウエアにおけるドウイニシャライズのフェーズが実行される。なお以下においては、適宜図面上で処理の順番を数字により示す。

さらに図２７に示すように、パワーマネージャー、サービスマネージャー、バーチャルロボット、デザインロボットの順に、オブジェクトマネージャーより順次ドウスタート（ＤｏＳｔａｒｔ）が送出され、これによりロボットシステムソフトウエアにおけるドウスタートのフェーズが実行されてこれらパワーマネージャー、サービスマネージャー等が動作を開始する。

このときデザインロボットにおいては、このドウスタート（ＤｏＳｔａｒｔ）により、バーチャルロボットに対してロボットの現在の形態を示す情報の送出を指示し、バーチャルロボットは、この指示によりロボットコンポーネントがどの様な順番で接続されているのかを示す結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）をデザインロボットに通知する。デザインロボットは、バーチャルロボットよりこの結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）を受け取り、この結合情報（ＣＰＣ Ｃｏｎｎｅｃｔｉｏｎ Ｉｎｆｏｒｍａｔｉｏｎ）より結合情報（Ｃｏｎｎｅｃｔｉｏｎ Ｄａｔａ）を生成した後、オブジェクトマネージャーにリプライ（Ｒｅｐｌｙ）を返す。

このようにしてドウスタートのフェーズが実行されると、ロボットにおいては、続いてサービスマネージャーに対して、オブジェクトマネージャーよりディスコネクト（ＤｉｓＣｏｎｎｅｃｔ）の指示が発行される。起動時においては、何らオブジェクトがロードされていないことにより、この場合、この指示に従ってサービスマネージャーは、デザインロボットより結合情報（Ｃｏｎｎｅｃ－ｔｉｏｎ　Ｄａｔａ）を取得し、オブジェクトマネージャーに処理の完了を通知する。オブジェクトマネージャーは、続いて同様にしてデザインロボットより結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を取得する。これによりサービスマネージャーにおいては、形態依存ソフトウエア等を構成する各オブジェクトについて、オブジェクト間通信の構築に必要な情報を取得するのに対し、オブジェクトマネージャーにおいては、これらのオブジェクトのロード、アンロードに必要な情報を取得する。

これによりオブジェクトマネージャーは、現在の形態に対応する必要なオブジェクトと、不必要なオブジェクトとのリストを作成する。さらに図２８に示すように、これらの必要なオブジェクトＡ～Ｃについて、順次ロードを指示し、これらオブジェクトＡ～Ｃを構築する。このときオブジェクトマネージャーは、図２９に示すように、このロードコマンドの実行により、各オブジェクトのオブジェクトＩＤ（ＯＩＤ－ａ、ＯＩＤ－ｂ、ＯＩＤ－ｃ）を取得する。

続いて図３０及び図３１に示すように、オブジェクトマネージャーは、このようにして取得したオブジェクトＩＤにより各オブジェクトに順次ドウイニシャライズ（ＤｏＩｎｉｔ）をコールする。各オブジェクトにおいては、このドウイニシャライズ（ＤｏＩｎｉｔ）によりサービスマネージャーに対して各サービスの登録を要求する。なおここでこのサービスの登録は、各サービス名、オブジェクトＩＤ、セレクタの登録により実行される。

サービスマネージャーは、各要求に応動して各オブジェクトのサービスを登録すると、登録の完了を各オブジェクトに通知し、各オブジェクトにおいては、この通知によりオブジェクトマネージャーにリプライ(Ｒｅｐｌｙ）を返す。

このようにして各オブジェクトの初期化が完了すると、オブジェクトマネージャーは、各オブジェクトのＤｉｓＣｏｎｎｅｃｔをサービスマネージャーに指示する。ここで図３２に示すように、サービスマネージャーは、上述したドウスタート（ＤｏＳｔａｒｔ）フェーズでデザインロボットが取得した結合情報を取得することによりリストを作成して保持する。ここでこのリストは、Ｃｏｎ－ｎｅｃｔに係るリスト、ＤｉｓＣｏｎｎｅｃｔに係るリスト等により構成され、各リストは、それぞれ第１の実施の形態についた上述した記述による結合情報により、各オブジェクトのサービス名、データフォームで構成される。またサービスマネージャーは、各オブジェクトのドウイニシャライズ（ＤｏＩｎｉｔ）で取得したサービスをサービスリストにより保持する。ここでこのサービスリストは、各オブジェクト毎にサービス情報としてまとめられて形成され、各サービスにあっては、各サービスに係るオブジェクトＩＤ、セレクタによるサービスエントリと、各サービス名とにより構成される。

サービスマネージャーは、図３３に示すように、オブジェクトマネージャーより各オブジェクトの結合（Ｃｏｎｎｅｃｔ）が指示されると、これら各オブジェクトより取得したサービスの情報と、デザインロボットから取得した結合情報とを照合する。さらに照合結果に基づいて、順次各オブジェクトに通信対象を登録し、これにより各オブジェクトを順次データ通信可能に結合する。

すなわち図３４に示すように、サービスマネージャーは、各オブジェクトからサービスが登録されると、各サービスのサービス名、サービスエントリよるサービス情報を各オブジェクト毎にまとめてサービス情報（Ｓｅｒｖｉｃｅ Ｂｉｎｄｉｎｇ）を作成し、このサービス情報（Ｓｅｒｖｉｃｅ Ｂｉｎｄ－ｉｎｇ）を登録する。これによりサービスマネージャーは、サービスリストを作成する。またデザインロボットから取得した結合情報と、過去に記録した結合情報とから新しい結合状態と現在の結合状態とを把握し、接続情報、切断情報を作成する。さらにこれら接続情報、切断情報からＣｏｎｎｅｃｔに係るリスト、ＤｉｓＣｏｎｎｅｃｔに係るリストを作成する。

サービスマネージャーは、サービスリストとＣｏｎｎｅｃｔに係るリストとの照合により、結合するサービスのエントリーを検出し、この検出結果に従って対応するオブジェクトにサービスのエントリーを通知する。またサービスリストとＤｉｓＣｏｎｎｅｃｔに係るリストとの照合により、結合を遮断するサービスのエントリーを検出し、この検出結果に従って対応するオブジェクトにサービスのエントリーを通知する。なお図３０においては、オブジェクトが未だ結合されていない起動時の処理であることにより、サービスマネージャーにおいては、切断情報を作成することなく一連の処理を実行する。これにより図３５に示すように、サービスマネージャーは、オブザーバーとサブジェクトとの間で、データ交換可能に対応するメソドを接続する。なお図３６は図１９との対比によりこのようにして結合されたオブジェクトＡ及びオブジェクトＢを示す略線図である。

サービスマネージャーは、このようにして結合情報による接続が完了すると、オブジェクトマネージャーに接続の完了を通知する（図３０）。オブジェクトマネージャーにおいては、この通知が得られると、図３８に示すように、各オブジェクトにドウスタート（ＤｏＳｔａｒｔ）を指示する。これにより図２８との対比により図３８に示すように、各オブジェクトＡ～Ｃにおいては、結合が完了してデータ通信可能に動作を開始し、ロボットにおいては、各オブジェクト間におけるデータ通信により形態に応じて種々の動作を実行できるようになされている。

図３９は、図１８との対比に係るプラグイン、プラグアウトによる形態変更時のシーケンスを示すタイムチャートである。ロボットにおいては、図４０に示すように、バーチャルロボットにより形態の変更が通知されると、この一連のシーケンスを実行する。すなわちオブジェクトマネージャーにおいては、この通知により図４１に示すように、ロボットシステムソフトウエアの上位に位置する各オブジェクト（図３９においてはオブジェクトＡ～Ｃ）に順次ドウストップ（ＤｏＳｔｏｐ）を送出し、これらオブジェクトの動作を停止する。さらに続いて起動時とは逆の順序で、デザインロボット、バーチャルロボット、サービスマネージ

ャー、パワーマネージャーに順次ドウストップ（ＤｏＳｔｏｐ）を送出し、これらオブジェクトの動作を制限する。なおここでこれらロボットシステムソフトウエアを構成する各オブジェクトにおいては、ドウストップ（ＤｏＳｔｏｐ）により上位のオブジェクトに関連する処理のみを停止し、例えば電源の管理等のロボットシステムソフトウエアを構成する各オブジェクト間の処理においては引続き処理を継続するようになされている。

このようにして全てのオブジェクトについてドウストップ（ＤｏＳｔｏｐ）の処理を完了すると、オブジェクトマネージャーは、ロボットシステムソフトウエアを構成するパワーマネージャー、サービスマネージャー、バーチャルロボット、デザインロボットに対して、起動時と同様の順序により順次ドウスタート（ＤｏＳｔａｒｔ）を送出し、これによりパワーマネージャー、サービスマネージャー等の動作を指示する。

このときデザインロボットにおいては、図２７について上述したと同様に、バーチャルロボットより結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）を取得し、この結合情報（ＣＰＣ　Ｃｏｎｎｅｃｔｉｏｎ　Ｉｎｆｏｒｍａｔｉｏｎ）より結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を生成する。なおオブジェクトマネージャーにおいては、ユーザーによる形態の変更が完全に完了するのを待って結合情報を取得可能とするために、例えばドウストップ（ＤｏＳｔｏｐ）の後、所定時間経過して（ＤｏＳｔａｒｔ）を指示することが考えられる。

このようにしてドウスタート（ＤｏＳｔａｒｔ）を送出してパワーマネージャー等から応答が得られると、オブジェクトマネージャーは、サービスマネージャーに対して、ディスコネクト（ＤｉｓＣｏｎｎｅｃｔ）を指示する。ここでこのように形態が変更された場合には、現在ロードされているオブジェクトと、形態の変更に対応するオブジェクトが異なることにより、サービスマネージャーは、図４３においてＣｈｅｃｋ　Ｃｏｎｎｅｃｔｉｏｎにより示すように、デザインロボットより変更された形態に係る結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ

）を取得する。さらにサービスマネージャーは、この取得した結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）より図３４について上述したようにして切断情報を作成し、この切断情報により対応するオブジェクトの結合を解除する。

このようにして切断を完了すると、サービスマネージャーは、オブジェクトマネージャーに処理の完了を通知し、オブジェクトマネージャーは、続いて図４４に示すように、同様にしてデザインロボットより結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）を取得する。さらにオブジェクトマネージャーは、このようにして取得した結合情報（Ｃｏｎｎｅｃｔｉｏｎ　Ｄａｔａ）に基づいて、対応するオブジェクトにドウデストロイ（ＤｏＤｅｓｔｒｏｙ）をコールし、対応するオブジェクトにより専有されたメモリを開放する。

さらにオブジェクトマネージャーは、続いてこのオブジェクトのアンロードを指示する。これら一連の処理によりサービスマネージャーは、形態の変更により必要とされなくなったオブジェクトについて、オブジェクト間通信できないようにオブジェクト間の結合を遮断するのに対し、オブジェクトマネージャーは、このようにして必要とされなくなったオブジェクトをアンロードする。

このようにして必要とされなくなったオブジェクトをアンロードすると、オブジェクトマネージャーは、図４５に示すように、新たに必要となったオブジェクトＤのロードを指示した後、このロードしたオブジェクトにドウイニシャライズ（ＤｏＩｎｉｔ）を指示する。続いてオブジェクトマネージャーは、サービスマネージャーにオブジェクト間の結合を指示し、この場合オブジェクトマネージャーは、図３４について説明した接続情報に基づいて対応するオブジェクト間で、起動時と同様にデータ通信が可能になるように結合情報を通知する。オブジェクトマネージャーは、このようにしてデータ通信可能に各オブジェクトが結合されると、各オブジェクトにドウスタート（ＤｏＳｔａｒｔ）を送出して一連の処理を完了する。これら一連の処理により図３７との対比によりオブジェクトＡ～Ｄの関係を図４６に示すように、オブジェクトＣに代えてオブジェクトＤが登録され、図４７に示すように、これらオブジェクトＡ、Ｂ、Ｄは、動作を開始するこ

とになる。

図４８は、このようにして起動時、形態変更時に係るサービスマネージャーの一連の処理を示す状態遷移図である。すなわちサービスマネージャーは、初期化された後、待機し、オブジェクトマネージャーからの指示により切断情報、接続情報を生成し、起動時においては、このうちの接続情報により各オブジェクトを結合する。また形態変更時にあっては、切断情報により対応するオブジェクトの結合を遮断した後、接続情報により各オブジェクトを結合する。

このようにしてデータ通信可能に結合された各オブジェクトは、サブジェクト側のメモリ空間に存在するローカルデータをオブザーバー側のメモリ空間にコピーすることにより、又はサブジェクトとオブザーバーとの共有メモリを用いてデータ通信する。

図４９は、このうちのコピーによるデータ通信を示す略線図である。サブジェクトにおいては、メモリ空間に存在するローカルデータをオブザーバー側のメモリ空間にコピーし、オブザーバー側のｉｎｖｏｋｅされるメソッドにコピー先の先頭アドレスを通知する。これによりロボットにおいては、データ量の少ない場合、配信する相手先が少ない場合に、簡易にデータ通信できるようになされている。

これに対して図５０は、共有メモリを用いたデータ通信を示す略線図である。サブジェクトにおいては、メモリ空間にマッピングされている共有メモリにデータを書き込み、この共有メモリのアドレスをメッセージとしてオブザーバーに送出する。オブザーバーにおいては、この通知されたアドレスにより共有メモリ領域を自信のメモリ空間にマッピングして使用する。これによりロボットにおいては、画像データ、音声データ等の大容量のデータを送信する場合、配信先の数が多い場合に適用して、簡易かつ確実にデータ通信できるようになされている。

このようにして共有メモリによりデータ通信する際に、オブジェクトにおいては、データを配信したオブジェクトの数を計測できるようになされている。すなわち図５１に示すように、データを書き換え可能なオブジェクトにおいては、共

有メモリにデータを書き込むと対応するカウンタのカウント値を値１にセットする。なおこのデータを書き換え可能なオブジェクトは、データを配信したサブジェクトをメンバーに持つオブジェクトである。

サブジェクトは、Ｎｏｔｉｆｙを実行した数だけカウンタをインクリメントし、オブザーバは、使用の済んだ領域のＩＤをレディーの際に返却する。サブジェクトは、レディーを受けると、対応する共有メモリのカウンタを値１だけデクリメントする。

これによりこのデータ書き換え可能なオブジェクトにおいては、このカウント値により他のオブジェクトによる共有メモリのアクセスが完了したか否かを判断できるようになされ、この判断結果により共有メモリを書き換えて確実にデータを保護するようになされている。

このようにしてコピー又は共有メモリによりデータ通信する際に、ロボットにおいては、図５２に示すように、所定のコマンドによりデータ通信に供するデータを順次オブザーバーのメモリ空間にコピーした後、又は共有メモリに書き込んだ後、所定のコマンドによりこれら複数のデータをオブザーバーで取得可能に、アドレスを通知する。これによりロボットでは、複数のデータをまとめて配信できるようになされ、その分解使い勝手を向上できるようになされている。

第２の実施の形態によれば、ドウイニシャライズ（ＤｏＩｎｉｔ）等の命令をＣａｌｌ命令により処理するオペレーションシステムにおいても、現在の形態に対応する適切なソフトウエアを簡易に設定することができ、また従来に比してオブジェクトの独立性を向上することができる。

（５）他の実施の形態

なお上述の実施の形態においては、ドウイニシャライズ等のセレクタ番号については、予め設定しておく場合について述べたが、本発明はこれに限らず、結合情報により各オブジェクトのドウイニシャライズ等を起動することによりセレクタ番号を種々に設定してもよく、さらにはオブジェクトマネージャーへのオブジェクトの登録に従ってドウスタート等を起動することによりセレクタ番号を種々

に設定するようにしてもよい。

また上述の実施の形態においては、形態依存ソフトウエアについてのみ結合情報によりオブジェクトを構築する場合について述べたが、本発明はこれに限らず、全てのソフトウエアを結合情報により構築する場合、さらにはデバイスドライバレイヤーのみを結合情報により構築する場合等に広く適用することができる。

また上述の実施の形態では、オブザーバーとサブジェクトによるオブザーバーパターンを適用したロボットに本発明を適用した場合について述べたが、本発明はこれに限らず、サーバークライアントシステム等にも広く適用することができる。因みにこの場合、例えばクライアントが結合情報をデータ入力として受け取ってサーバーと結合することにより、サーバーの名前をソースコード中に記述しないでシステムを構築することが可能となる。かくするにつき、この場合は、クライアント、サーバーがそれぞれオブジェクトを構成することになる。

また上述の実施の形態においては、レディー、ノーティファイによりデータ交換する場合について述べたが、本発明はこれに限らず、種々の手法によりデータ交換するオブジェクト間通信に広く適用することができる。

上述のように本発明によれば、形態毎に保持された部品検出結果に対応する情報と、現在の形態に係る部品検出結果との比較結果に基づいて、形態に応じてソフトウエアを変更することにより、現在の形態に対応する適切なソフトウエアを簡易に設定することができる。また結合情報に基づいて、データ交換可能にオブジェクトを結合することにより、従来に比してオブジェクトの独立性を向上することができる。

産業上の利用の可能性

本発明は、ペット型のロボットや、サーバクライアントシステムに利用することができる。

請 求 の 範 囲

1．部品を交換して形態を変更できるようになされたロボットであって、

取り付けられた前記部品を検出して部品検出結果を出力する部品検出手段と、

前記部品を交換して得られる形態毎に、前記部品検出結果に対応する情報を保持する情報保持手段と、

前記部品検出結果と前記情報保持手段に保持した情報との比較結果に基づいて、形態に応じてソフトウエアを変更するソフトウエア変更手段と、

前記ソフトウエア変更手段により設定されたソフトウエアに従って、全体の動作を制御する制御手段と

を備えることを特徴とするロボット。

2．前記ソフトウエア変更手段は、

前記比較結果に基づいて、前記ソフトウエアを構成するオブジェクトを変更することにより、前記形態に応じてソフトウエアを変更する

ことを特徴とする請求の範囲第1項に記載のロボット。

3．前記ソフトウエアは、

前記形態の変更に依存する形態依存のソフトウエアと、

前記形態の変更に依存しない形態独立のソフトウエアとを有し、

前記ソフトウエア変更手段は、

前記形態依存のソフトウエアの変更により、前記形態に応じてソフトウエアを変更する

ことを特徴とする請求の範囲第1項に記載のロボット。

4．前記部品検出結果に対応する情報が、

テキストデータによる記述である

ことを特徴とする請求の範囲第１項に記載のロボット。

５．オブジェクト間でデータ通信する演算処理装置において、
データ交換可能に前記オブジェクトを結合するオブジェクトマネージャーと、
前記オブジェクトマネージャーに、前記オブジェクトの結合に必要な結合情報を提供する結合情報提供手段とを備える
ことを特徴とする演算処理装置。

６．前記結合情報が、
結合するオブジェクトと、通信するデータの形式とを特定する情報である
ことを特徴とする請求の範囲第５項に記載の演算処理装置。

７．前記結合情報提供手段は、
複数の結合情報が記録されてなるファイルより、前記結合情報を選択して前記オブジェクトマネージャーに提供する
ことを特徴とする請求の範囲第５項に記載の演算処理装置。

８．前記オブジェクトマネージャーは、
前記結合情報に従って、対応するオブジェクト間の結合を切り離した後、前記オブジェクトを結合することにより、前記オブジェクトにより構成されるソフトウエアを変更する
ことを特徴とする請求の範囲第５項に記載の演算処理装置。

９．前記ソフトウエアの変更が、電力消費モードの切り換えである
ことを特徴とする請求の範囲第８項に記載の演算処理装置。

１０．前記結合情報が、

テキストデータによる記述である

ことを特徴とする請求の範囲第５項に記載の演算処理装置。

１１．オブジェクト間でデータ通信するオブジェクト間通信方法において、

所定の情報提供手段より提供される結合情報に基づいて、データ交換可能に前記オブジェクトを結合する

ことを特徴とするオブジェクト間通信方法。

１２．前記結合情報が、

結合するオブジェクトと、通信するデータの形式とを特定する情報である

ことを特徴とする請求の範囲第１１項に記載のオブジェクト間通信方法。

１３．前記結合情報が、

複数の結合情報が記録されてなるファイルより選択されて提供される

ことを特徴とする請求の範囲第１１項に記載のオブジェクト間通信方法。

１４．前記結合情報に従って、対応するオブジェクト間の結合を切り離した後、前記オブジェクトを結合することにより、前記オブジェクトにより構成されるソフトウエアを変更する

ことを特徴とする請求の範囲第１１項に記載のオブジェクト間通信方法。

１５．前記ソフトウエアの変更が、電力消費モードの切り換えである

ことを特徴とする請求の範囲第１４項に記載のオブジェクト間通信方法。

１６．前記結合情報が、

テキストデータによる記述である

ことを特徴とする請求の範囲第１１項に記載のオブジェクト間通信方法。

１７．部品を交換して形態を変更できるようになされたロボットであって、
オブジェクト間でデータ通信することによりオブジェクトにより構成されるソフトウエアにより全体の動作が制御され、
データ交換可能に前記オブジェクトを結合するオブジェクトマネージャーと、
前記オブジェクトマネージャーに、前記オブジェクトの結合に必要な結合情報を提供する結合情報提供手段とを備える
ことを特徴とするロボット。

１８．前記結合情報が、
結合するオブジェクトと、通信するデータの形式とを特定する情報である
ことを特徴とする請求の範囲第１７項に記載のロボット。

１９．前記結合情報が、
複数の結合情報が記録されてなるファイルより選択されて提供される
ことを特徴とする請求の範囲第１７項に記載のロボット。

２０．前記結合情報に従って、対応するオブジェクト間の結合を切り離した後、前記オブジェクトを結合することにより、前記ソフトウエアを変更する
ことを特徴とする請求の範囲第１７項に記載のロボット。

２１．前記ソフトウエアの変更が、電力消費モードの切り換えである
ことを特徴とする請求の範囲第２０項に記載のロボット。

２２．前記部品を検出する検出手段を有し、
前記結合情報提供手段により、
前記検出手段の検出結果に応じて前記結合情報が提供されることにより、前記ソフトウエアの変更が前記形態に対応してなる

