JEITA講座 IT最前線

CORBAからWebサービスへ

2003年1月6日

NEC インターネット基盤開発本部
佐治 信之 (saji@cd.jp.nec.com)

NEC

目次

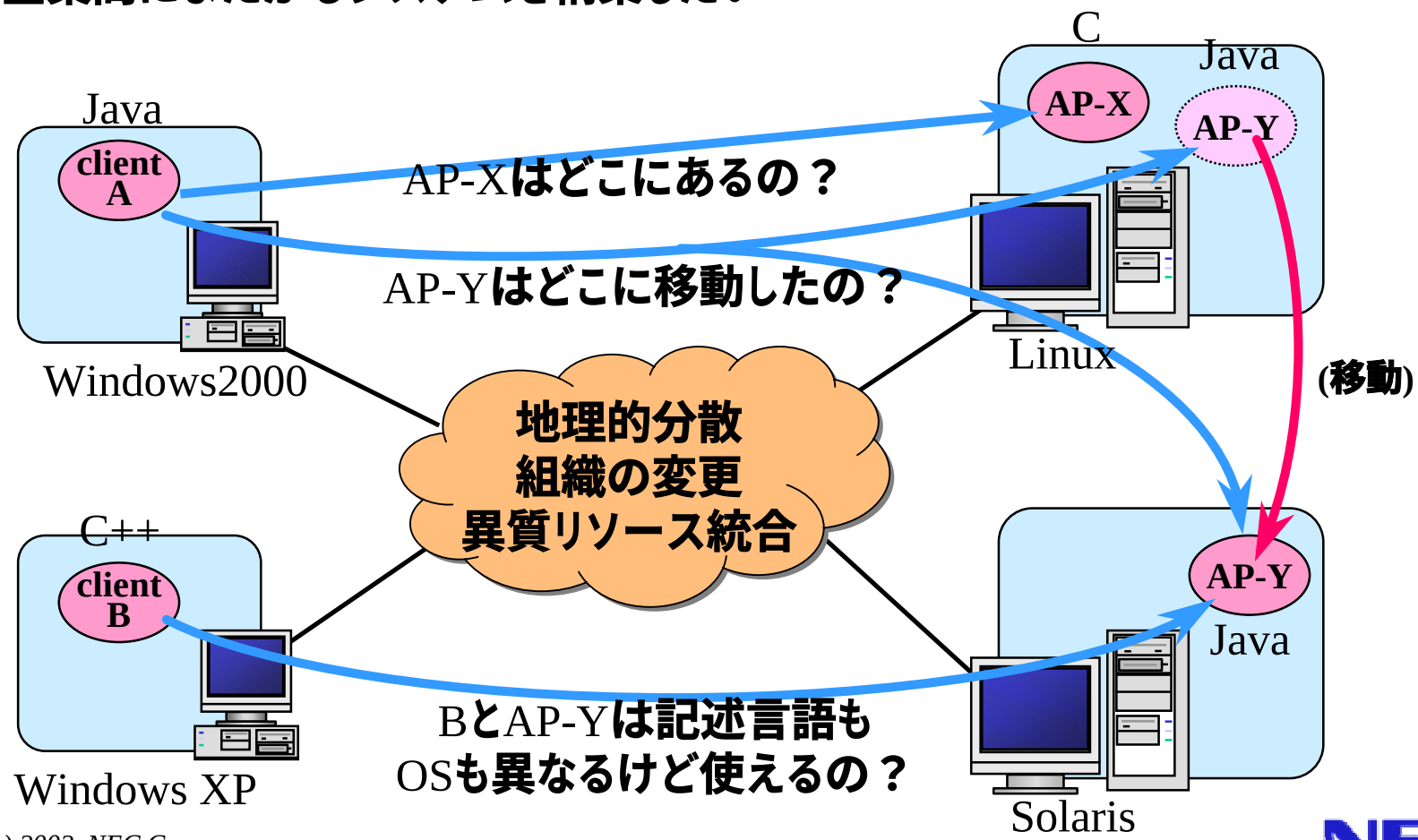
- ORB技術のご利益
- CORBA概説
- CORBA適用事例
- Webサービス技術の台頭
- 相互接続実証実験の試み
- アプリケーションサーバ製品への組込み

ORB技術のご利益

NEC

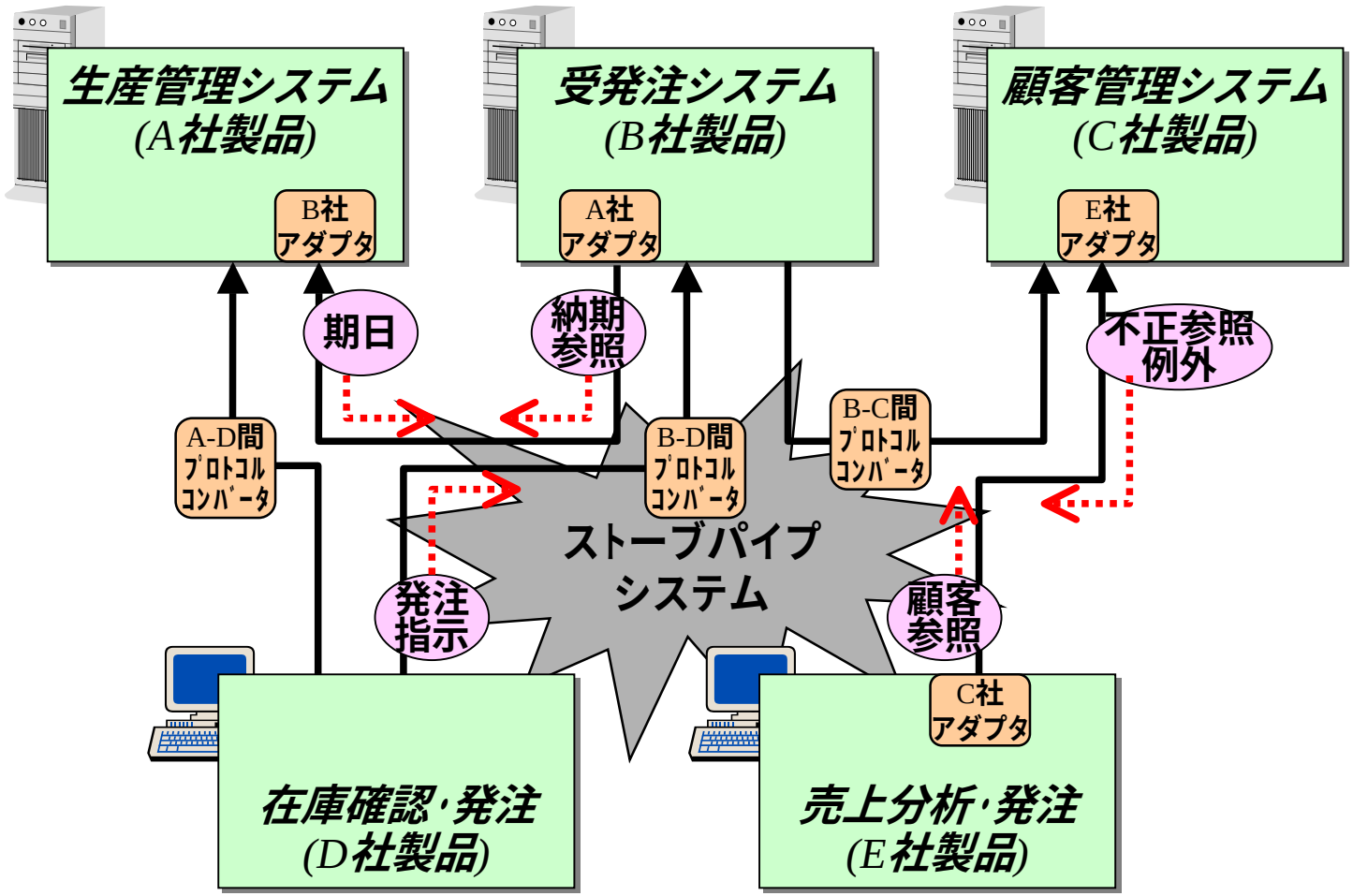
ITの分散協調化の要請

- 既存APをそのまま生かしつつ、新システムも統合できるようにしておきたい
- 世界各地の拠点のシステム間を連携させたい
- 企業間にまたがるシステムを構築したい



アプリケーション間連携の課題

マルチベンダー、マルチプラットフォーム環境でのAP相互運用



Object Request Broker技術

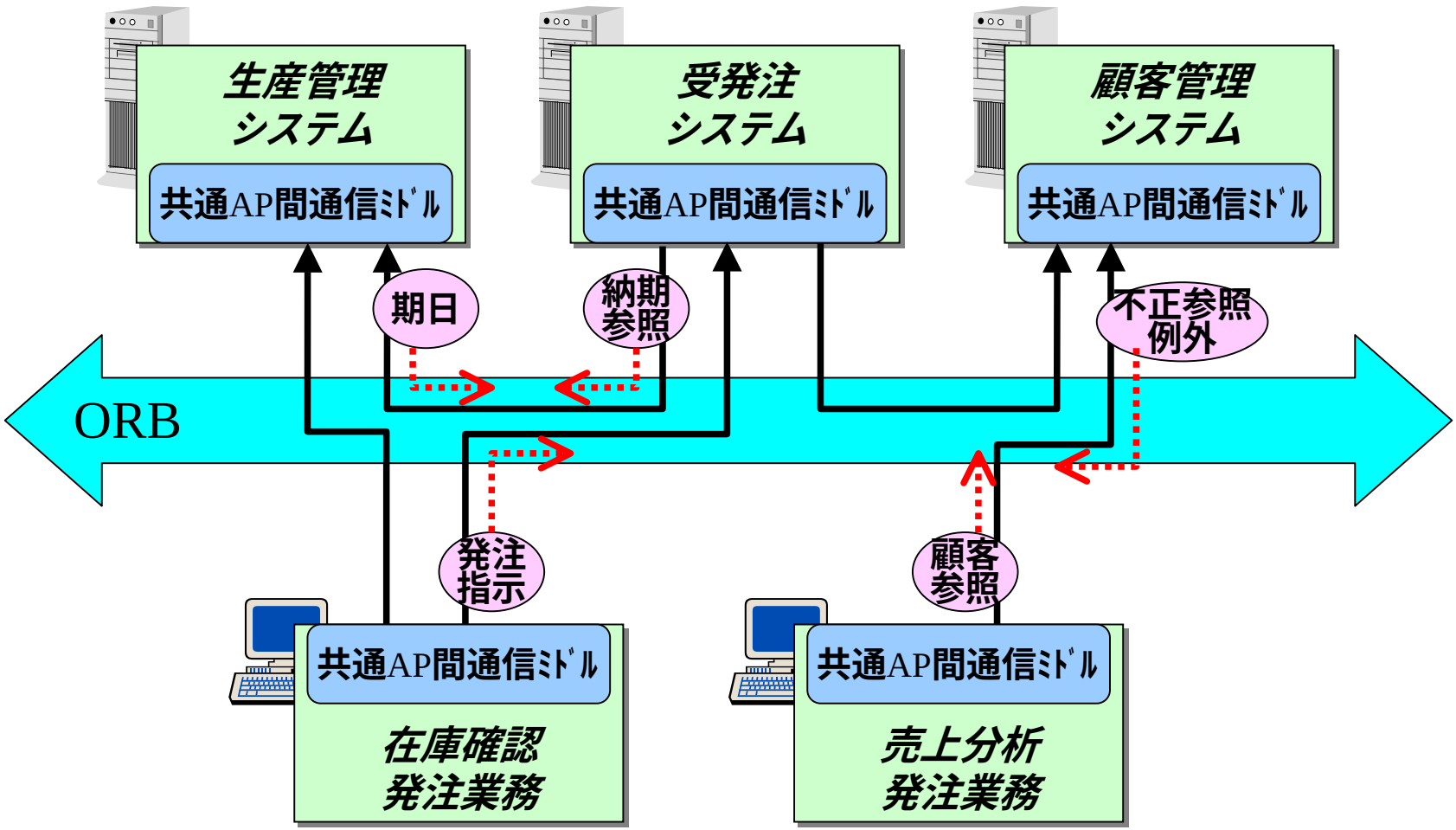
- ストーブパイプシステムではなく、共通のソフトウェアバスにオブジェクト(アプリケーションの機能単位)をプラグ＆プレイできる環境
- プラットフォーム(各種UNIX、Windows、携帯機器、組み込み機器他)に依存しない、ソフトウェアバス
- 開発言語(COBOL、C++、Java、...)に依存しない仕掛け
- 組織の変更に耐える位置情報管理(ネーミング)の仕掛け
- オブジェクトベースの機能拡張性や規模拡張性
- 世界標準の通信プロトコル