ことを特徴とする請求の範囲第２０項に記載のロボット。

２３．前記結合情報が、

テキストデータによる記述である

ことを特徴とする請求の範囲第１７項に記載のロボット。

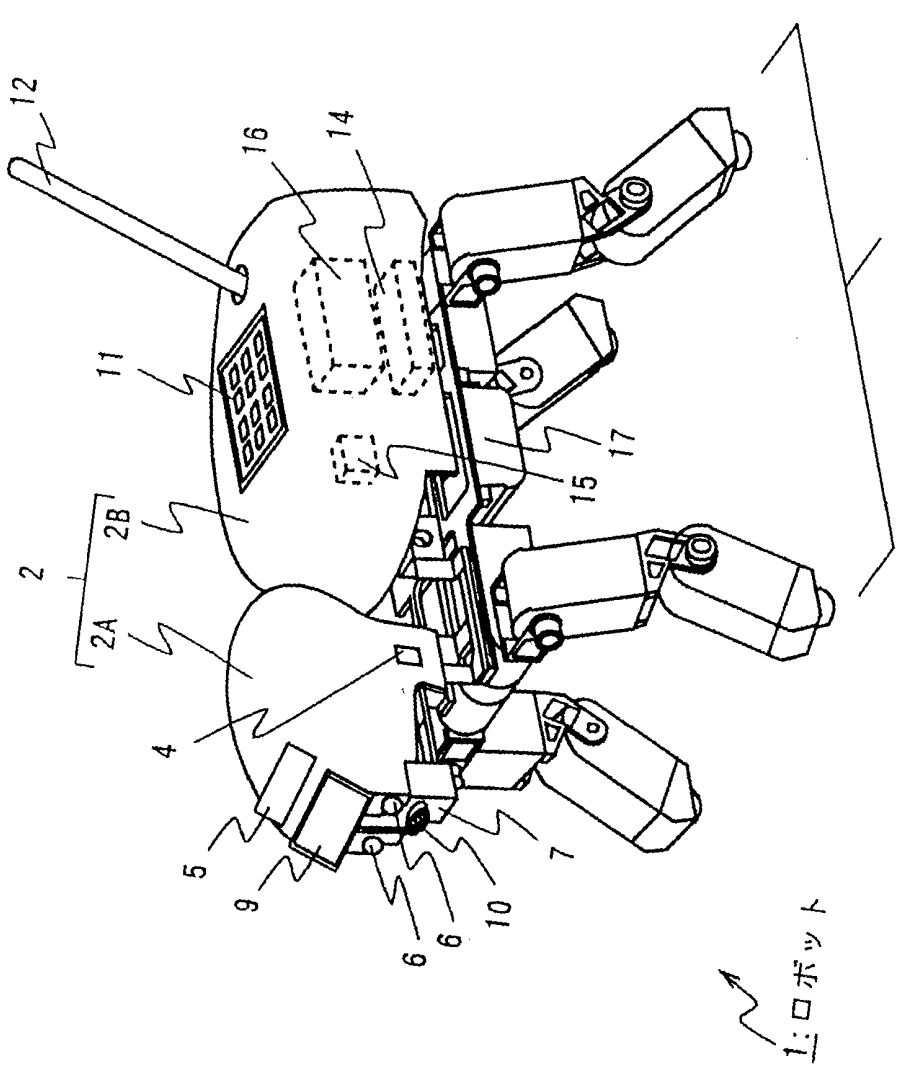


図1

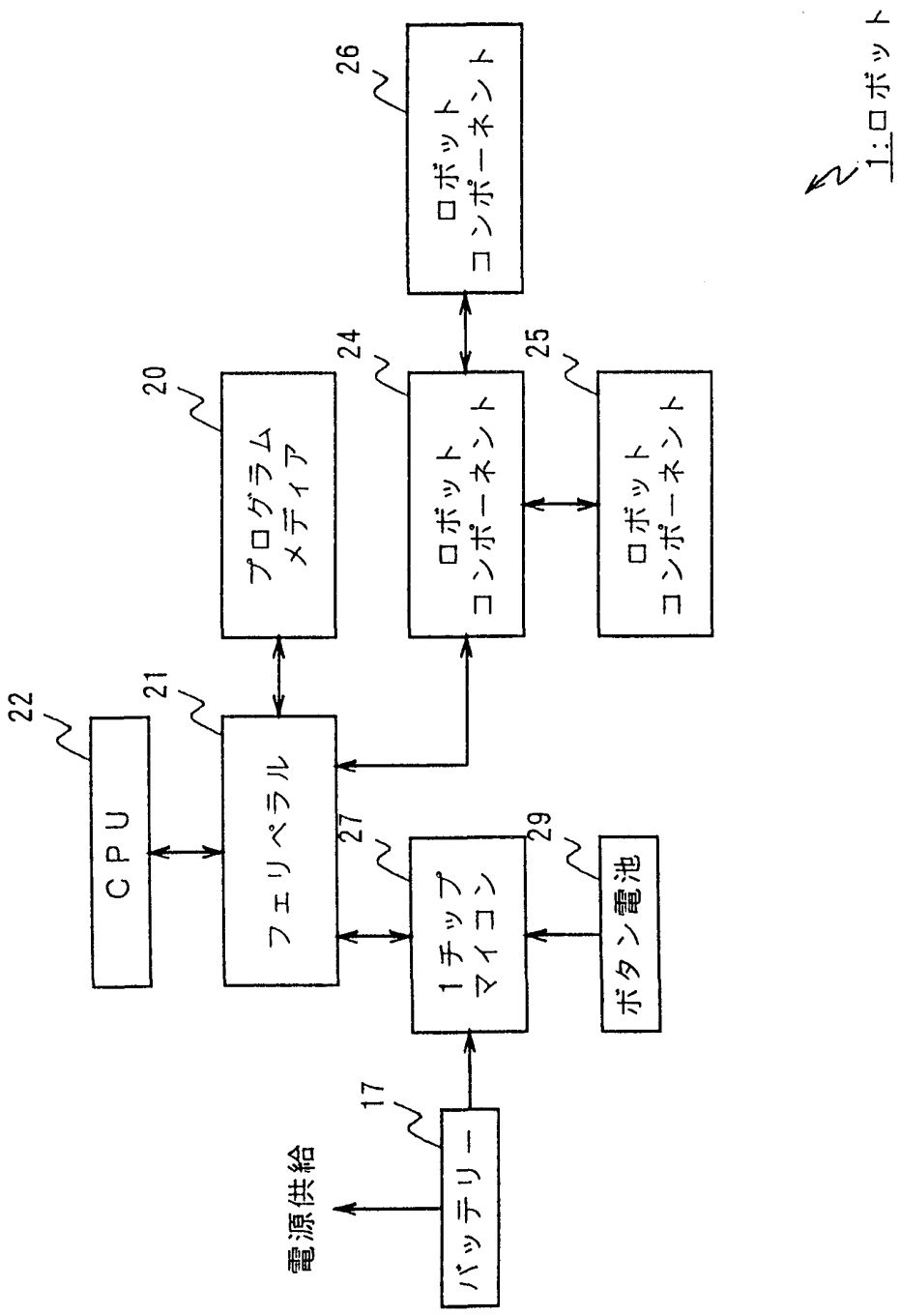

1:ロボット
26
ロボット
コンポーネント
24
25
20
プログラム
メディア
ロボット
コンポーネント
ロボット
コンポーネント
22
21
CPU
フェリペラル
27
29
1チップ
マイコン
ボタン電池
17
バッテリー
電源供給

図2

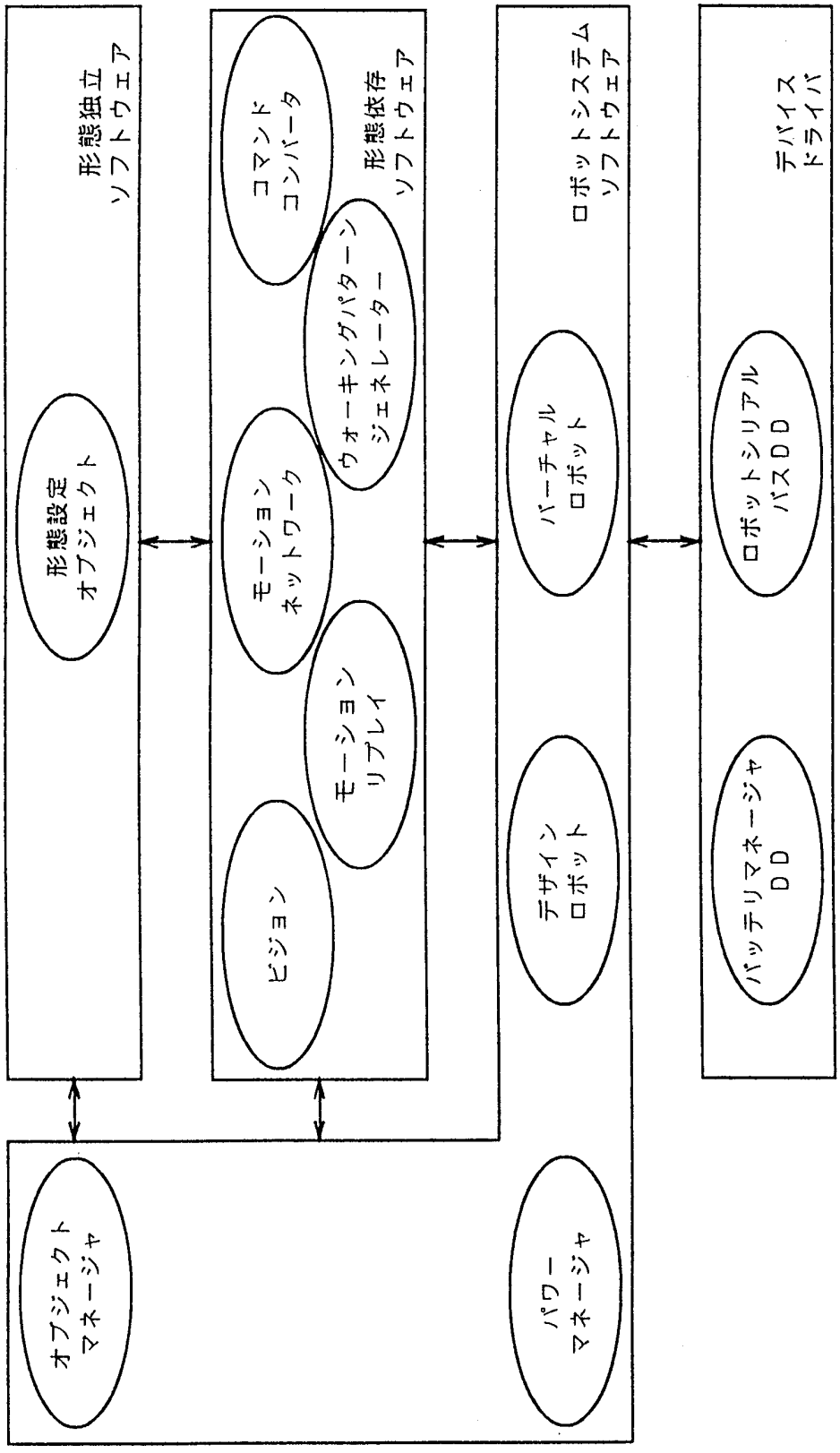

形態独立
ソフトウェア
形態設定
オブジェクト
形態依存
ソフトウェア
コマンド
コンバータ
ウォーキングパターン
ジェネレーター
モーション
ネットワーク
モーション
リプレイ
ビジョン
ロボットシステム
ソフトウェア
バーチャル
ロボット
デザイン
ロボット
パワー
マネージャ
デバイス
ドライバ
ロボットシリアル
バスDD
バッテリマネージャ
DD
オブジェクト
マネージャ

図3

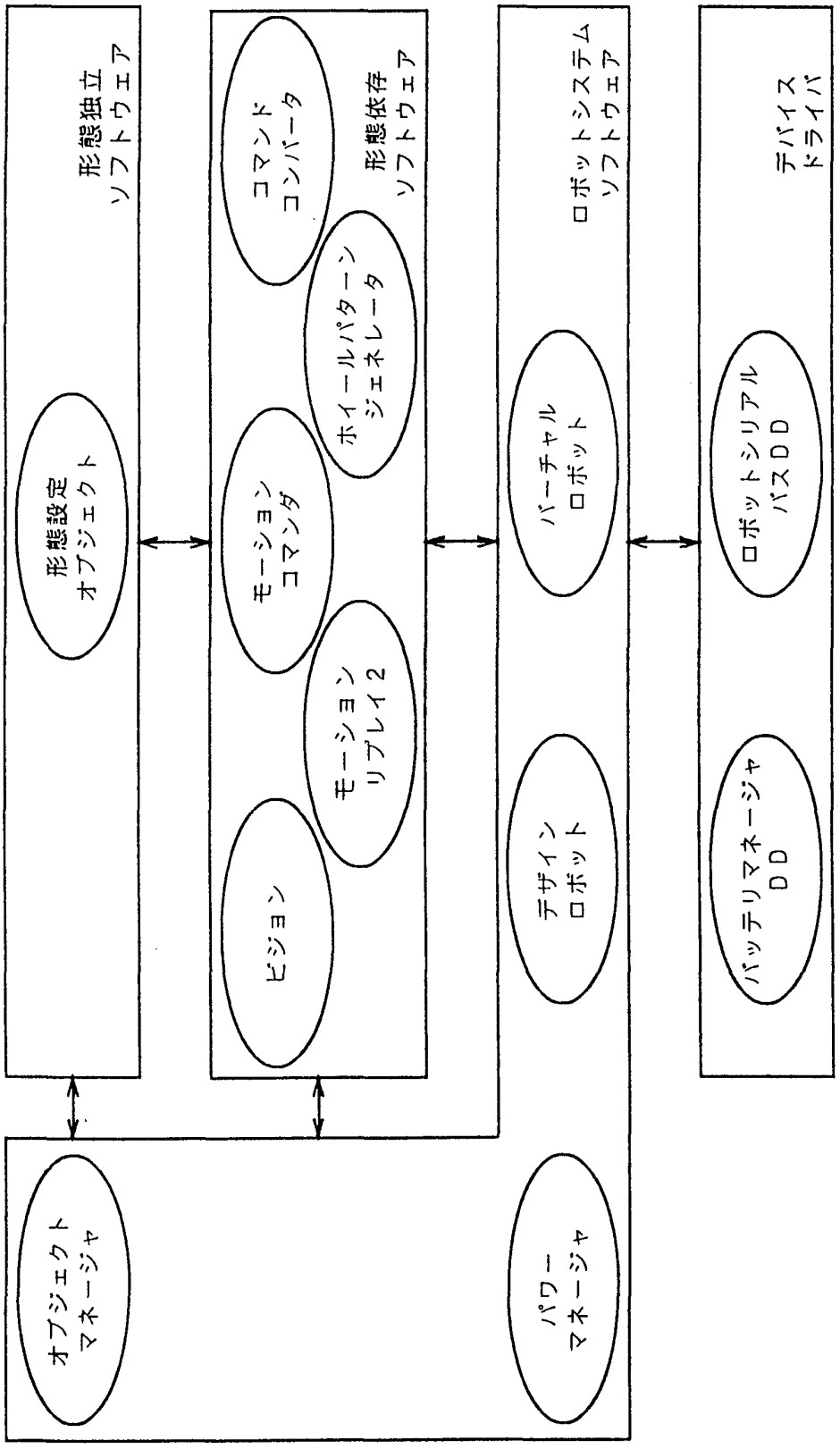

形態独立
ソフトウェア
形態依存
ソフトウェア
ロボットシステム
ソフトウェア
デバイス
ドライバ
形態設定
オブジェクト
コマンド
コンバータ
ホイールパターン
ジェネレータ
モーション
コマンダ
モーション
リプレイ2
ビジョン
バーチャル
ロボット
デザイン
ロボット
ロボットシリアル
バスDD
バッテリマネージャ
DD
オブジェクト
マネージャ
パワー
マネージャ

図4

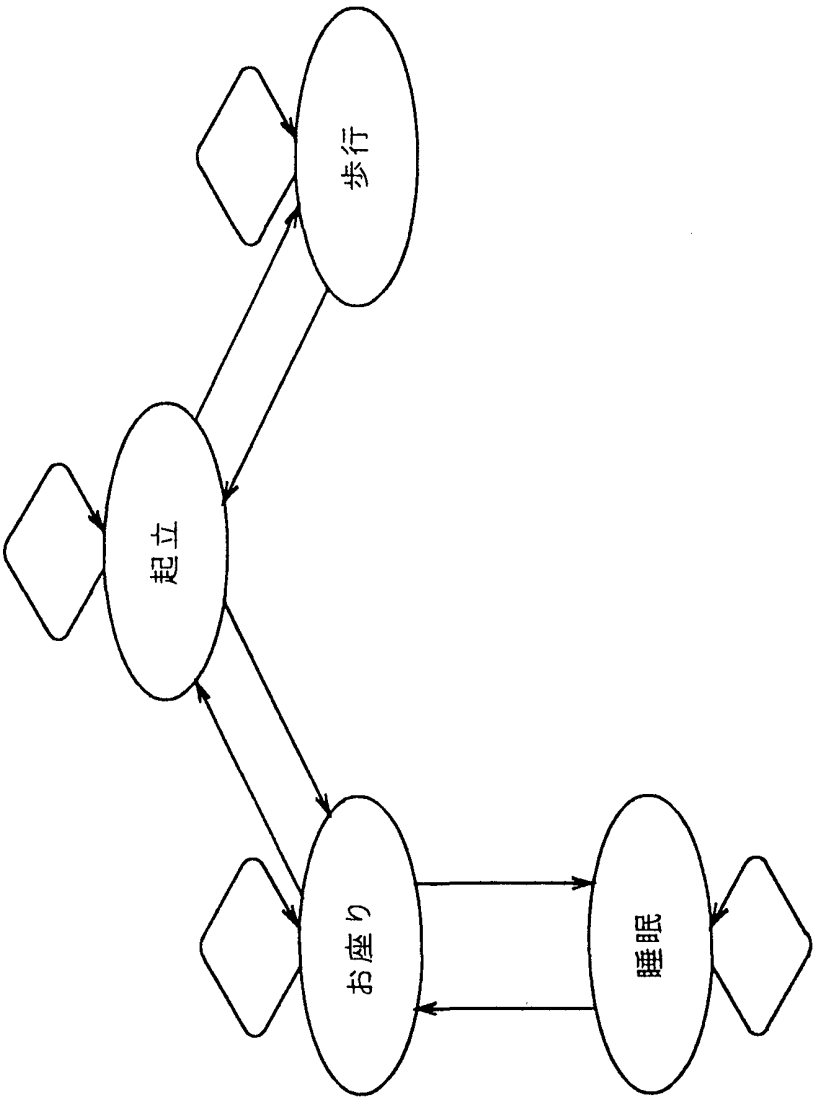


図5

状態 / 入力	Ｓｌｅｅｐ	Ｒｅｓｔ	Ｗａｋｅｕｐ	Ｍｏｖｅ
１分何も検出されない	Ｓｌｅｅｐ	Ｓｌｅｅｐ	Ｒｅｓｔ	Ｗａｋｅｕｐ
赤、大きさ＜１０が１０秒以上続いた	Ｒｅｓｔ	Ｗａｋｅｕｐ	Ｍｏｖｅ	Ｍｏｖｅ
赤、大きさ＞＝１０	Ｗａｋｅｕｐ	Ｍｏｖｅ	Ｍｏｖｅ	Ｍｏｖｅ

図６

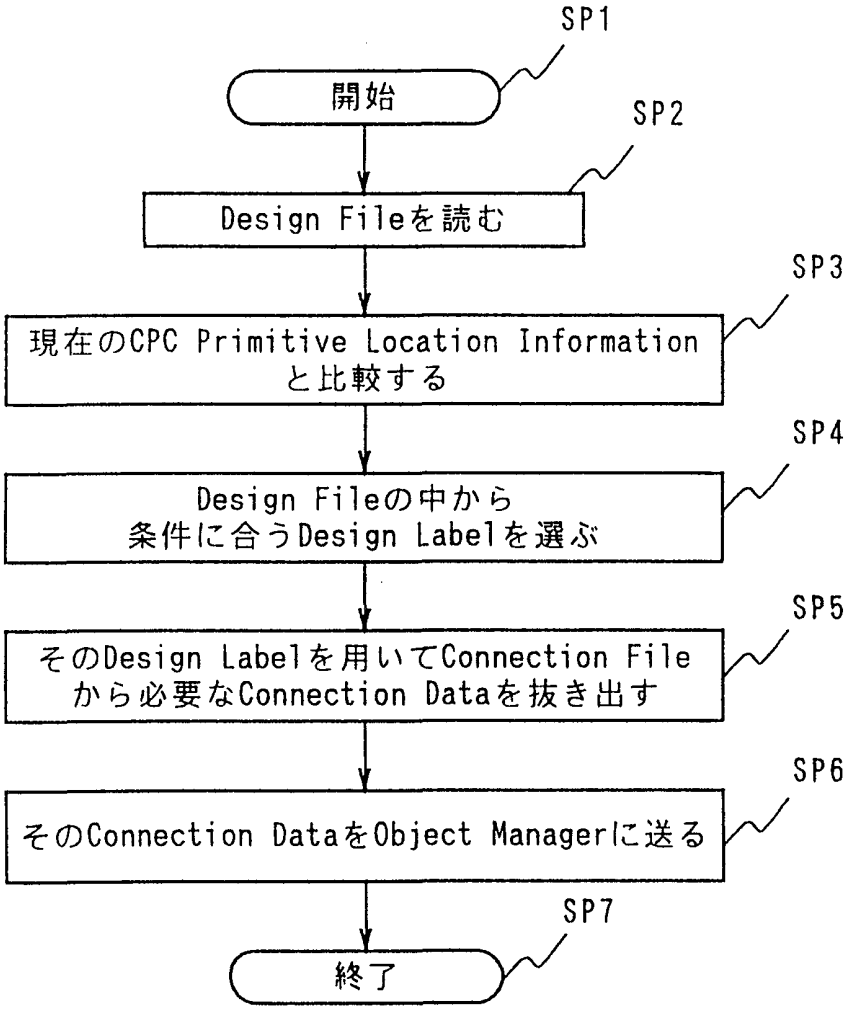

SP1
開始
SP2
Design Fileを読む
SP3
現在のCPC Primitive Location Information
と比較する
SP4
Design Fileの中から
条件に合うDesign Labelを選ぶ
SP5
そのDesign Labelを用いてConnection File
から必要なConnection Dataを抜き出す
SP6
そのConnection DataをObject Managerに送る
SP7
終了