CORBA、JavaRMI、SOAPなどが解になりうる

NEC

共通ソフトウェアバス(ORB)の導入

マルチベンダー、マルチプラットフォーム環境でのAP相互運用の実現

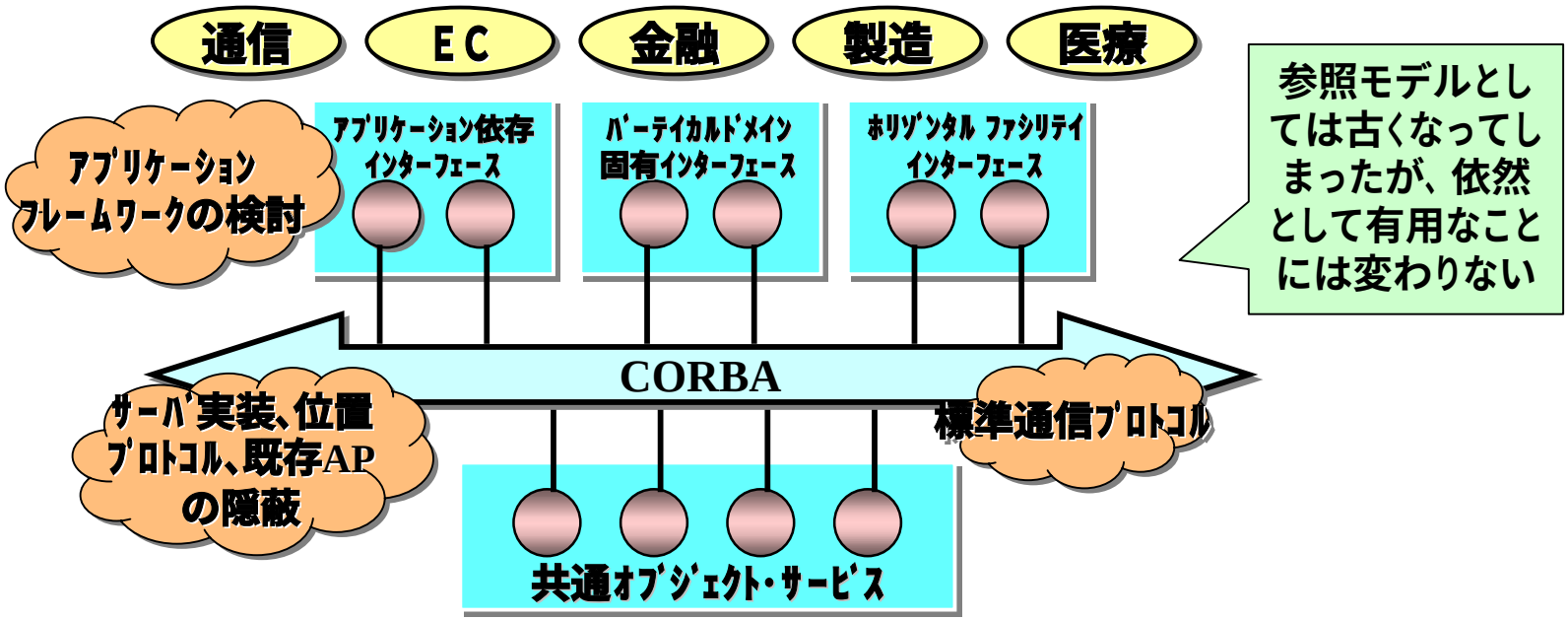


NEC

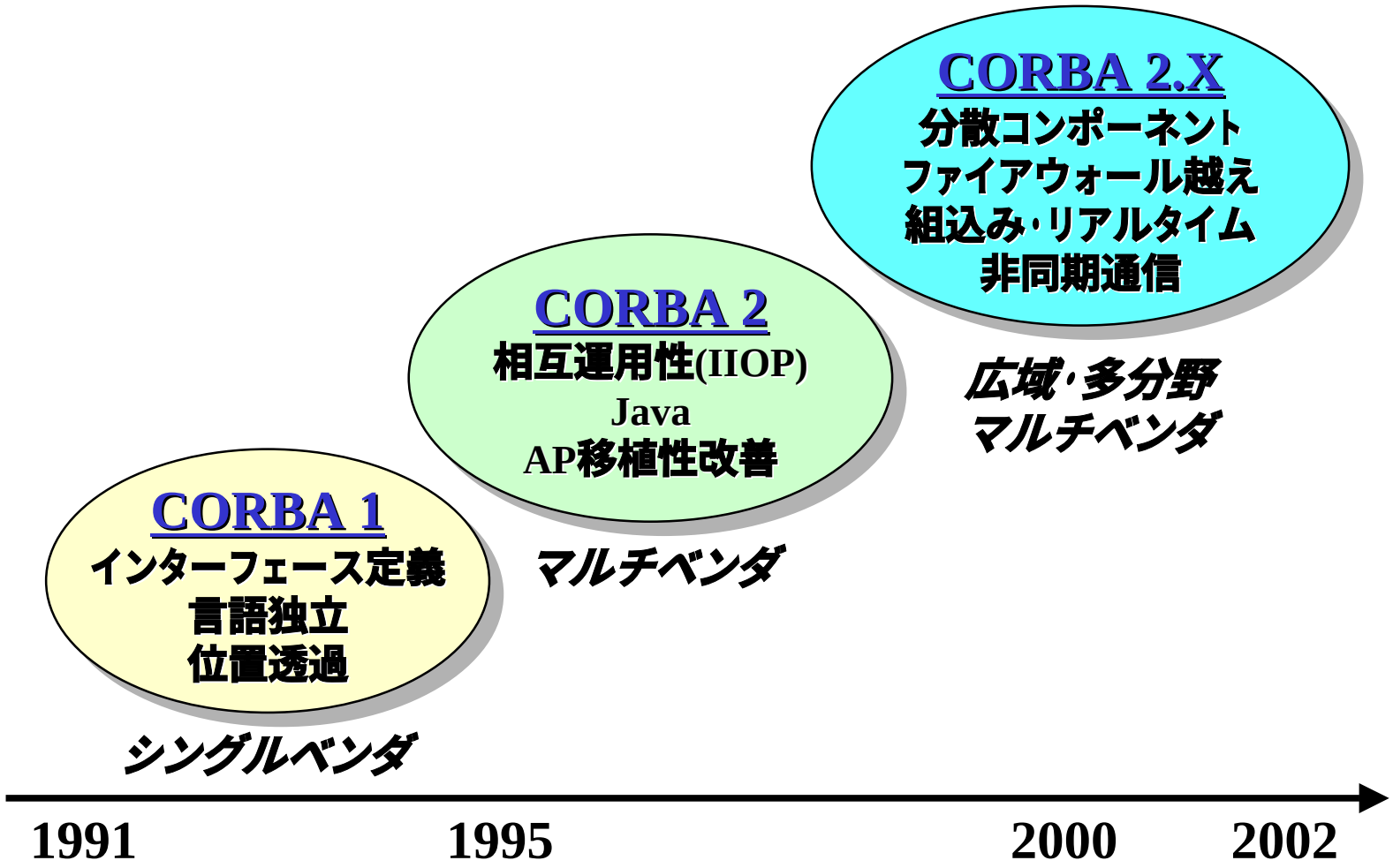
CORBA概説

OMG と CORBA

- 民間主導の標準仕様策定組織 OMG(Object Management Group、1989年設立)で制定
- Object Management Architecture (参照モデル)に基づく分散オブジェクトミドルウェア仕様
  - システムのマルチベンダ構成下で共通のインタフェース(IDL)と通信プロトコル(IIOP)を提供し、相互運用性を確保
  - 様々な処理を場所・ハードウェア・OS・開発言語の違いを超えて連携
  - 位置管理機構(ネーミングサービス)により、組織やシステムの変更に柔軟に対応
- Model Driven Architecture によるモデルの再構成化が進行中