図7

```
DRX-4Legged{
    DRX Head;
    DRX RightFrontLeg;
    DRX LeftFrontLeg;
    DRX RightRearLeg;
    DRX LeftRearLeg;
}
DRX Wheel{
    DRX Head;
    DRX RightFrontLeg;
    DRX LeftFrontLeg;
    DRX RightRearWheel;
    DRX LeftRearWheel;
}
DRX Head{
    DRX CameraLink;
}
Virtual CameraLink{
    DRX CameraLink;
    Camera;
    Coordinate:/
}
DRX RightFrontLeg{
    Coordinate:/C1/C0-Joint;
    Coordinate:/C1/C0/C0-Joint;
    Coordinate:/C1/C0/C0/C0-Joint;
    Coordinate:/C1/C0/C0/C0/C0-TouchSensor;
}
DRX LeftFrontLeg{
    Coordinate:/C2/C0-Joint;
    Coordinate:/C2/C0/C0-Joint;
    Coordinate:/C2/C0/C0/C0-Joint;
    Coordinate:/C2/C0/C0/C0/C0-TouchSensor;
}
DRX RightRearLeg{
    Coordinate:/C3/C0-Joint;
    Coordinate:/C3/C0/C0-Joint;
    Coordinate:/C3/C0/C0/C0-Joint;
    Coordinate:/C3/C0/C0/C0/C0-TouchSensor;
}
DRX LeftRearLeg{
    Coordinate:/C4/C0-Joint;
    Coordinate;/C4/C0/C0-Joint;
    Coordinate:/C4/C0/C0/C0-Joint;
    Coordinate:/C4/C0/C0/C0/C0-TouchSensor;
}
DRX RightRearWheel{
    Coordinate:/C3/C0-wheel;
}
DRX LeftRearWheel{
    Coordinate:/C4/C0-wheel;
}
```

図8

```
CPC Current Information{
    Coordinate:/C0/C0-Joint;
    Coordinate:/C0/C0/C0-Joint;
    Coordinate:/C0/C0/C0/C0-Joint;
    Coordinate:/C0/C0/C/C0/C0-Camera;
    Coordinate:/C1/C0-Joint;
    Coordinate:/C1/C0/C0-Joint;
    Coordinate:/C1/C0/C0/C0-Joint;
    Coordinate:/C1/C0/C0/C0/C0-TouchSensor;
    Coordinate:/C2/C0-Joint;
    Coordinate:/C2/C0/C0-Joint;
    Coordinate:/C2/C0/C0/C0-Joint;
    Coordinate:/C2/C0/C0/C0/C0-TouchSensor;
    Coordinate:/C3/C0-Joint;
    Coordinate:/C3/C0/C0-Joint;
    Coordinate:/C3/C0/C0/C0-Joint;
    Coordinate:/C3/C0/C0/C0/C0-TouchSensor;
    Coordinate:/C4/C0-Joint;
    Coordinate:/C4/C0/C0-Joint;
    Coordinate:/C4/C0/C0/C0-Joint;
    Coordinate:/C4/C0/C0/C0/C0-TouchSensor;
}
```

図9

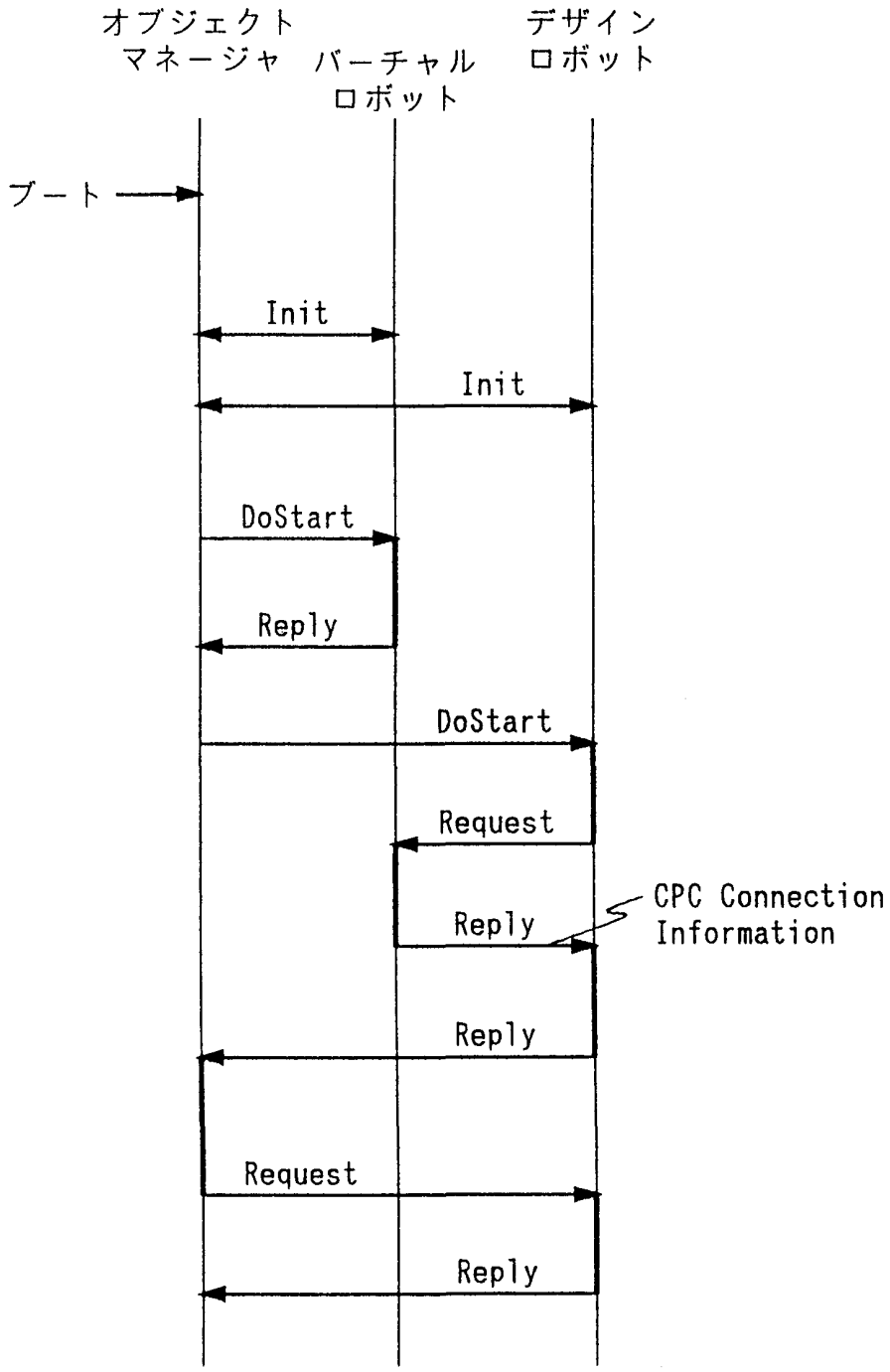

オブジェクト
マネージャ
バーチャル
ロボット
デザイン
ロボット
ブート
Init
Init
DoStart
Reply
DoStart
Request
Reply
CPC Connection
Information
Reply
Request
Reply

図１０

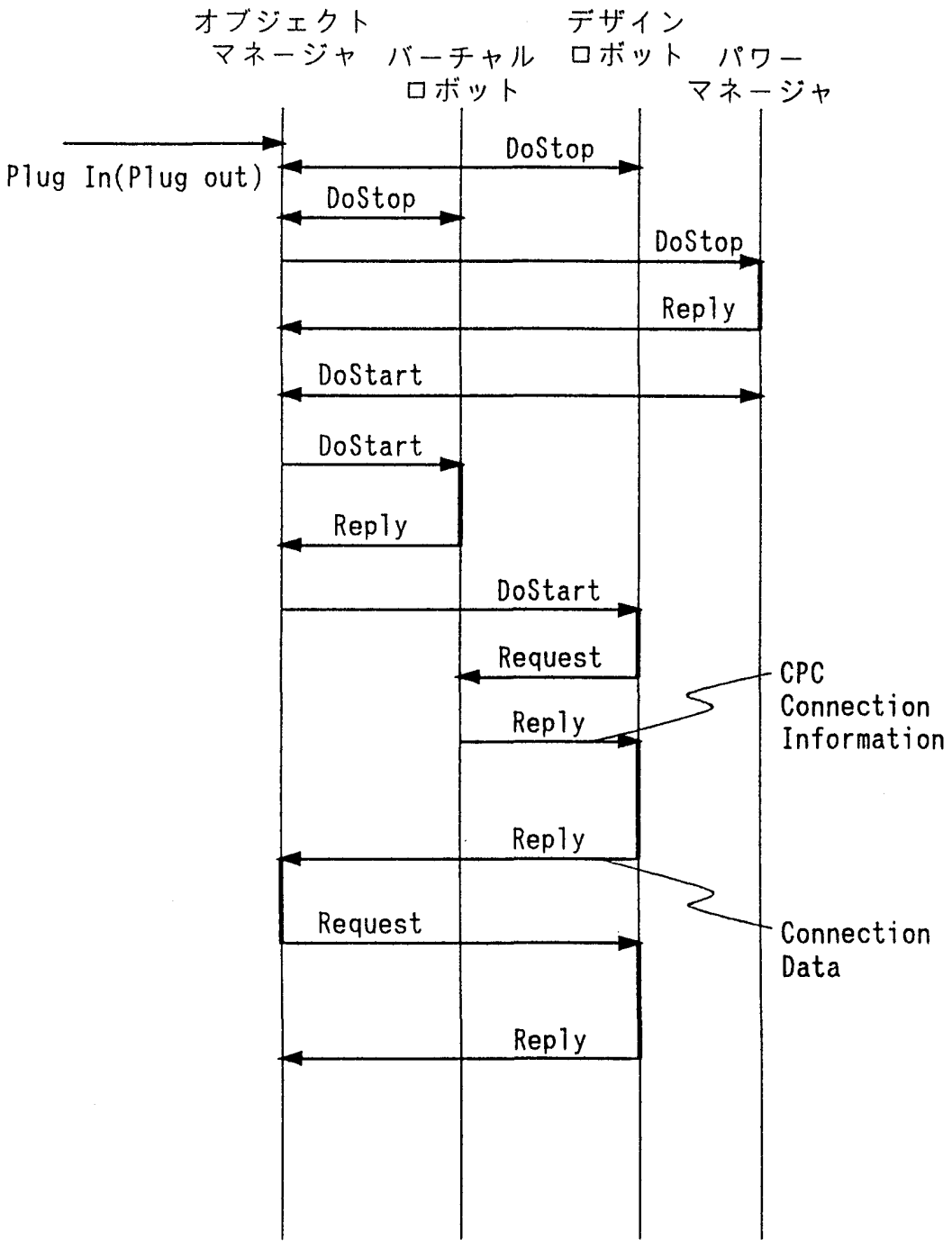

オブジェクト
マネージャ
バーチャル
ロボット
デザイン
ロボット
パワー
マネージャ
Plug In(Plug out)
DoStop
DoStop
DoStop
Reply
DoStart
DoStart
Reply
DoStart
Request
Reply
CPC
Connection
Information
Reply
Request
Connection
Data
Reply

図１１

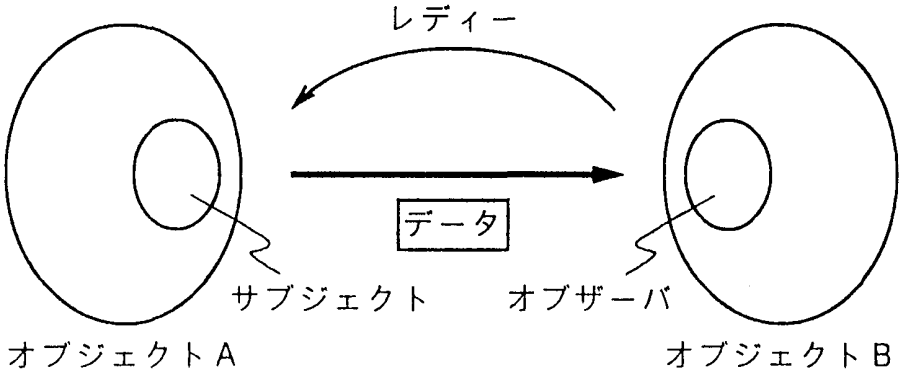

レディー
データ
サブジェクト
オブザーバ
オブジェクトA
オブジェクトB

図１２

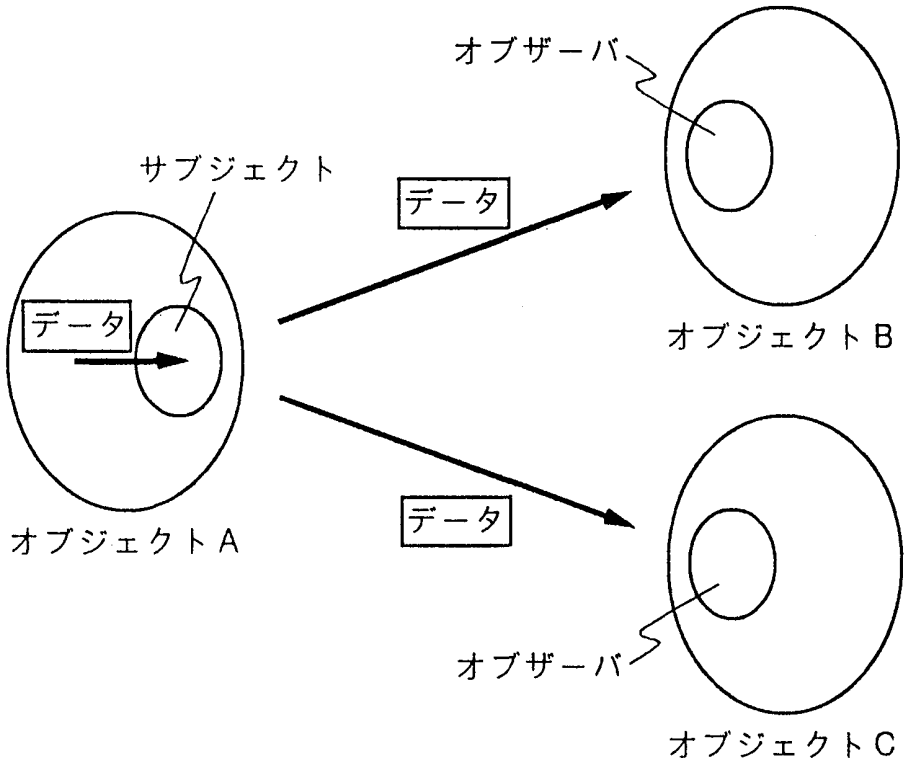

オブザーバ
サブジェクト
データ
データ
オブジェクトB
データ
オブジェクトA
オブザーバ
オブジェクトC

図１３

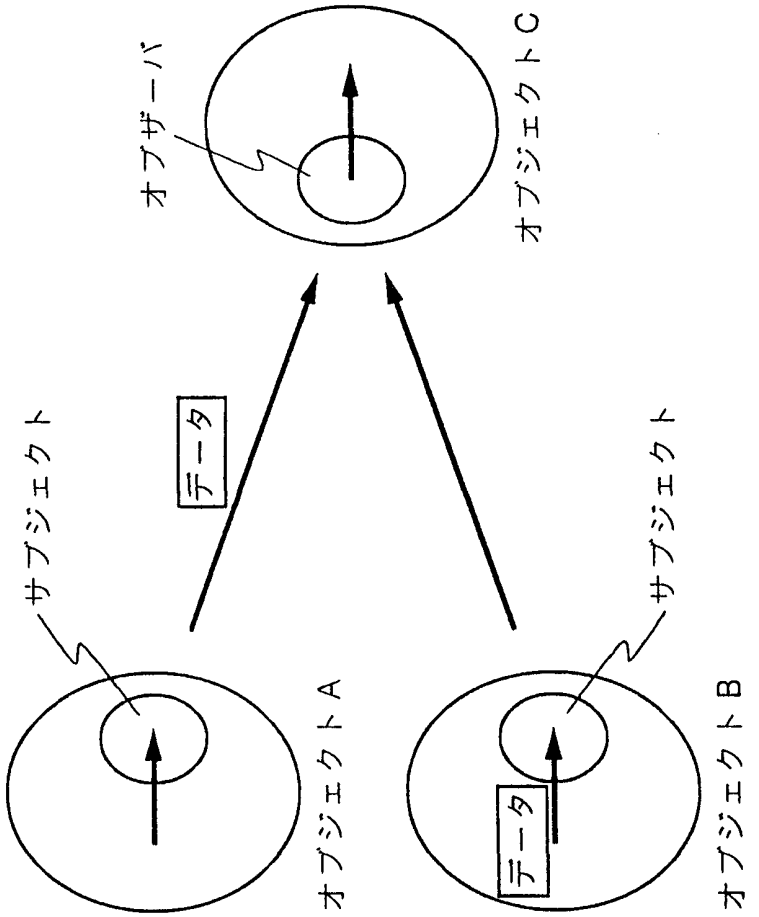


図１４

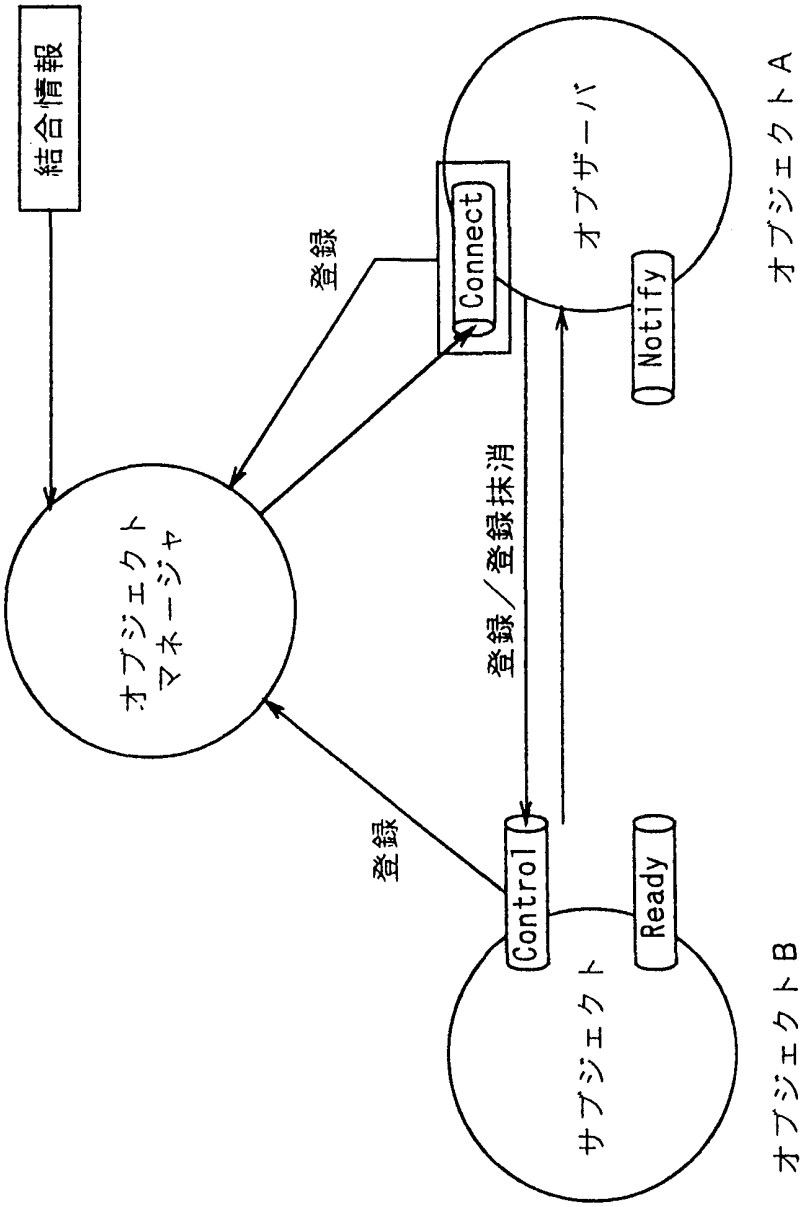

結合情報
登録
Connect
オブザーバ
Notify
オブジェクトA
オブジェクトマネージャ
登録／登録抹消
登録
Control
Ready
サブジェクト
オブジェクトB

図１５

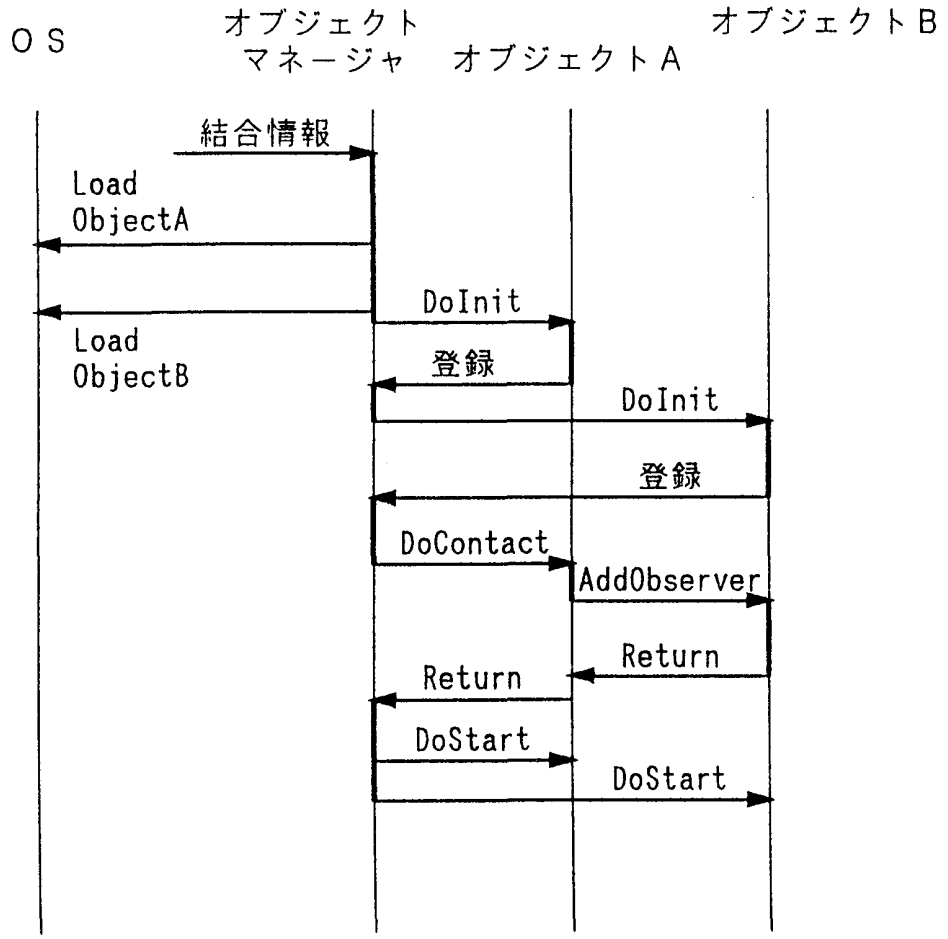

ＯＳ
オブジェクト
マネージャ
オブジェクトＡ
オブジェクトＢ
結合情報
Load
ObjectA
DoInit
Load
ObjectB
登録
DoInit
登録
DoContact
AddObserver
Return
Return
DoStart
DoStart

図１６

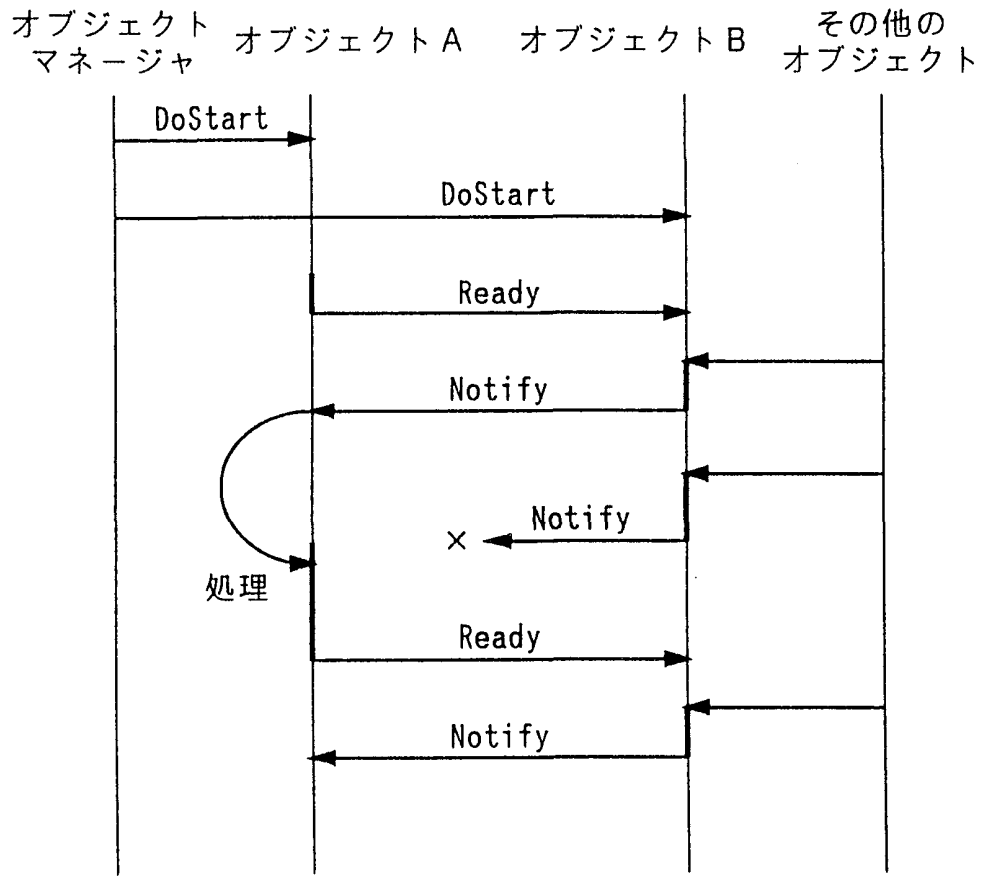

オブジェクト
マネージャ
オブジェクトA
オブジェクトB
その他の
オブジェクト
DoStart
DoStart
Ready
Notify
Notify
×
処理
Ready
Notify

図１７

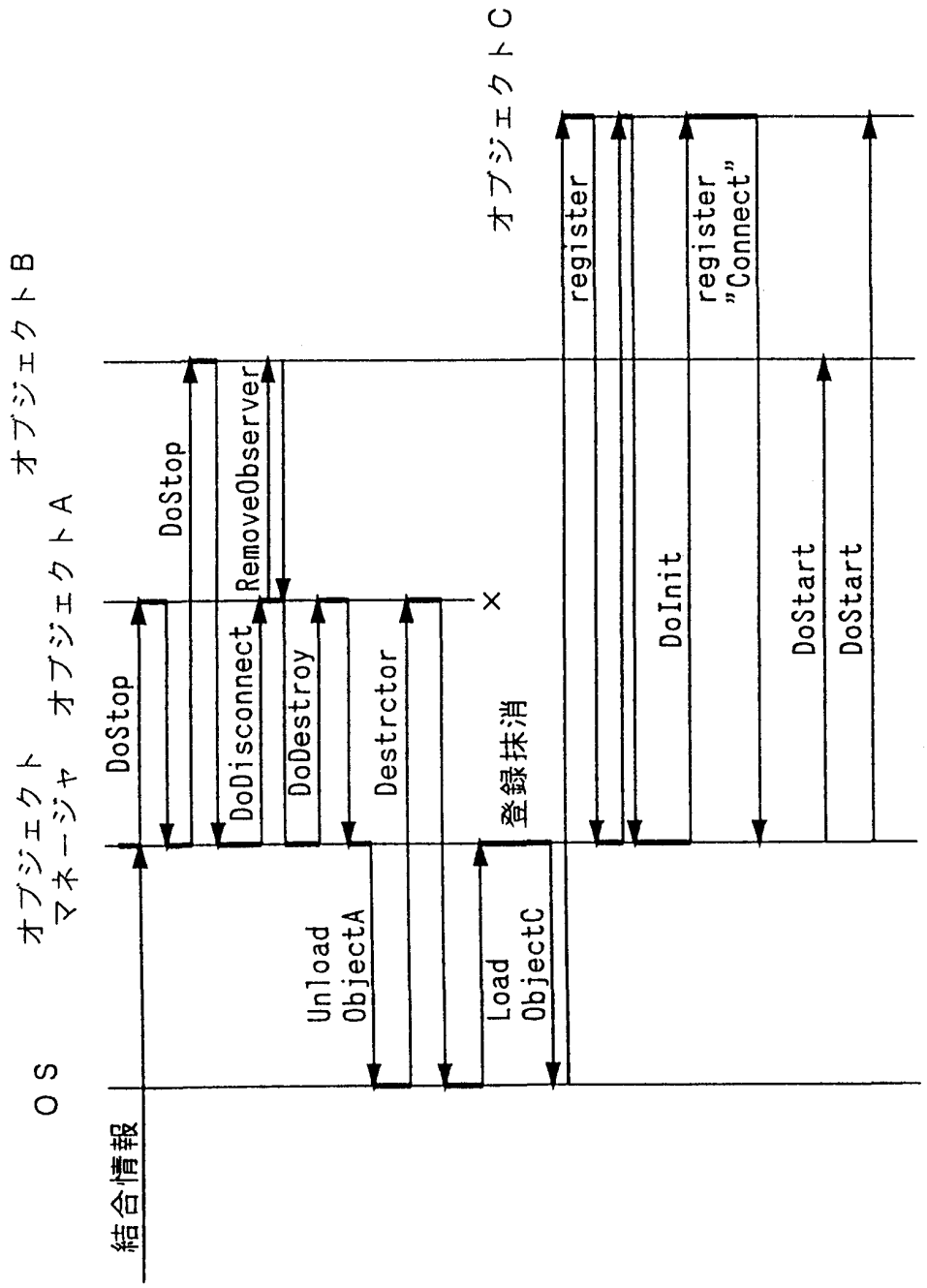

OS
オブジェクト
マネージャ
オブジェクトA
オブジェクトB
オブジェクトC
結合情報
DoStop
DoStop
DoDisconnect
RemoveObserver
DoDestroy
Destrctor
Unload
ObjectA
登録抹消
Load
ObjectC
register
register
"Connect"
DoInit
DoStart
DoStart

図１８

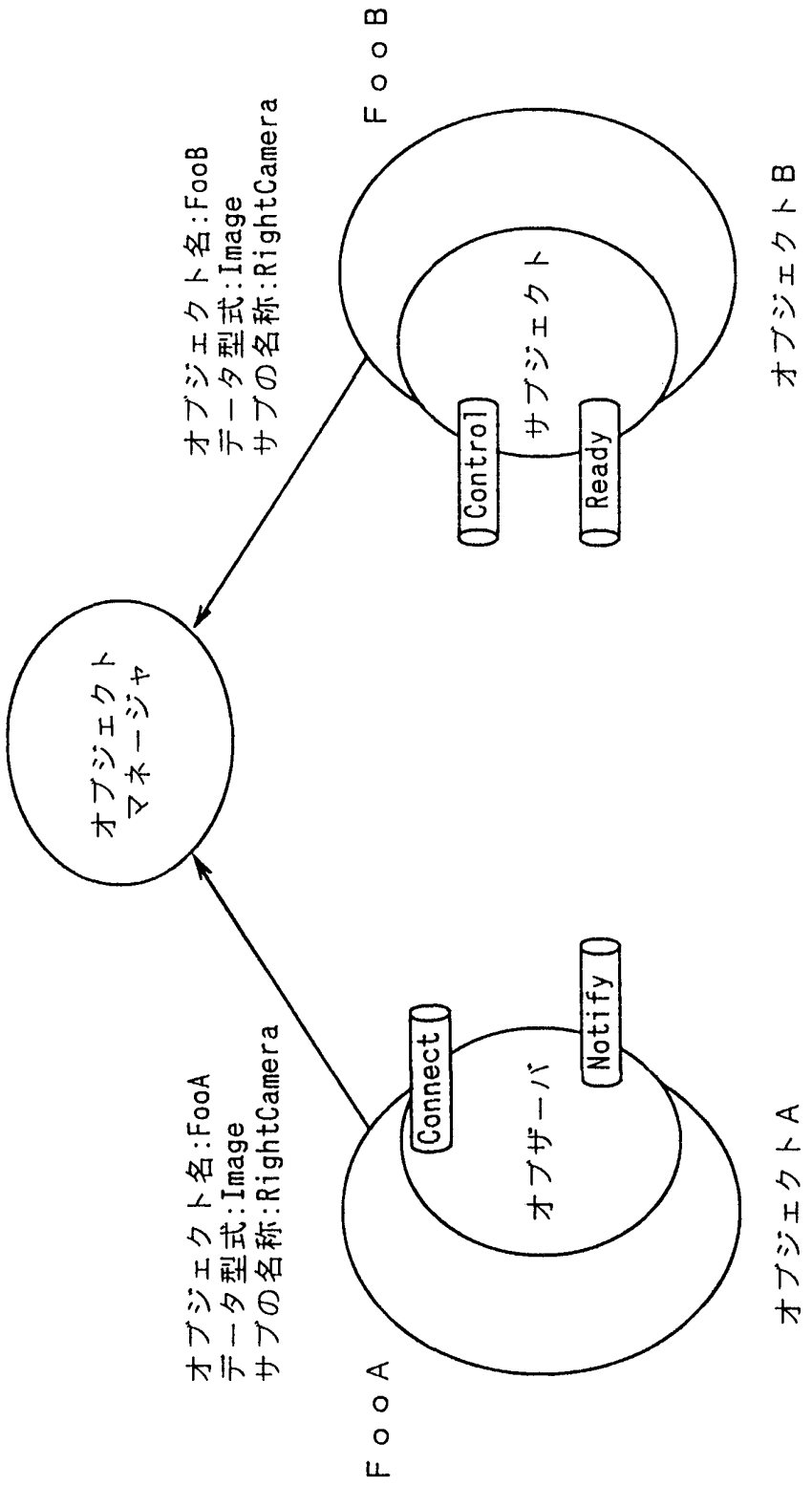

オブジェクト名:FooA
データ型式:Image
サブの名称:RightCamera
オブジェクト名:FooB
データ型式:Image
サブの名称:RightCamera
オブジェクト
マネージャ
FooA
FooB
オブジェクトA
オブジェクトB
オブザーバ
サブジェクト
Connect
Notify
Control
Ready

図１９

FooB.Image.RightCamera.S,FooA.Image.RightCamera.O

連想記憶1 ↓

Key:"FooA.Image.RightCamera.O"

出力:"FooB.Image.RightCamera.S"

図２０

FooB.Image.RightCamera、1,5.S

連想記憶2 ↓

Key:FooB.Image.RightCamera.S

出力:1,5

FooA.Image.RightCamera、2,5.O

連想記憶2 ↓

Key:FooA.Image.RightCamera.O

出力:2,5

図２１

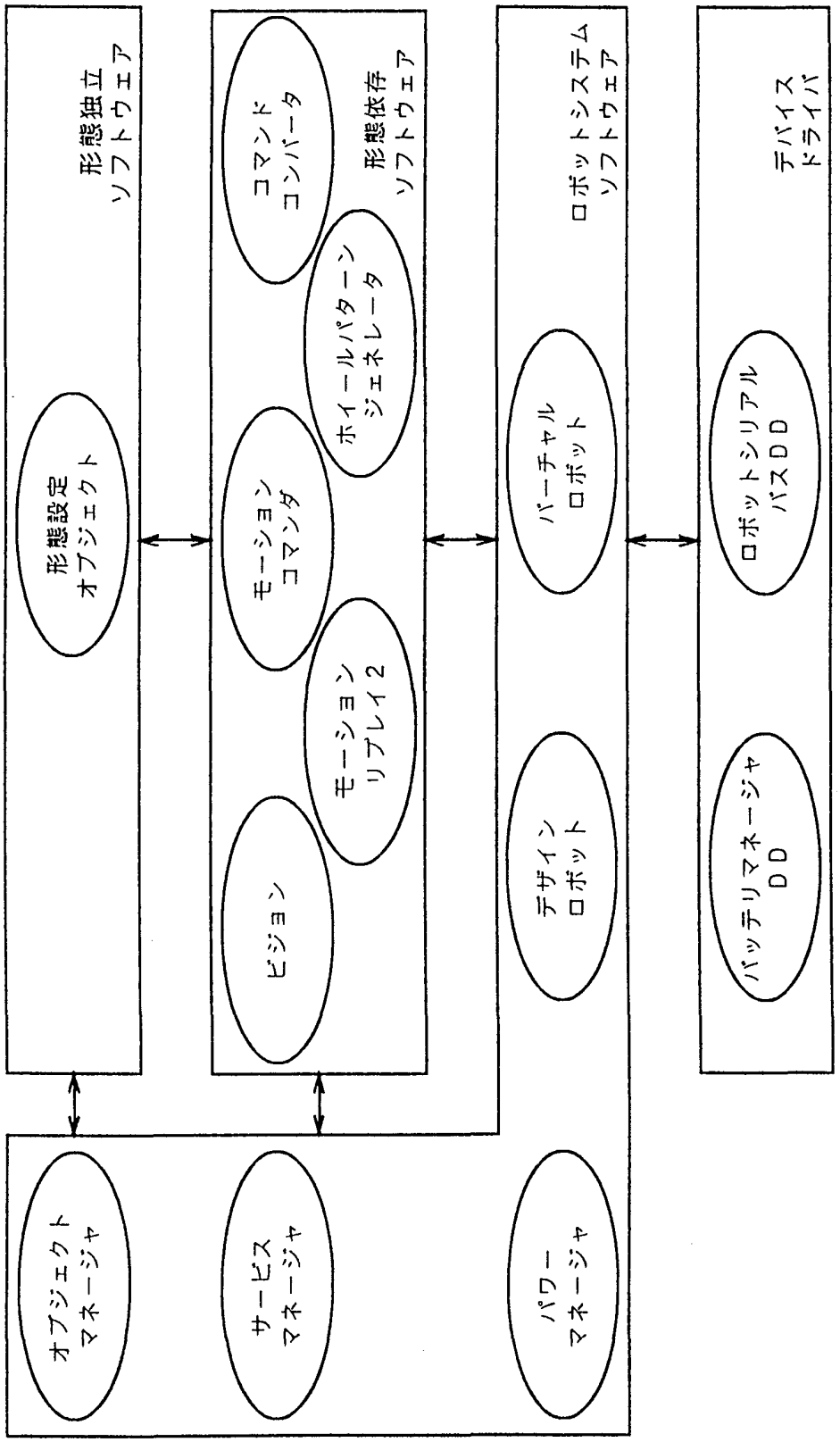

形態独立
ソフトウェア
形態設定
オブジェクト
形態依存
ソフトウェア
コマンド
コンバータ
ホイールパターン
ジェネレータ
モーション
コマンダ
モーション
リプレイ2
ビジョン
ロボットシステム
ソフトウェア
バーチャル
ロボット
デザイン
ロボット
パワー
マネージャ
デバイス
ドライバ
ロボットシリアル
バスDD
バッテリマネージャ
DD
オブジェクト
マネージャ
サービス
マネージャ

図22

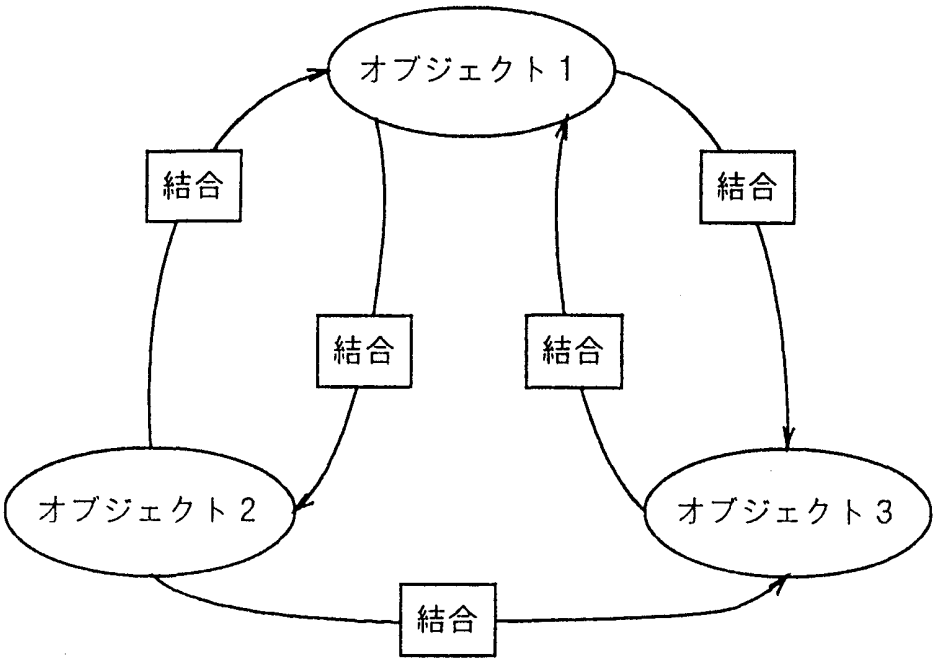

オブジェクト１
結合
結合
結合
結合
オブジェクト２
オブジェクト３
結合

図２３

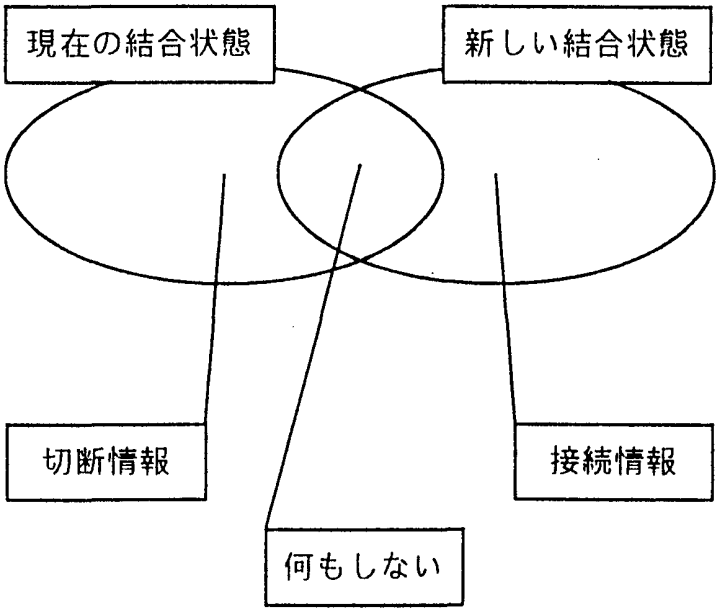

現在の結合状態
新しい結合状態
切断情報
何もしない
接続情報

図２４

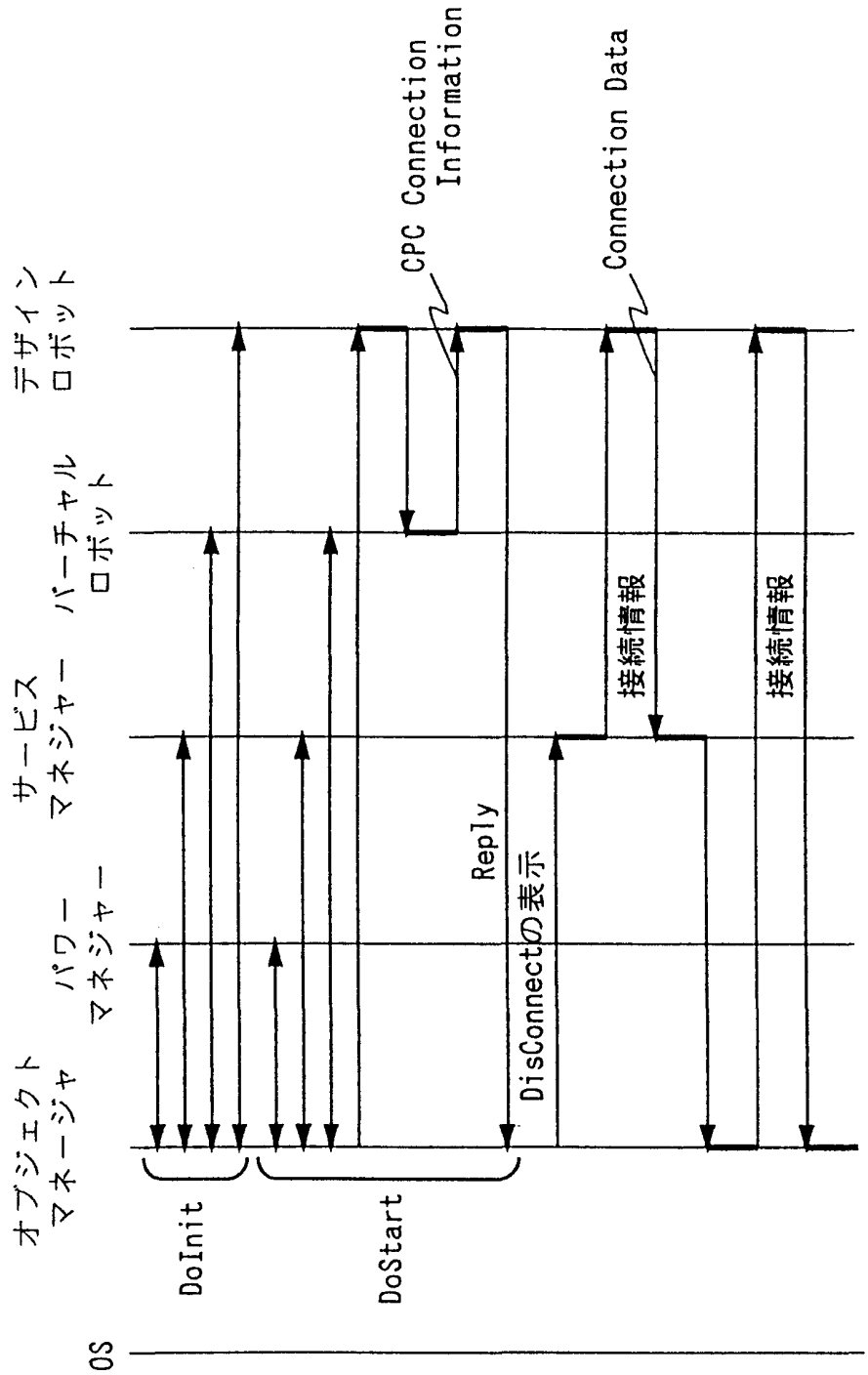

オブジェクト
マネージャ
パワー
マネージャー
サービス
マネジャー
バーチャル
ロボット
デザイン
ロボット
OS
DoInit
DoStart
Reply
DisConnectの表示
CPC Connection
Information
Connection Data
接続情報
接続情報