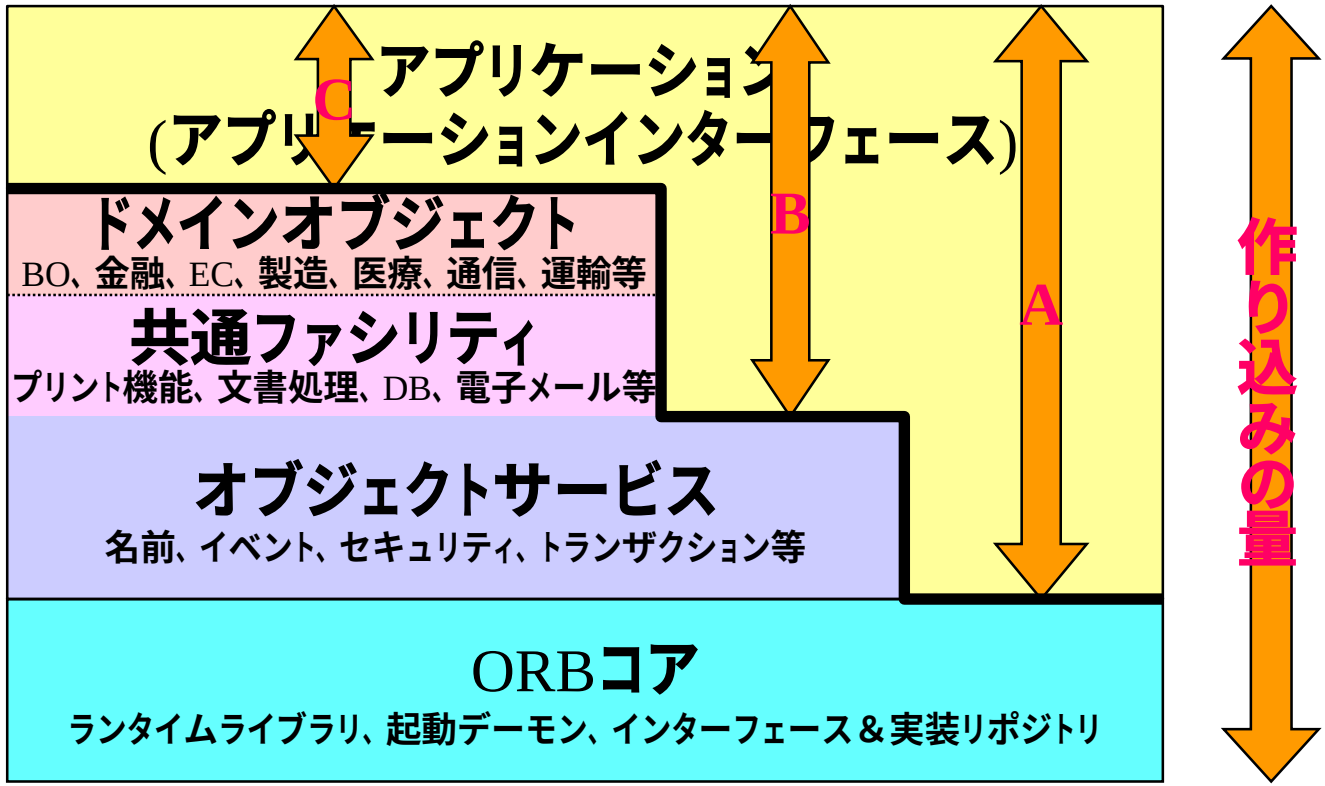


CORBA仕様の進化



NEC

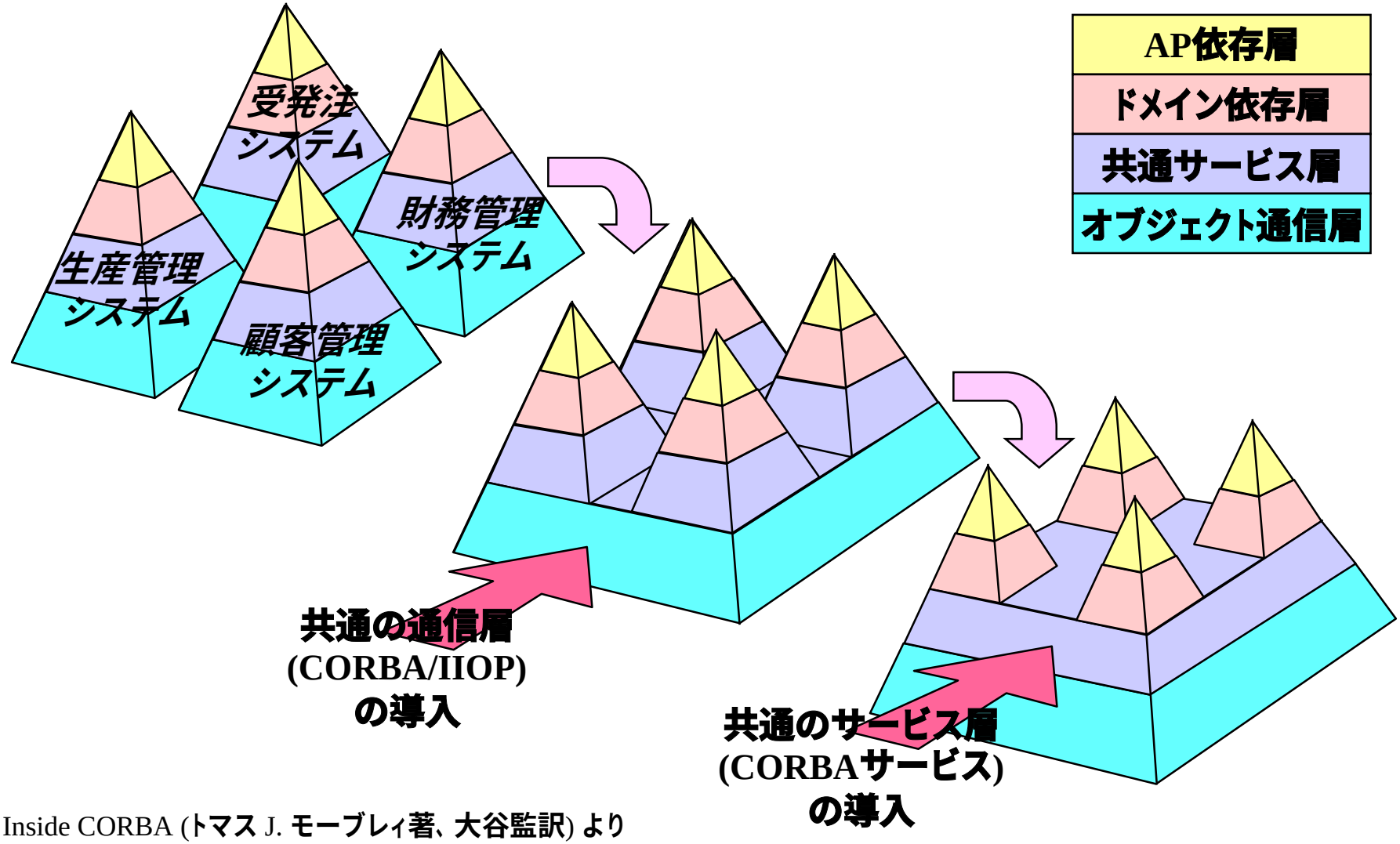
CORBAのAPアーキテクチャ



A … CORBAを通信ライブラリとしてのみ使用
B … 基本サービス(名前サービス等)を有効に活用
C … 共通ファシリティやドメインオブジェクトを駆使

NEC

CORBAベースの共通化ステップ

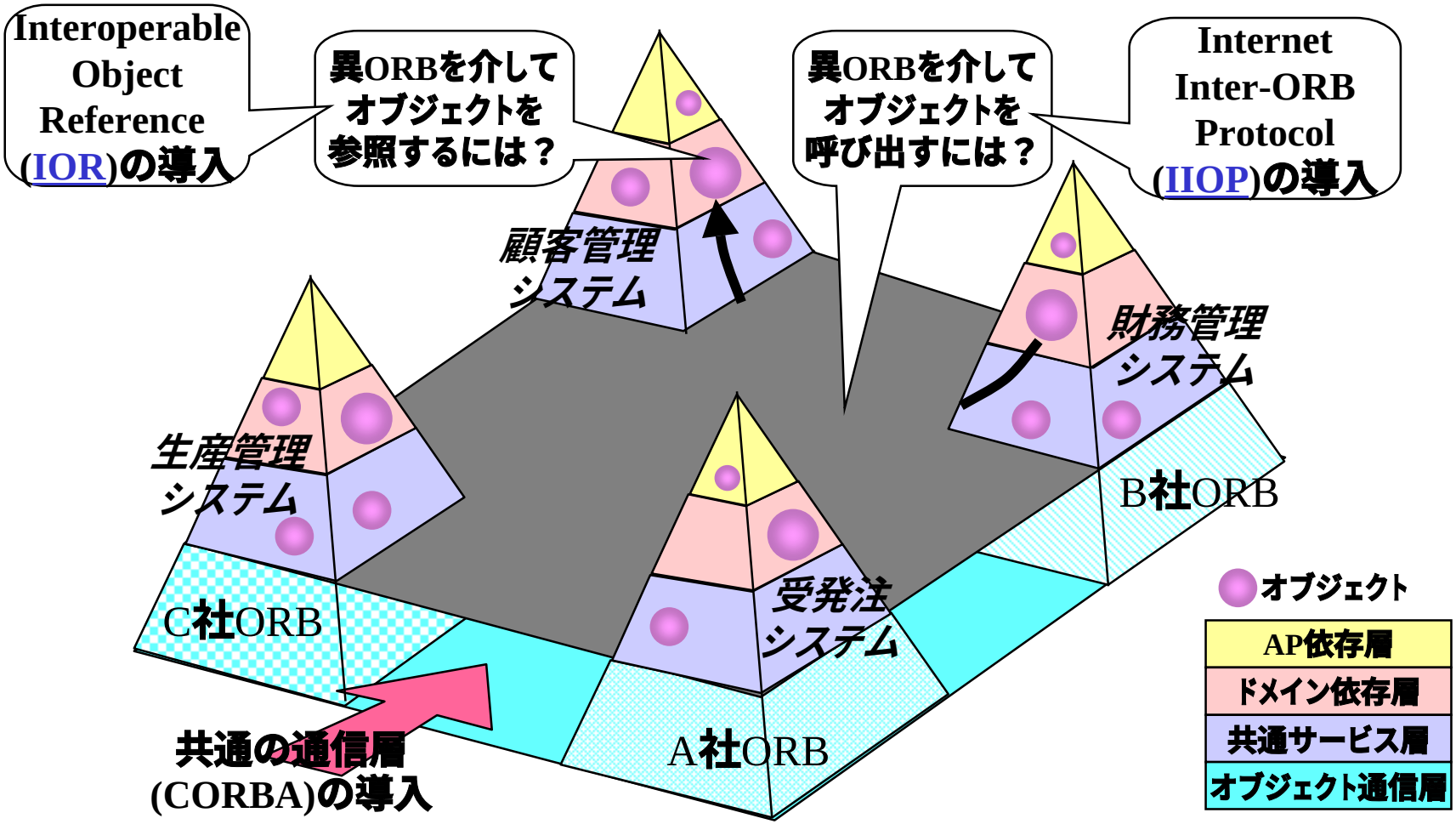


Inside CORBA (トマス J. モーブレィ著、大谷監訳) より

NEC

CORBA相互運用性とは

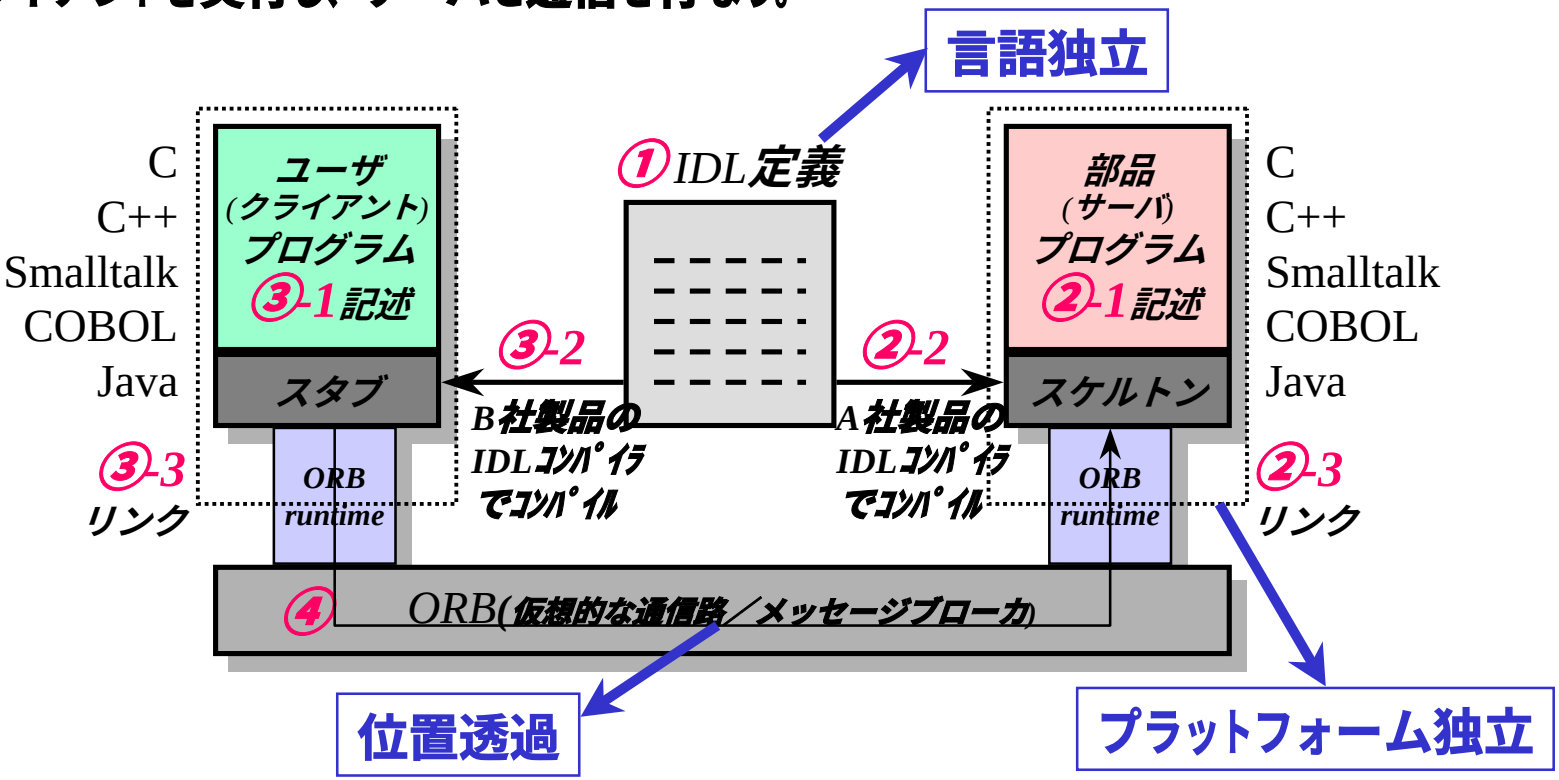
異なるORB上のAP同士の相互接続/運用を保証



NEC

CORBAプログラミング

① サーバのインタフェース仕様の定義(IDL) … 記述言語に依存しない
② IDL定義を元に、サーバのプログラムを作成。IDLコンパイラによりサーバスケルトンを生成。ORBの実行時ライブラリとこれらをリンクしてサーバ実行プログラムを作成。
③ クライアント実行プログラムもサーバと同様に作成。(異なる製品のIDLコンパイラを使用してもかまわない)
④ クライアントを実行し、サーバと通信を行なう。



IDL記述とクライアント・サーバプログラムの例

```
exception NoExist {
  string key;
};
exception AlreadyExists {
  string key;
};
interface db {
  void register_pair(in string key, in long value) raises(AlreadyExists);
  long get_value(in string key) raises(NoExist);
  void set_value(in string key, in long value) raises(NoExist);
  string get_keys(in long value);
  void clear();
  oneway void kill_server();
};
```

- IDLはオブジェクトのインタフェースのみを記述
- クライアントとオブジェクトの実装は従来のプログラミング言語で記述
- IDL記述から既存のプログラミング言語へのマッピングを規定

IDL定義

```
CORBA::Long dbImpl::get_value(const char* key)
{       // ハッシュテーブルから値を取り出す
        return gethash(key);
}
```