図２５

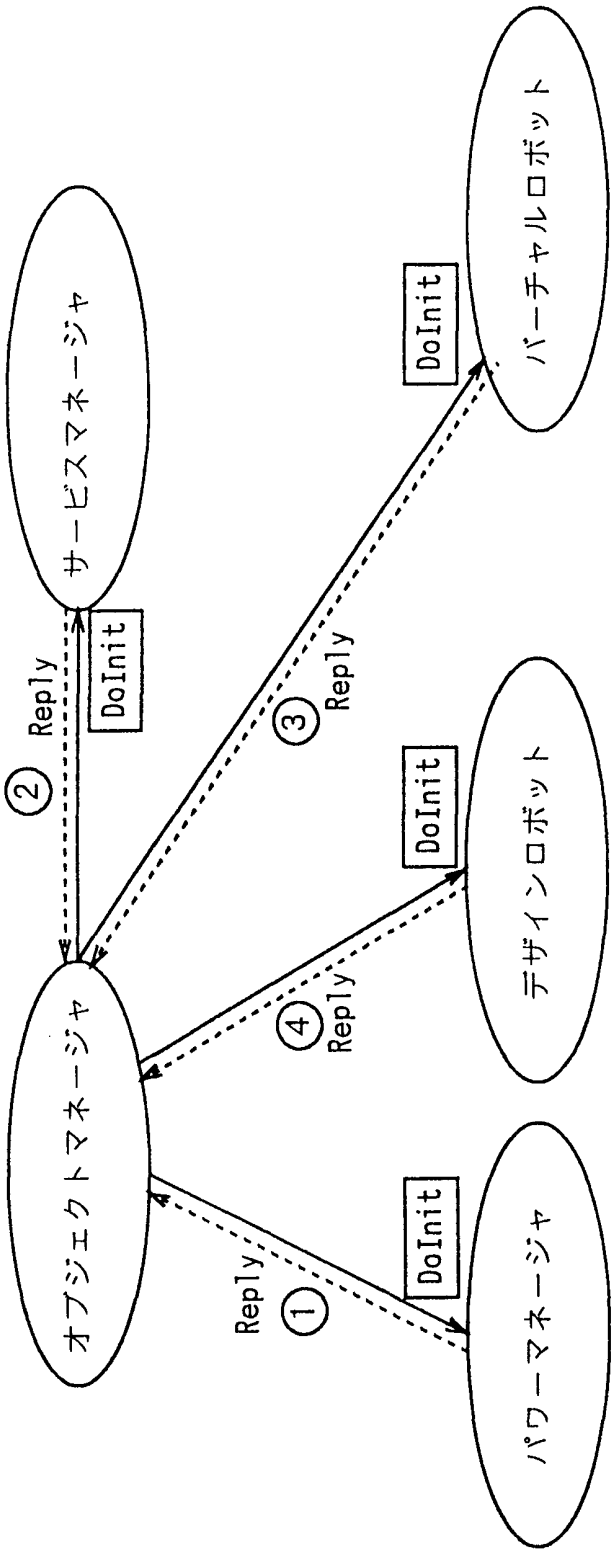

オブジェクトマネージャ
サービスマネージャ
パワーマネージャ
デザインロボット
バーチャルロボット
DoInit
Reply
①
②
③
④

図２６

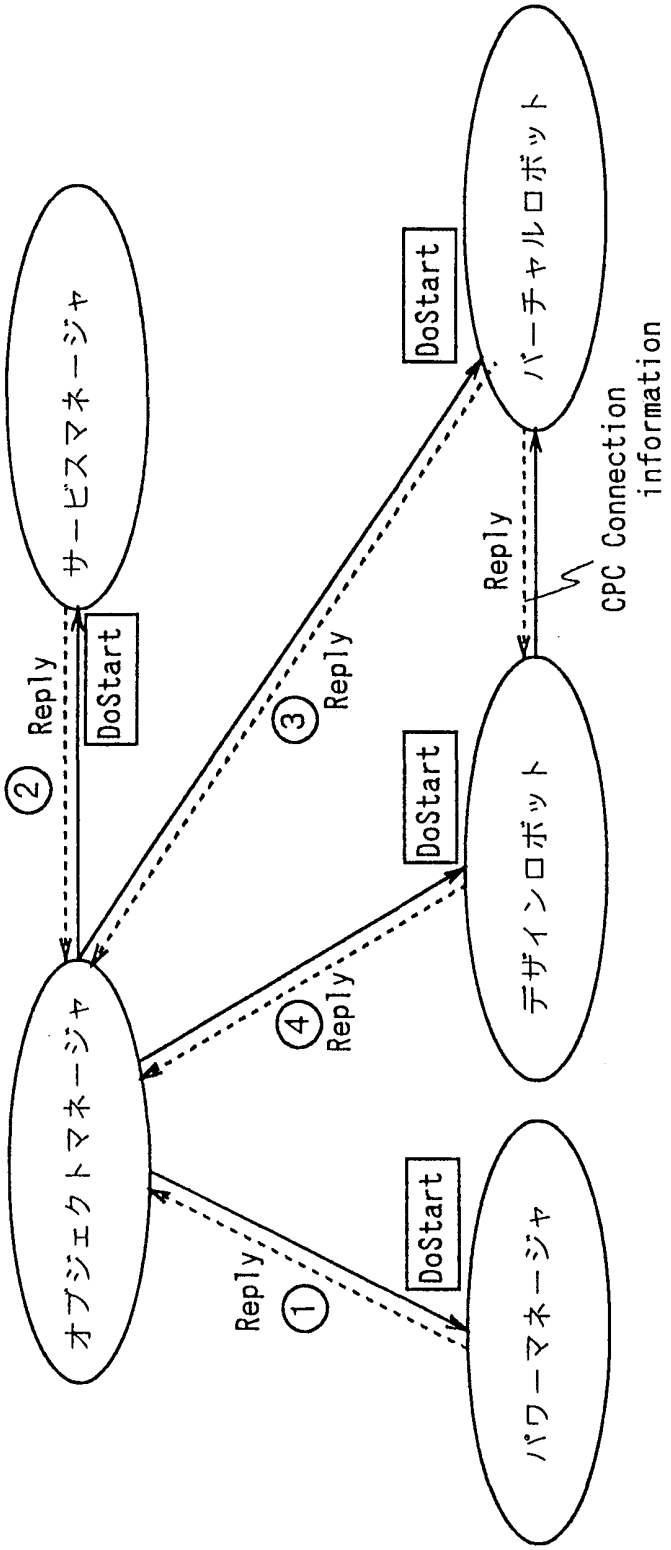

オブジェクトマネージャ
サービスマネージャ
バーチャルロボット
デザインロボット
パワーマネージャ
DoStart
Reply
①
②
③
④
CPC Connection information

図２７

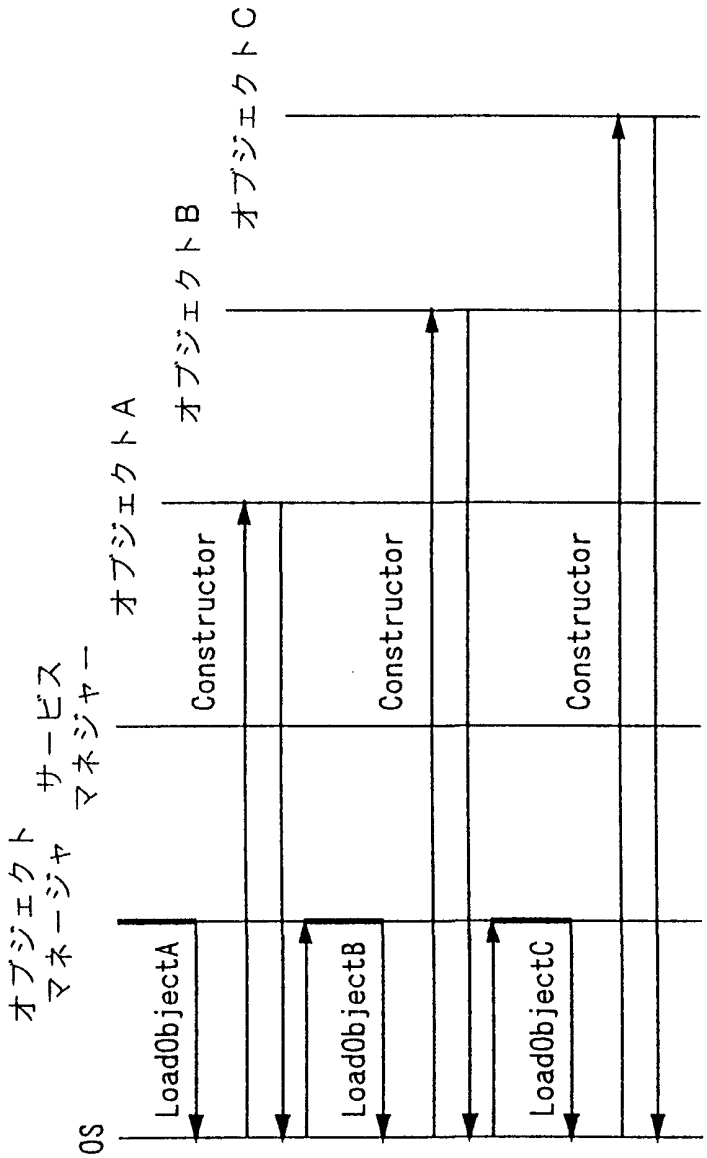

オブジェクトマネージャ
サービスマネージャー
OS
LoadObjectA
LoadObjectB
LoadObjectC
Constructor
Constructor
Constructor
オブジェクトA
オブジェクトB
オブジェクトC

図２８

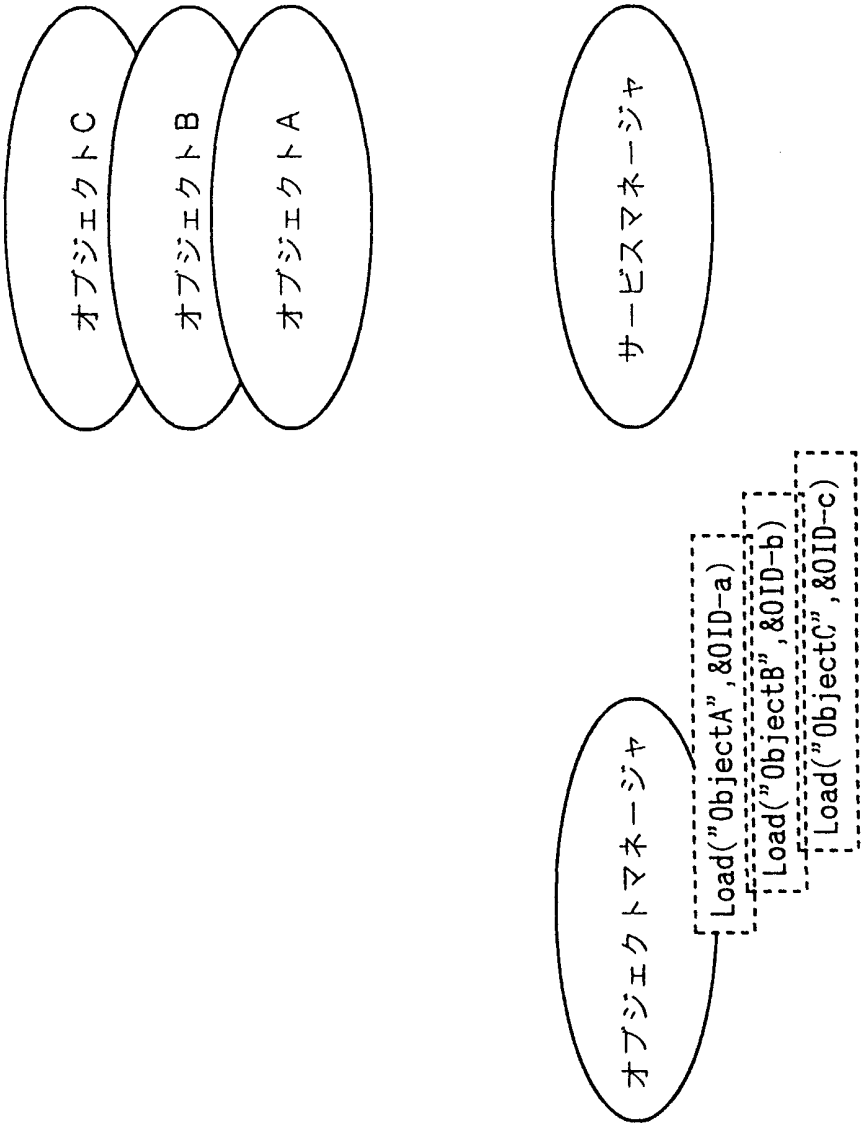

オブジェクトC
オブジェクトB
オブジェクトA
サービスマネージャ
オブジェクトマネージャ
Load("ObjectA",&OID-a)
Load("ObjectB",&OID-b)
Load("ObjectC",&OID-c)

図２９

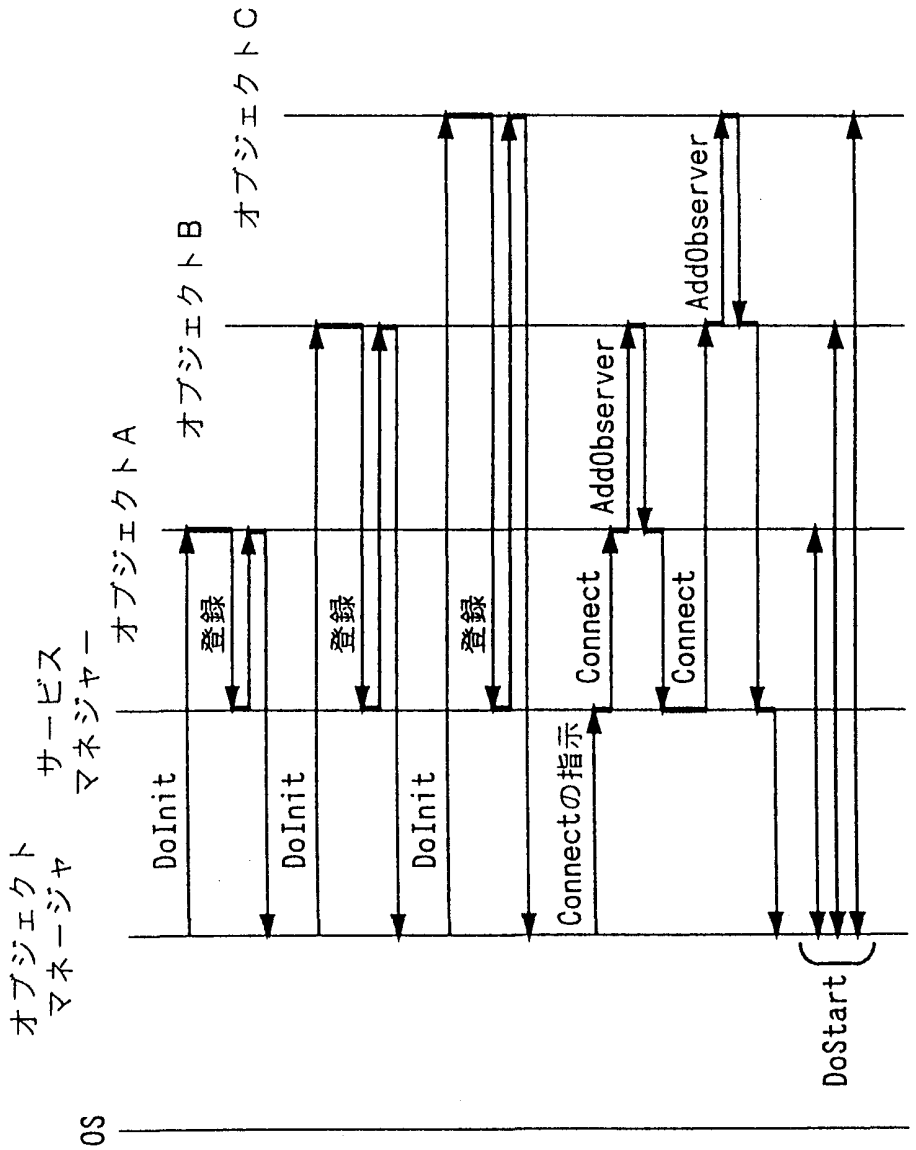

オブジェクトマネージャ
サービスマネジャー
オブジェクトA
オブジェクトB
オブジェクトC
OS
DoInit
DoInit
DoInit
登録
登録
登録
Connectの指示
Connect
Connect
AddObserver
AddObserver
DoStart

図３０

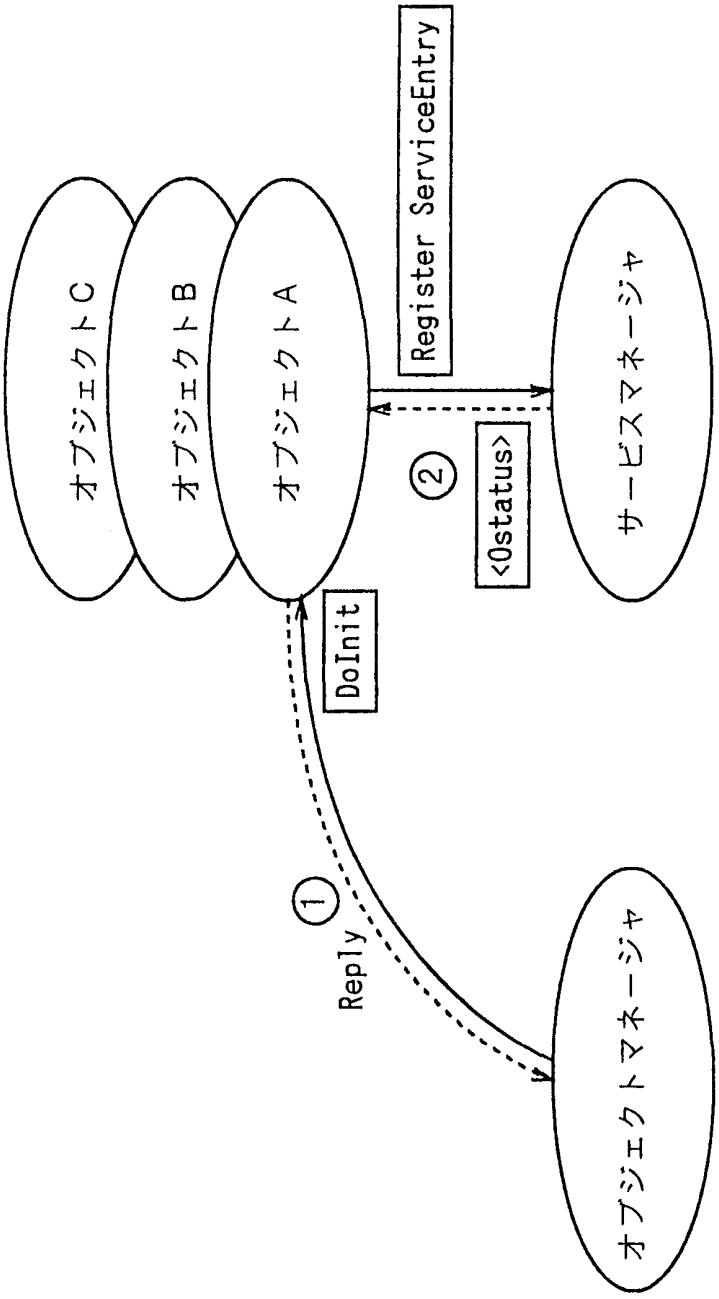

オブジェクトC
オブジェクトB
オブジェクトA
Register ServiceEntry
サービスマネージャ
②
<Ostatus>
DoInit
①
Reply
オブジェクトマネージャ

図３１

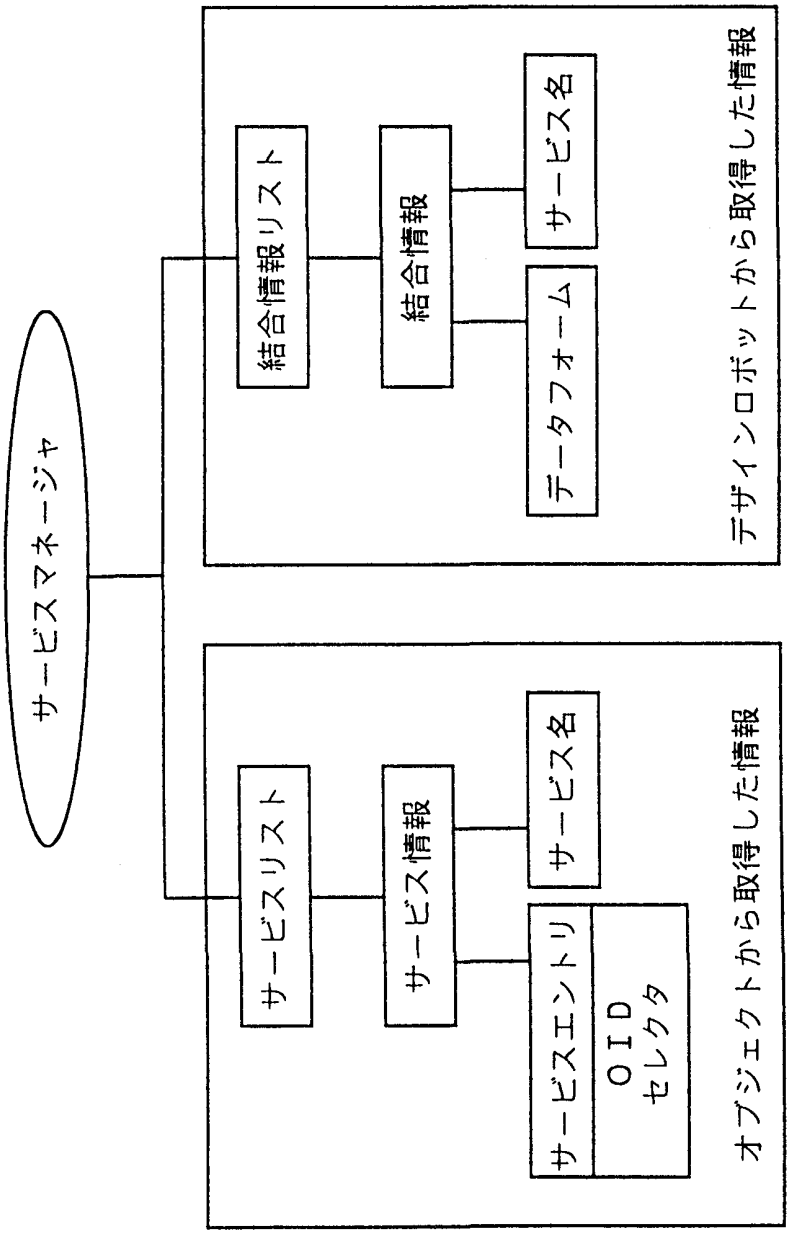

サービスマネージャ
オブジェクトから取得した情報
サービスリスト
サービス情報
サービス名
サービスエントリ
OID
セレクタ
デザインロボットから取得した情報
結合情報リスト
結合情報
サービス名
データフォーム

図３２

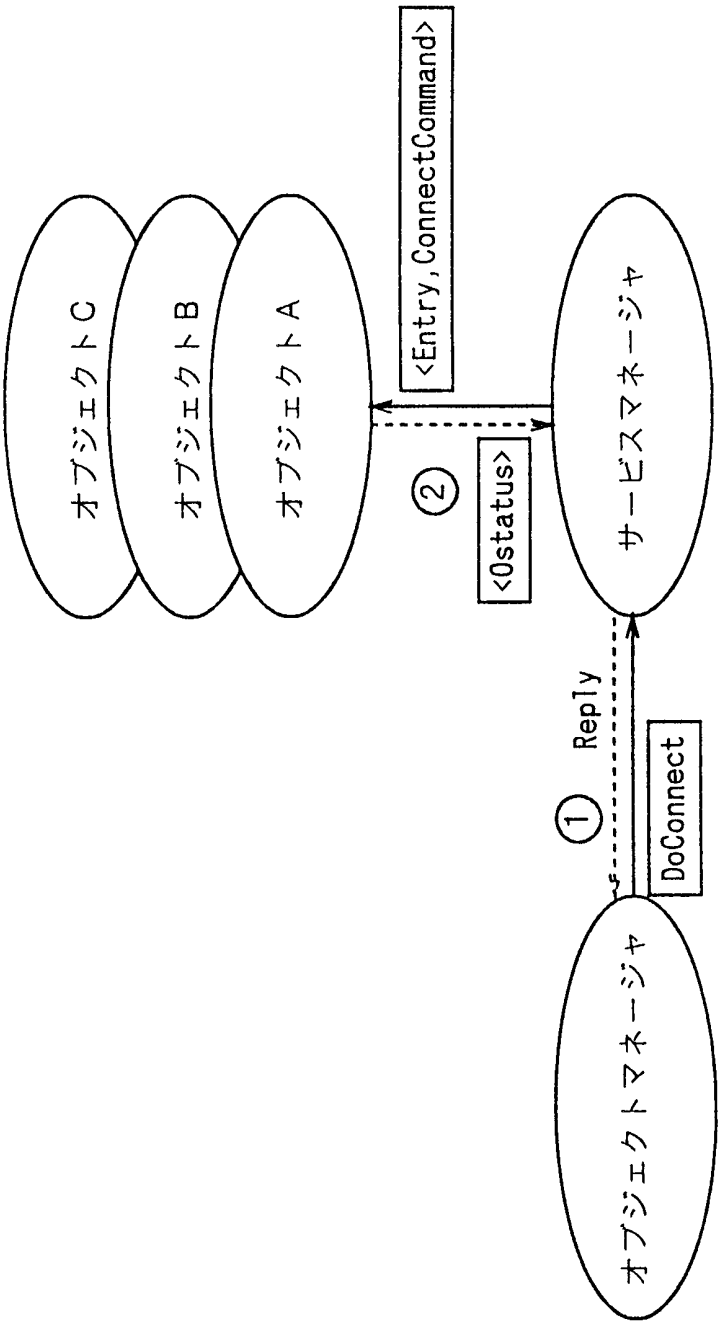

オブジェクトマネージャ
DoConnect
① Reply
サービスマネージャ
② <Ostatus>
<Entry,ConnectCommand>
オブジェクトA
オブジェクトB
オブジェクトC

図３３

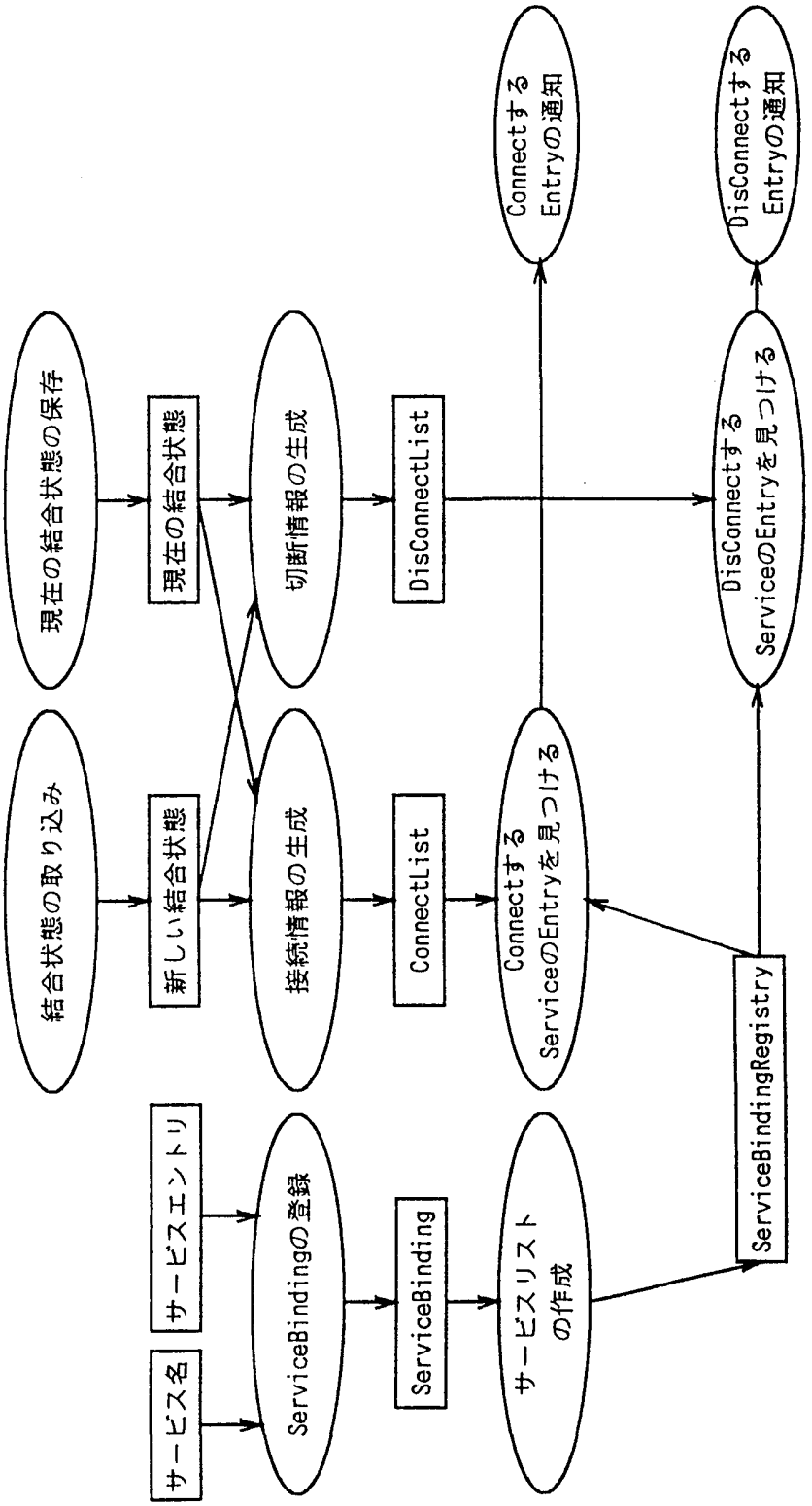

結合状態の取り込み
新しい結合状態
現在の結合状態の保存
現在の結合状態
接続情報の生成
切断情報の生成
ConnectList
DisConnectList
ConnectするServiceのEntryを見つける
DisConnectするServiceのEntryを見つける
Connectする Entryの通知
DisConnectする Entryの通知
サービス名
サービスエントリ
ServiceBindingの登録
ServiceBinding
サービスリストの作成
ServiceBindingRegistry

図３４

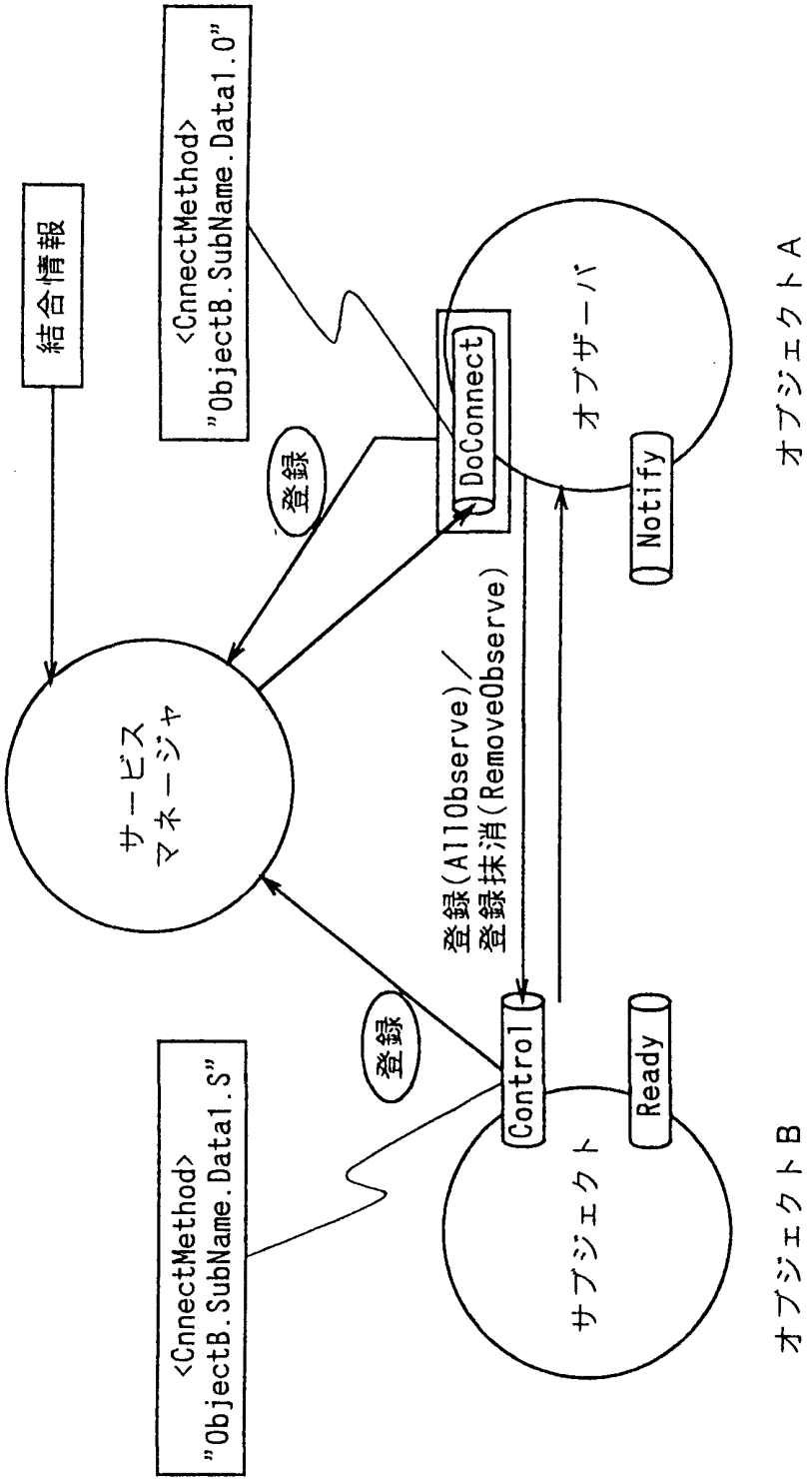

結合情報
<CnnectMethod>
"ObjectB.SubName.Data1.0"
登録
DoConnect
オブザーバ
Notify
オブジェクトA
サービス
マネージャ
登録(AllObserve)／
登録抹消(RemoveObserve)
登録
Control
Ready
サブジェクト
オブジェクトB
<CnnectMethod>
"ObjectB.SubName.Data1.S"

図３５

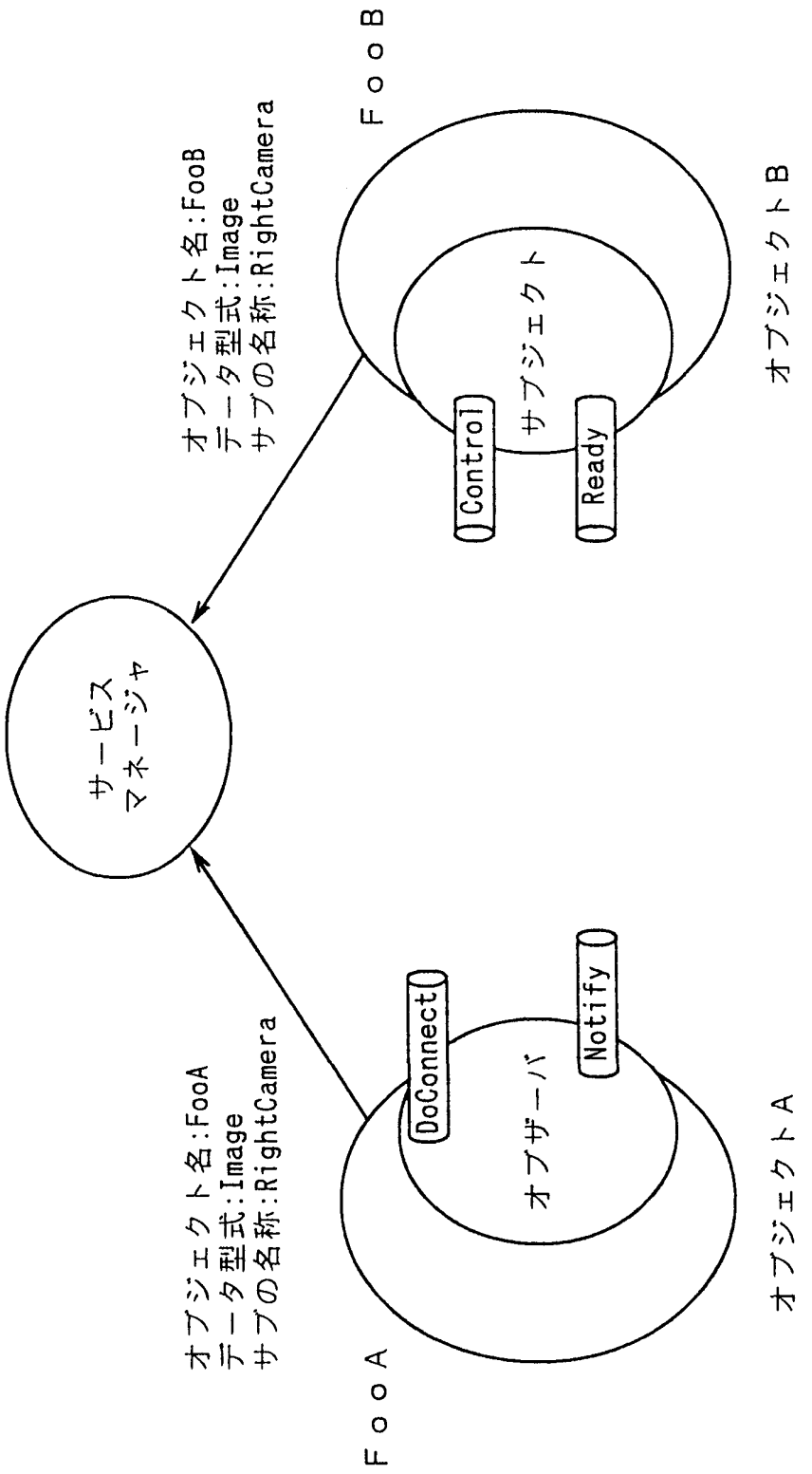

サービス
マネージャ
オブジェクト名:FooA
データ型式:Image
サブの名称:RightCamera
オブジェクト名:FooB
データ型式:Image
サブの名称:RightCamera
FooA
オブジェクトA
オブザーバ
DoConnect()
Notify ()
FooB
オブジェクトB
サブジェクト
Control
Ready