サーバプログラム

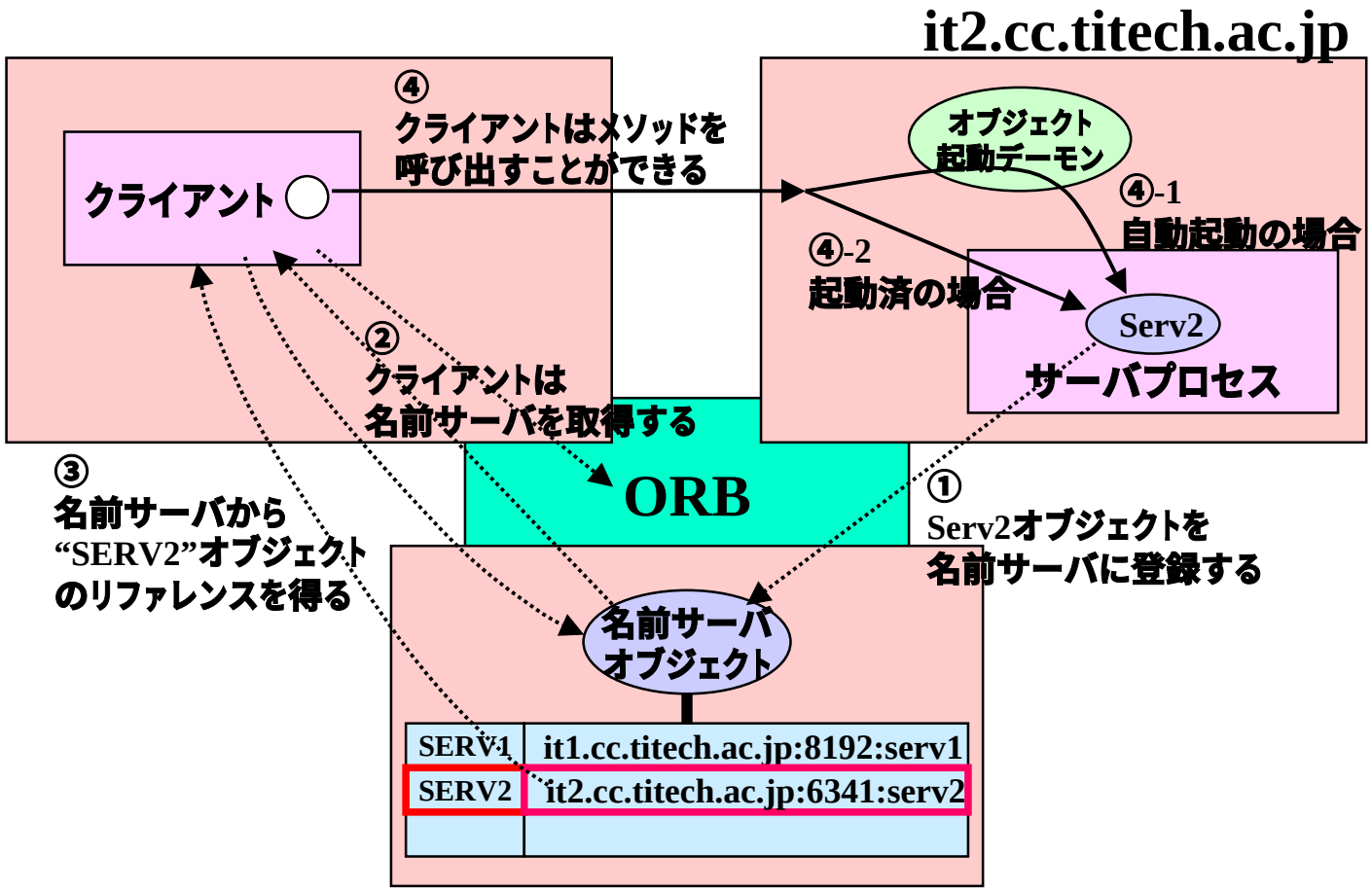
```
// ネームサーバアクセス
CORBA::Object_ptr _obj = namesvobj->resolve_str("myDB");
// スタブクラスに変換
db_ptr p_db = db::_narrow(_obj);
// 呼び出し
CORBA::Long retval = 0;
try {
  retval = p_db->get_value("KEY1");
} catch (const NoExist& e) {
}
```

クライアントプログラム

NEC

CORBAの動作

総てのサービス（オブジェクト）は常に利用可能



CORBAのメリットとデメリット

■ メリット

- 通信の相互運用性の確保(IIOP)
- システムのマルチベンダー化が可能
- マルチプラットフォーム展開が容易
- 高品質の確保
- プログラミング標準化の促進
- 将来性

■ デメリット

- 仕様化作業にはそれなりの時間がかかる
- CORBA製品はタダではない(オープンソース系もあるが)
- 通信に対するオーバーヘッドは避けられない
- 上位コンポーネント(モデル)の整備が進んでいない

CORBA適用事例

NEC

CORBA利用事例(海外)

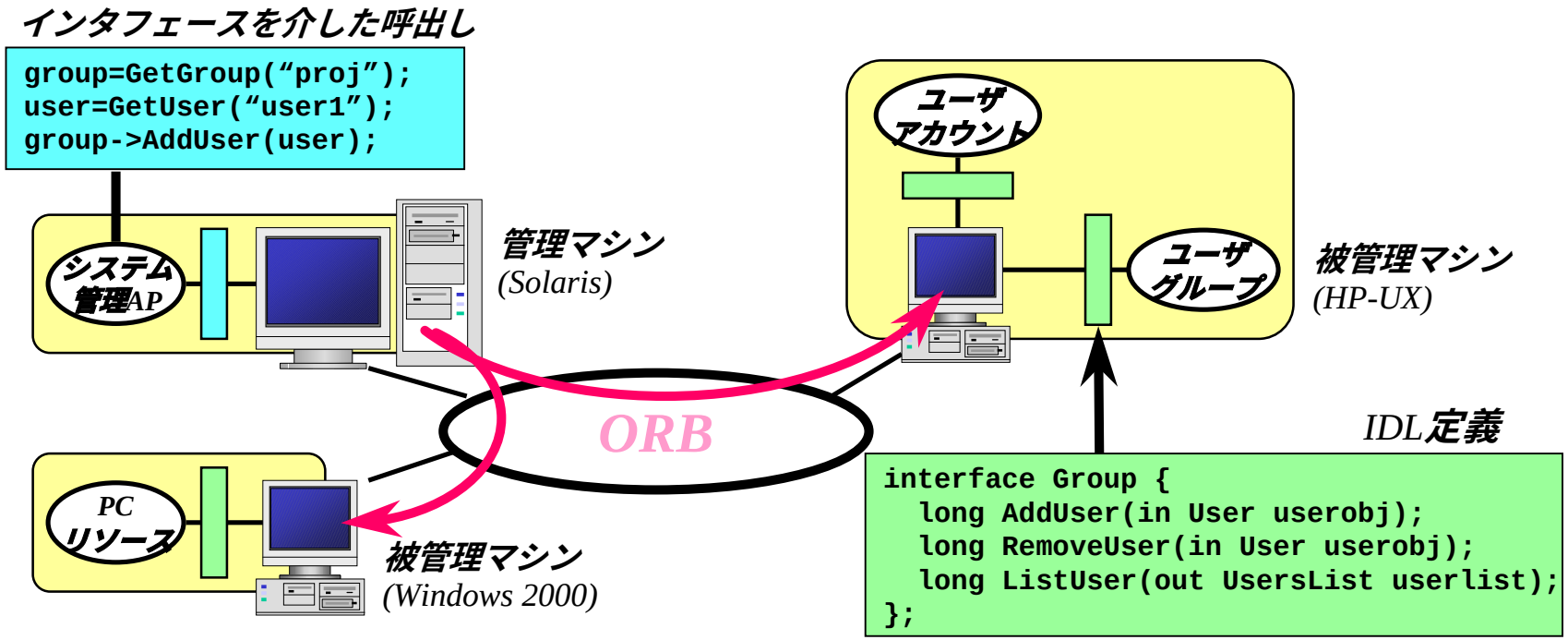
(OMG事例集より抜粋)