図３６

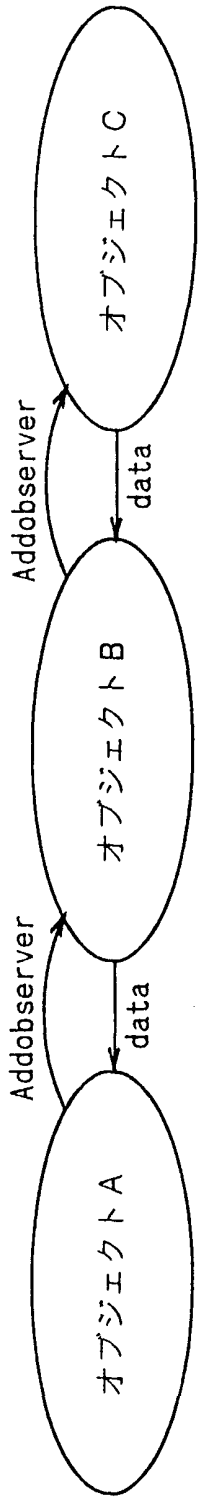

オブジェクトC
Addobserver
data
オブジェクトB
Addobserver
data
オブジェクトA

図３７

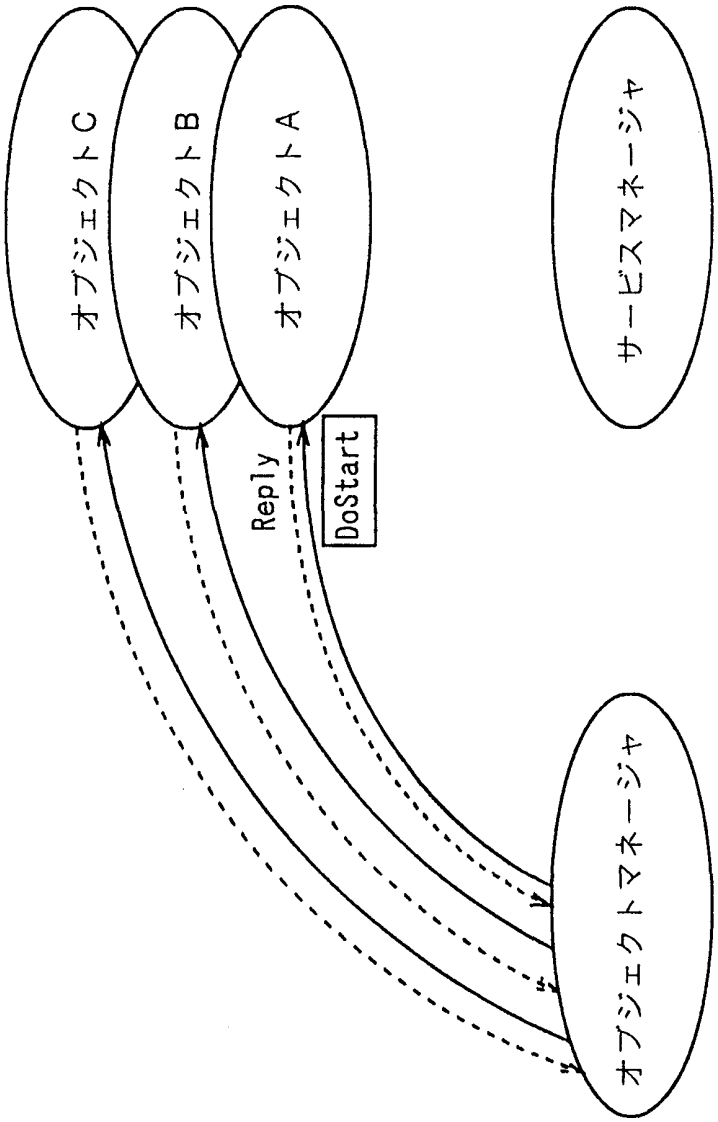

オブジェクトC
オブジェクトB
オブジェクトA
サービスマネージャ
DoStart
Reply
オブジェクトマネージャ

図３８

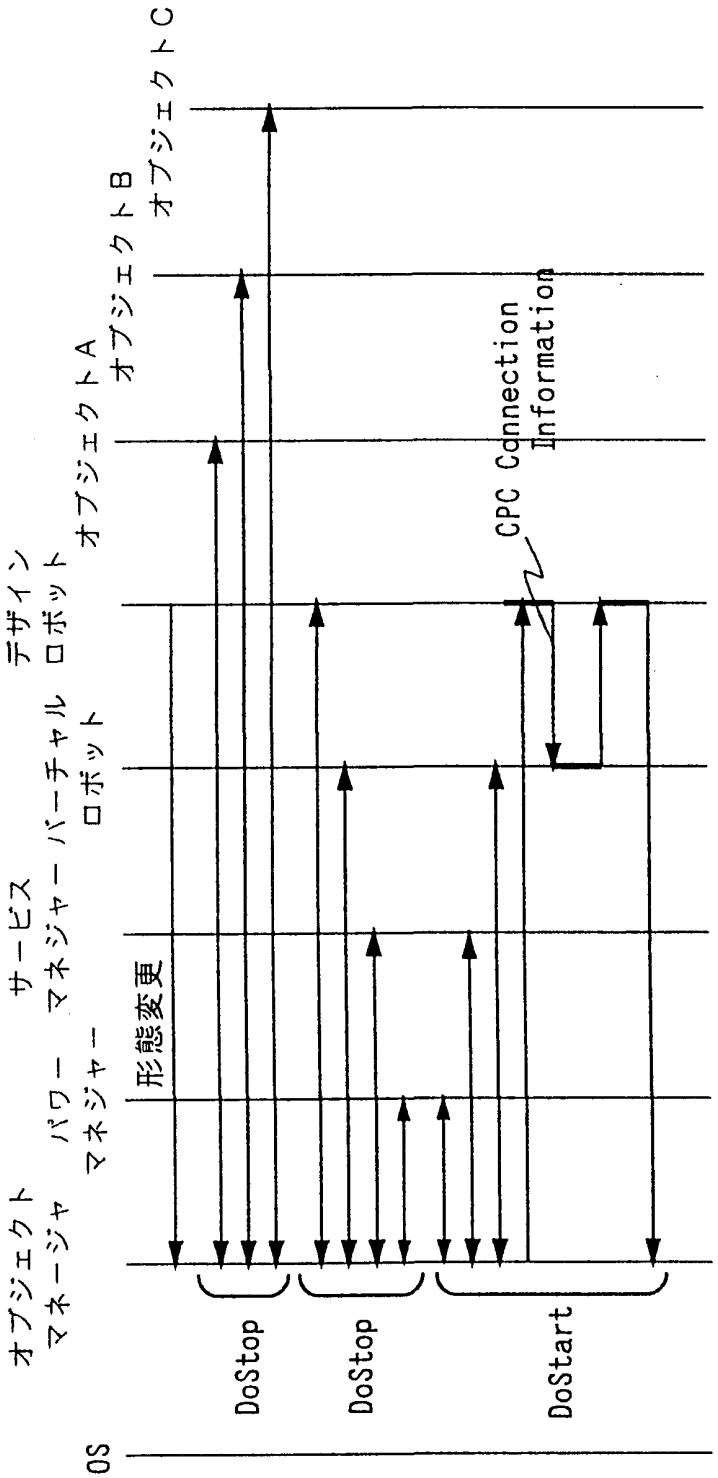

オブジェクトマネージャ
パワーマネジャー
サービスマネジャー
バーチャルロボット
デザインロボット
オブジェクトA
オブジェクトB
オブジェクトC
形態変更
CPC Connection Information
DoStop
DoStop
DoStart
OS

図３９

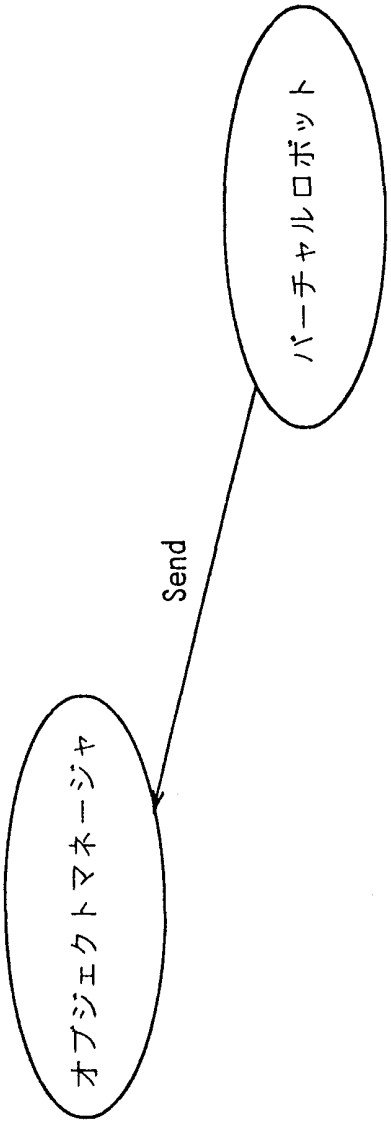


図４０

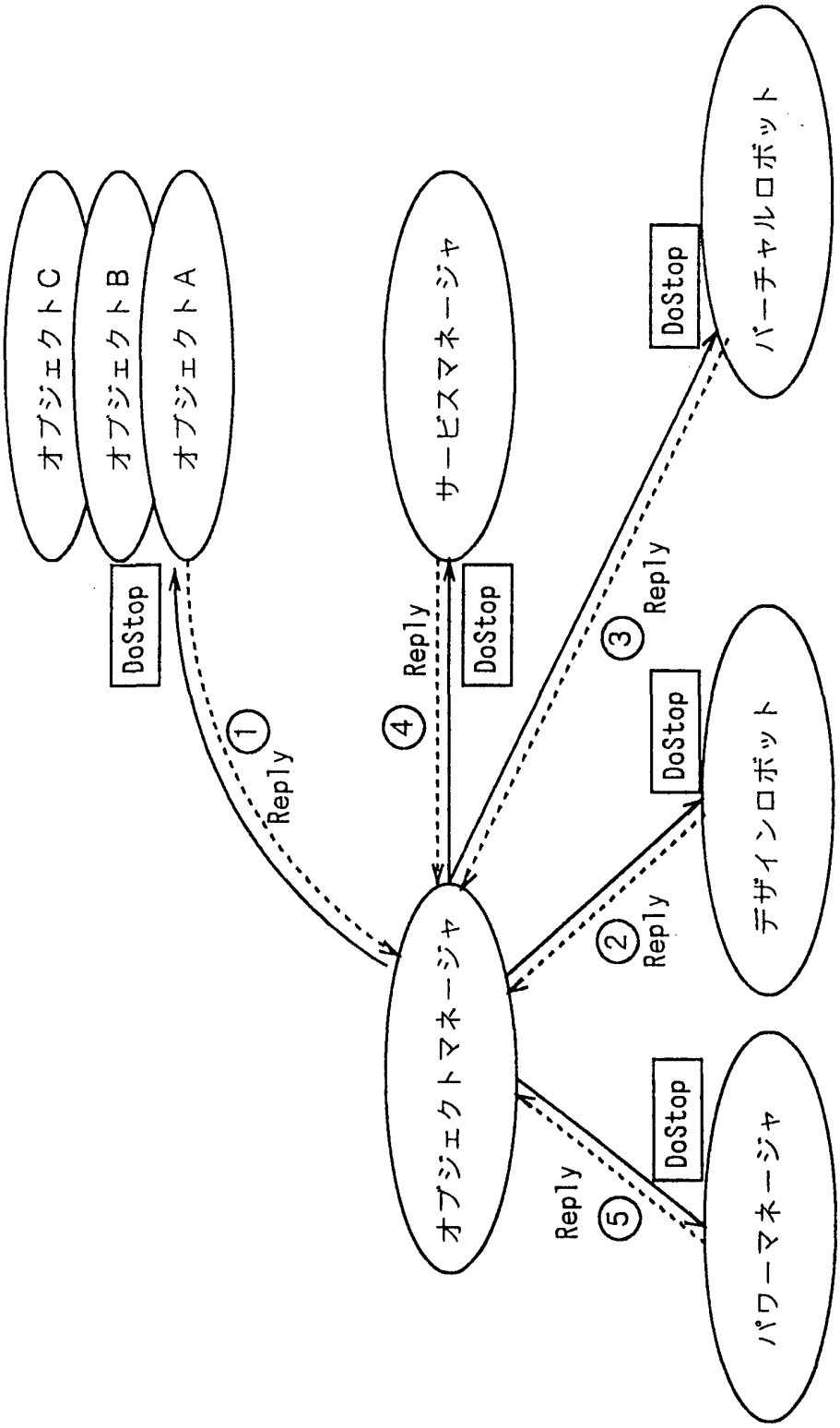

オブジェクトC
オブジェクトB
オブジェクトA
サービスマネージャ
バーチャルロボット
デザインロボット
パワーマネージャ
オブジェクトマネージャ
DoStop
Reply
①
②
③
④
⑤

図４１

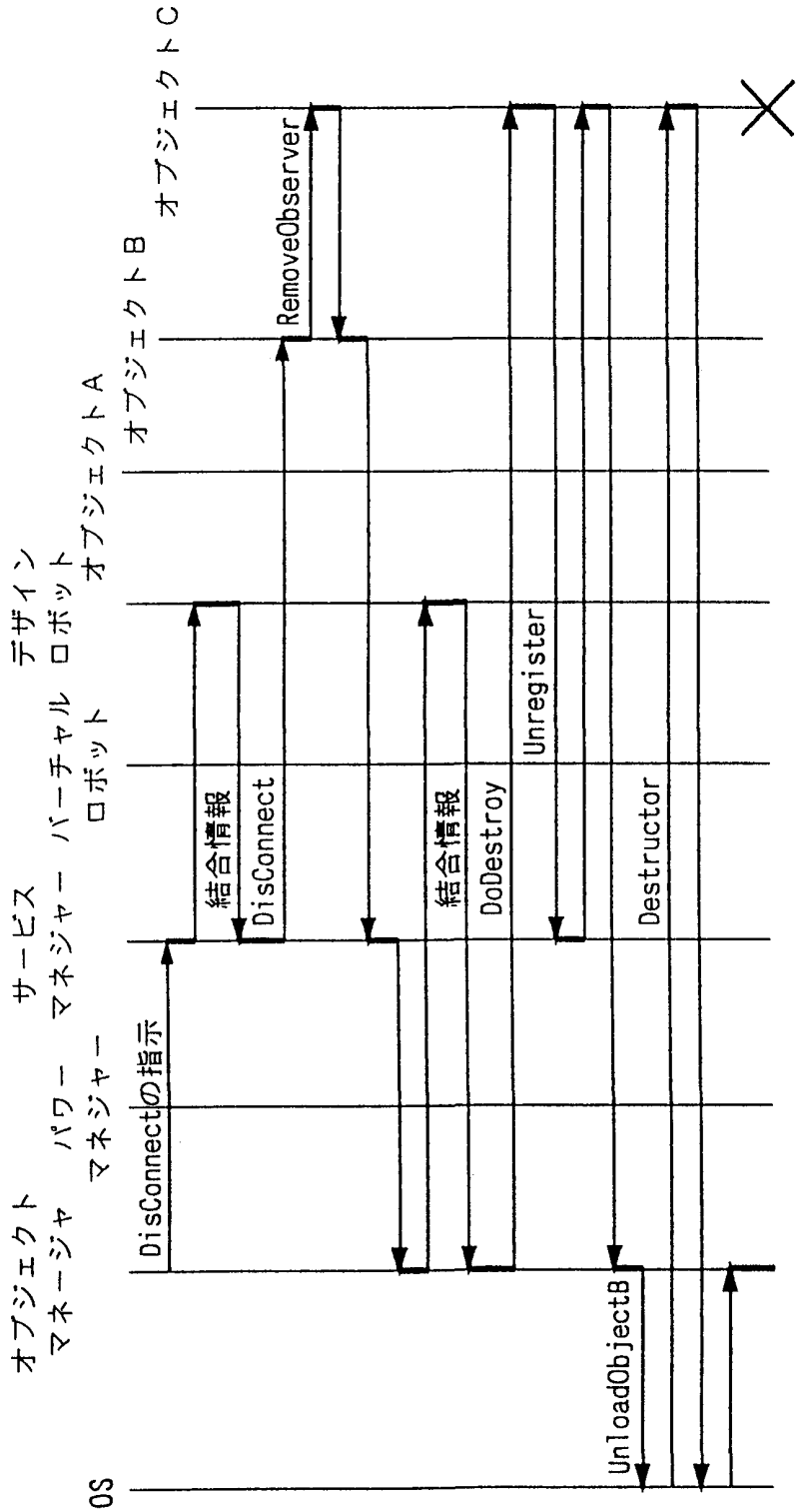

オブジェクト
マネージャ
パワー
マネジャー
サービス
マネジャー
バーチャル
ロボット
デザイン
ロボット
オブジェクトA
オブジェクトB
オブジェクトC
OS
DisConnectの指示
結合情報
DisConnect
結合情報
DoDestroy
Unregister
RemoveObserver
Destructor
UnloadObjectB

図４２

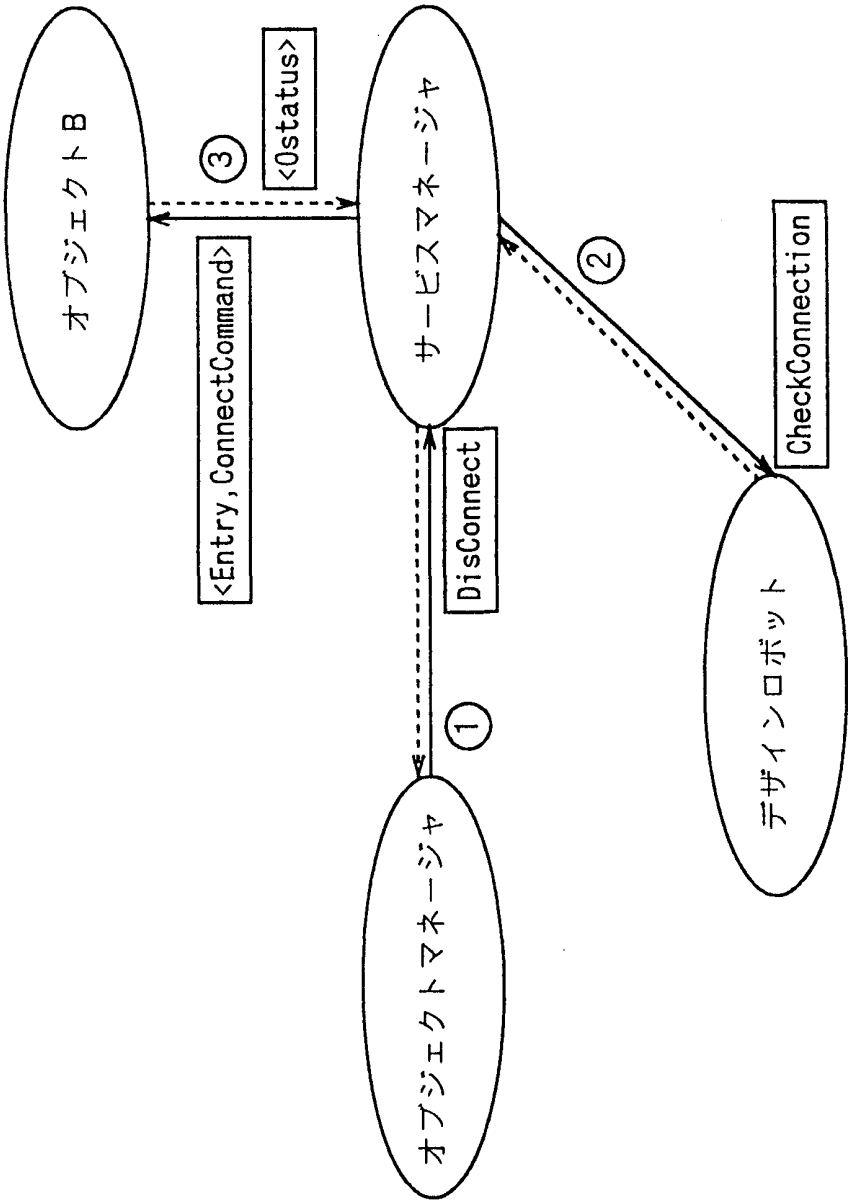

オブジェクトB
③
<Ostatus>
<Entry,ConnectCommand>
サービスマネージャ
②
CheckConnection
DisConnect
①
オブジェクトマネージャ
デザインロボット

図４３

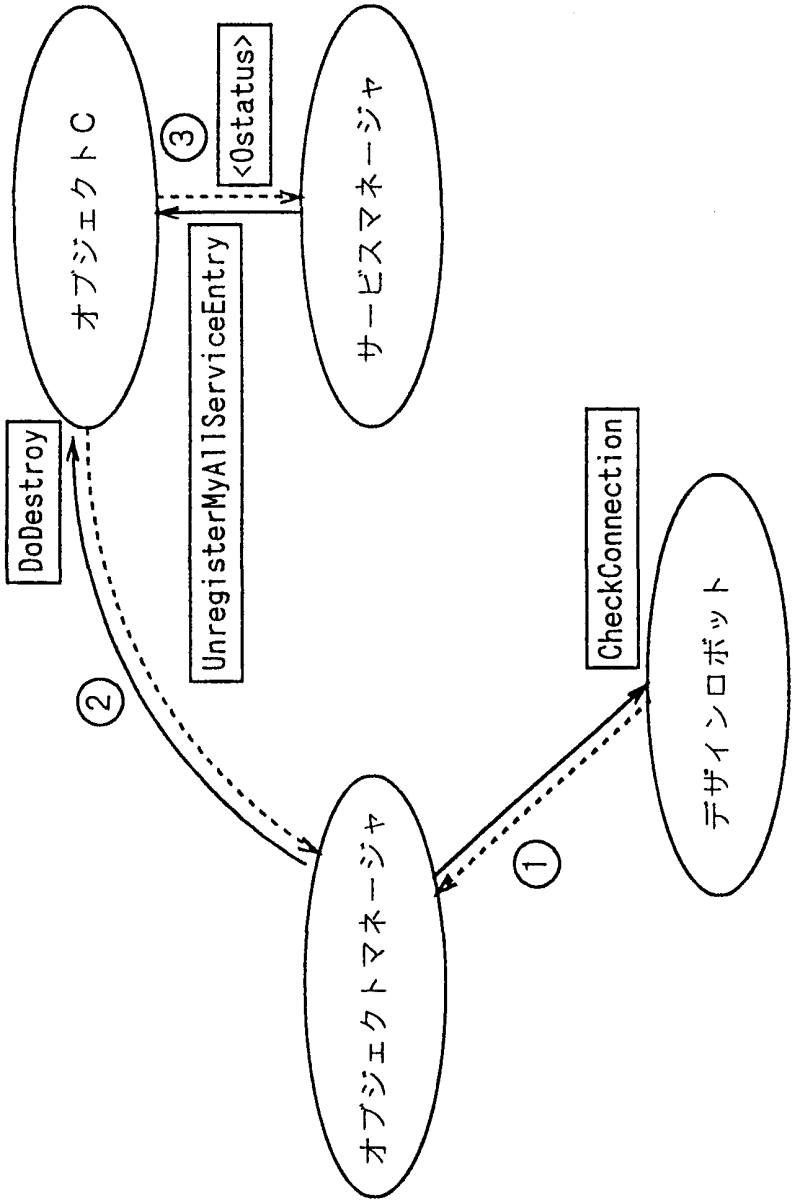

オブジェクトC
サービスマネージャ
<Ostatus>
③
UnregisterMyAllServiceEntry
DoDestroy
②
CheckConnection
デザインロボット
オブジェクトマネージャ
①

図４４

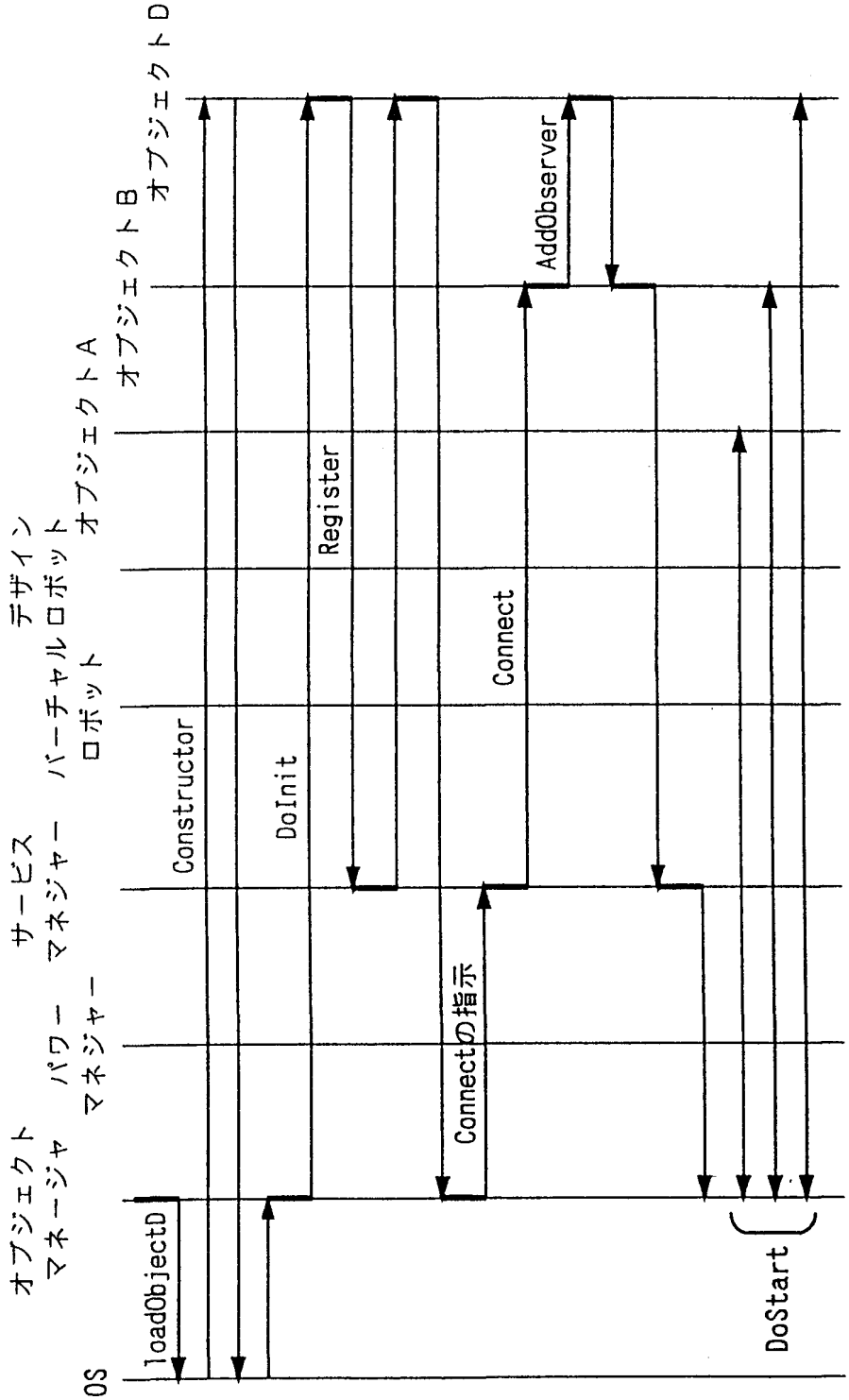

オブジェクトマネージャ
パワーマネジャー
サービスマネジャー
バーチャルロボットデザイン
ロボットオブジェクトA
オブジェクトB
オブジェクトD
OS
loadObjectD
Constructor
DoInit
Register
Connectの指示
Connect
AddObserver
DoStart

図45

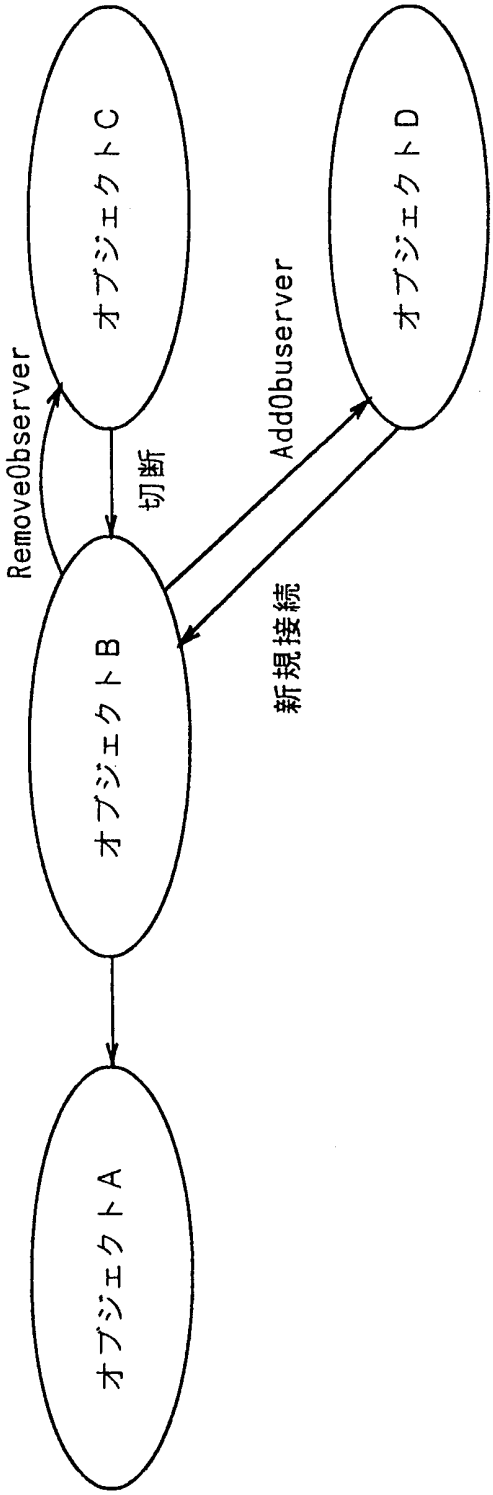

オブジェクトC
オブジェクトD
RemoveObserver
切断
AddObuserver
新規接続
オブジェクトB
オブジェクトA

図４６

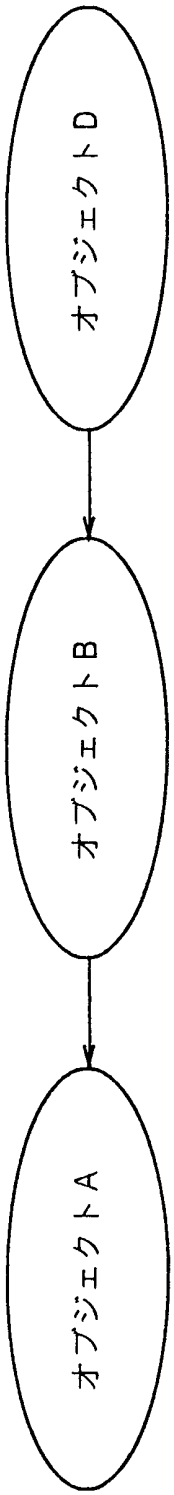


図４７

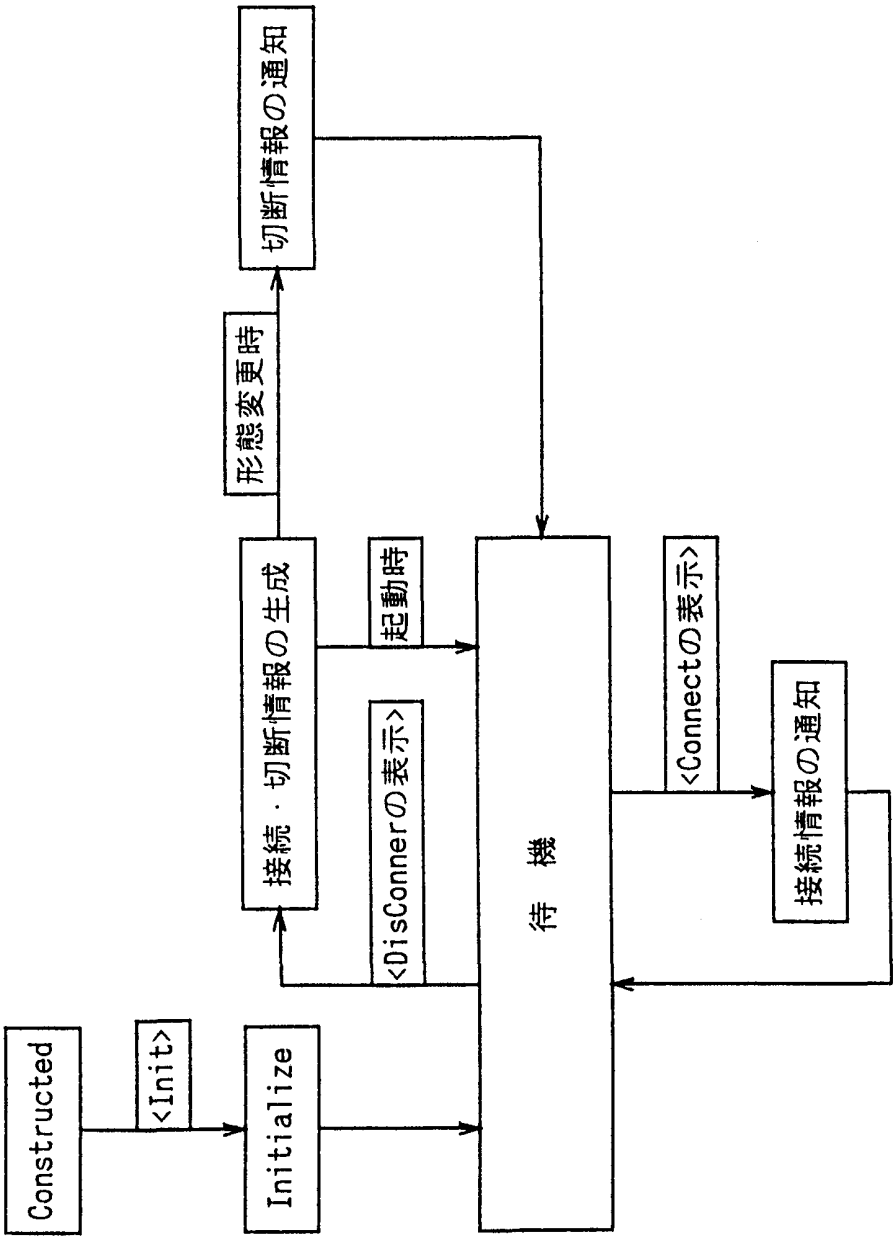

Constructed
<Init>
Initialize
待　機
接続・切断情報の生成
起動時
<DisConnerの表示>
形態変更時
切断情報の通知
<Connectの表示>
接続情報の通知

図４８

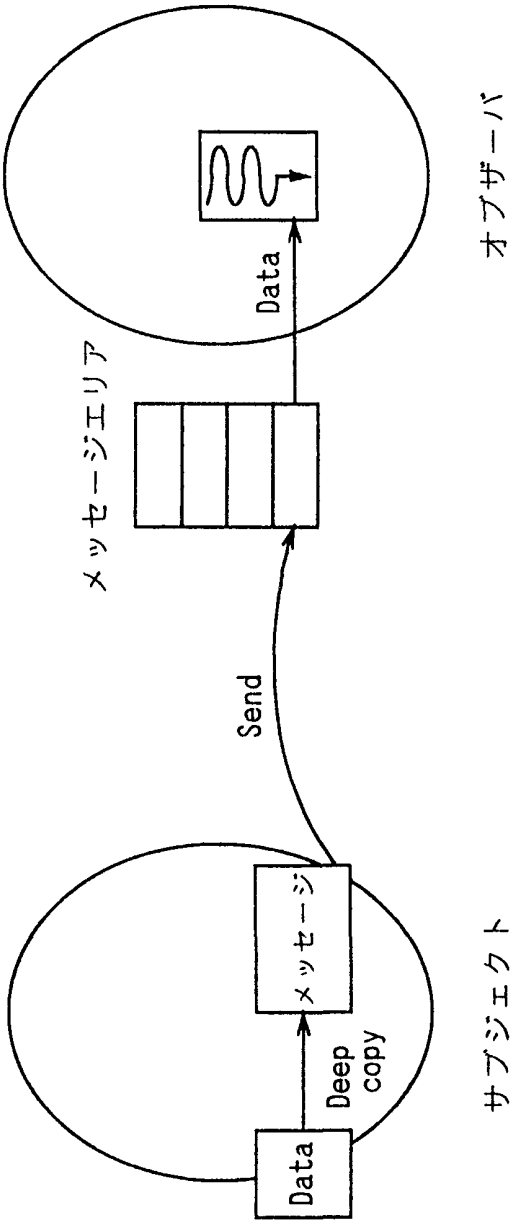


図４９

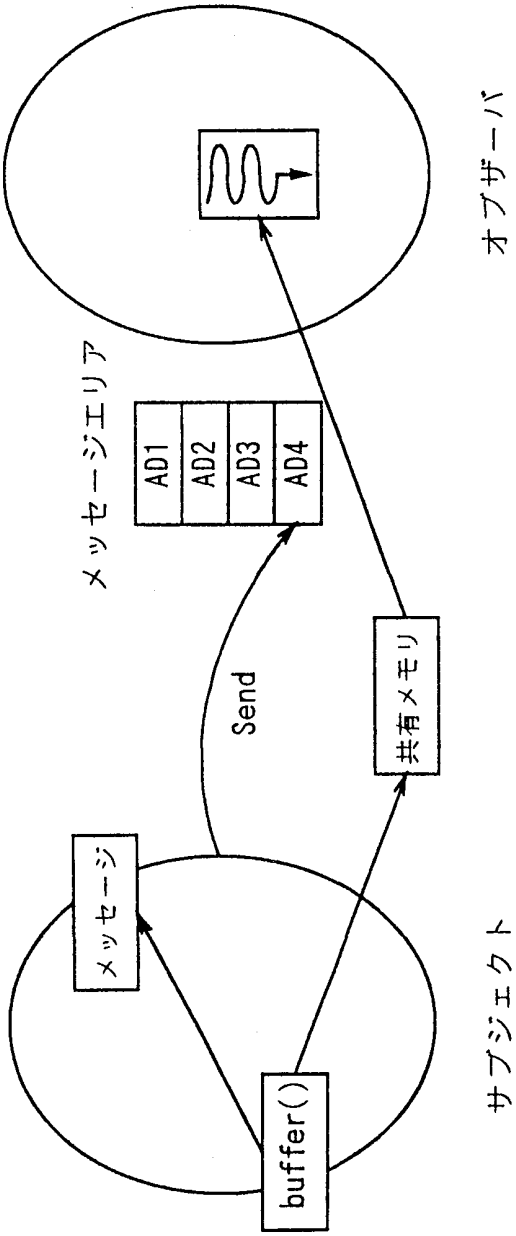

オブザーバ
メッセージエリア
AD1
AD2
AD3
AD4
Send
共有メモリ
メッセージ
buffer()
サブジェクト

図５０

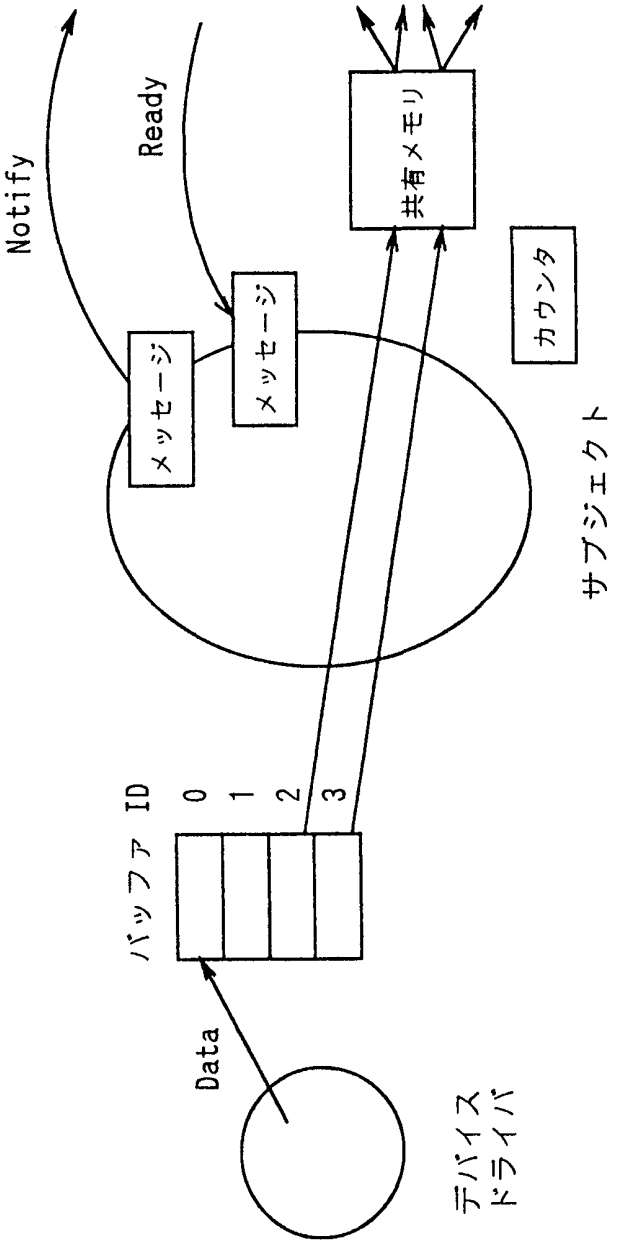

Notify
Ready
共有メモリ
メッセージ
メッセージ
カウンタ
サブジェクト
バッファ ID
0
1
2
3
Data
デバイス
ドライバ

図５１

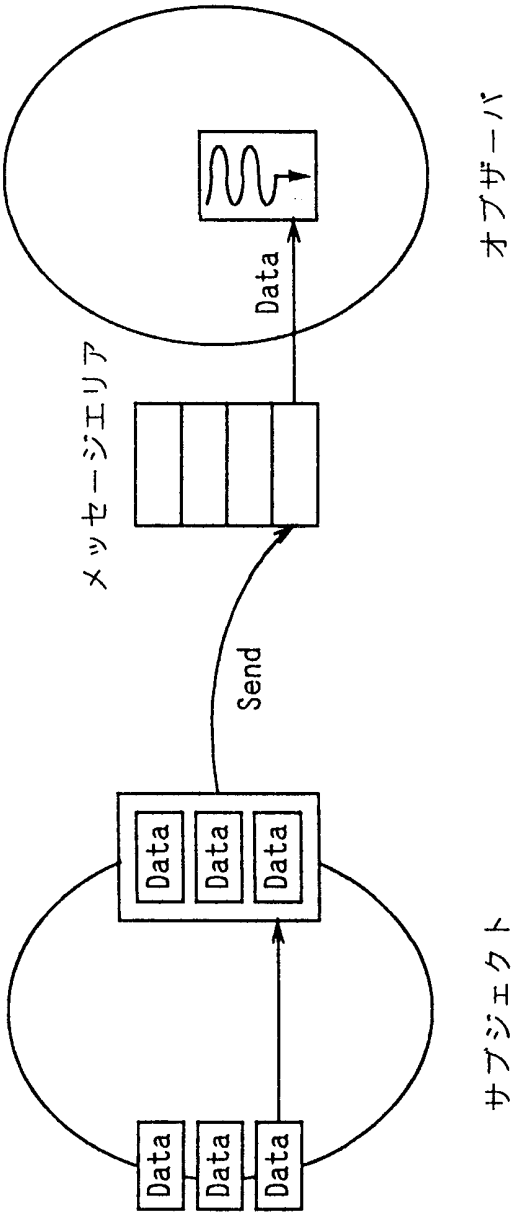

オブザーバ
Data
メッセージエリア
Send
Data
Data
Data
サブジェクト
Data
Data
Data

図５２

符　号　の　説　明

１……ロボット、２……本体、３……移動ユニット、１７……バッテリー

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP00/00123

A. CLASSIFICATION OF SUBJECT MATTER
Int7 G06F9/44 B25J 9/00

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int7 G06F9/44 B25J 9/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1971-1996 Jitsuyo Shinan Toroku Koho 1996-1997
Kokai Jitsuyo Shinan Koho 1971-1995 Toroku Jitsuyo Shinan Koho 1994-2000

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	Proceedings of the 1998 IEEE International Conference on Robotics & Automation(1998-5), H.Kitano,et al, "Sony Legged Robot for RoboCup Challenge",P.2605-2612	1-4,5-8,10-14,16-20,22,23
A	ibid.	9,15,21
A	IEEE Transactions on Software Engineering, Vol.23, No.12, (1997-12),D.B.Stewart,et al, "Design of Dynamically Reconfigurable Real-Time Software Using Port-Based Objects" P.759-776	1-23

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
11 April, 2000 (11.04.00)	16 May, 2000 (16.05.00)
Name and mailing address of the ISA/ Japanese Patent Office	Authorized officer
Facsimile No.	Telephone No.

Form PCT/ISA/210 (second sheet) (July 1992)

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ００／００１２３

A.　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int⁷　Ｇ０６Ｆ９／４４　　Ｂ２５Ｊ　９／００

B.　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int⁷　Ｇ０６Ｆ９／４４　　Ｂ２５Ｊ　９／００

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報　　１９７１－１９９６年
日本国公開実用新案公報　１９７１－１９９５年
日本国実用新案登録公報　１９９６－１９９７年
日本国登録実用新案公報　１９９４－２０００年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C.　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
X	Proceedings of the 1998 IEEE International Conference on Robotics & Automation(1998-5), H.Kitano,et al, "Sony Legged Robot for RoboCup Challenge", P.2605-2612	1-4,5-8,10-14,16-20,22,23
A	ibid.	9,15,21
A	IEEE Transactions on Software Engineering,Vol.23,No.12, (1997-12),D.B.Stewart,et al,"Design of Dynamically Reconfigurable Real-Time Software Using Port-Based Objects" P.759-776	1-23

☐ Ｃ欄の続きにも文献が列挙されている。　　☐ パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
１１．０４．００	16.05.00

国際調査機関の名称及びあて先	特許庁審査官（権限のある職員）		
日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	川崎　優　印	５Ｂ	８９４４
	電話番号　０３－３５８１－１１０１　内線　３５４５		