適用分類	システム分類	システム概要
航空/防衛	レガシー接続	NASA Goddard Space Flight Center センサーからのデータをクライアントにGUI表示するシステムをFat-Clientからmulti-tier Thin-Clientアーキテクチャにするに当ってLegacy C++を活かしながらクライアントとサーバ間の通信にCORBAを使用
	システム結合	Boeing さまざまなアプリケーションで扱われていた航空機の部品のリストをCORBAを使用して単一のビューに結合するなど。
	システム結合	Rockwell Science Center 飛行機や防衛システムなどのデザインシートを作成するアプリケーションLispで書かれていたがデザインシートデータをC++APなどと交換できる必要性がでてきた。これらAPの結合にCORBAを使用
銀行/金融	Webアプリ	Bank of America 顧客が自分のクレジットカードアカウントの情報をインタネット経由でアクセスできるシステム。バンカメではCORBAを本格的に使用した2番目のアプリケーション。1番目はCRMシステム
	Webアプリ	Nations Bank テレフォンバンキング/WebバンキングシステムのCORBAを使用
政府	システム結合	過去のある住所に於ける犯罪発生件数は、警察/消防の派遣の際に危険度を測る重要なデータであるが、このデータを管理している部門と警察/消防は別部門で紙ベースで行われていたが、CORBAを使用して結合することにより即座に利用できるようになった。
製造業	システム結合/ レガシー接続	National Semiconductor UK 既存のインベントリデータベースと結合した製造/報告システム
出版/ﾏﾙﾁﾒﾃﾞｨｱ	標準データ交換	CNN Interactive 様々なニュースソースを収集し、データを配信するのにCORBAを使用
小売	レガシー接続	NIKE 消費者窓口システムをCORBAを使用してメインフレームと結合
テレコム	その他	AT&T 規制緩和に伴う競争力強化のためのIT 強化のため、20〜40のシステムにCORBAを採用している
輸送/旅行	レガシー接続/ システム結合	American Airline CORBAを使用して既存システムを活かしながら新機能を追加。マシンを追加することにより負荷分散を行う機能などを実現
	システム結合	FedEX パーセルトラッキング、リアルタイム出荷解析
宣伝/ﾏｰｹﾃｨﾝｸﾞ	標準データ交換	Deutsche Presse Agentur マルチメディア記事配信システム

NEC

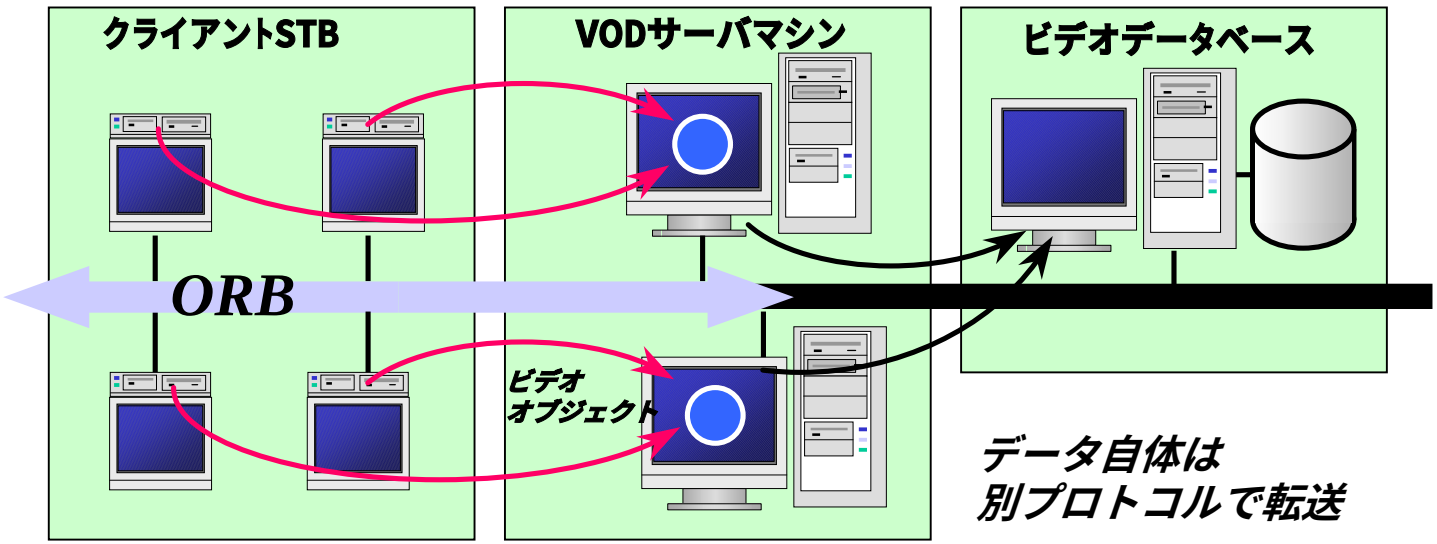
事例1：システム管理基盤

- 構成･障害･運転管理、性能管理、インストール･デリバリ管理
- バックアップ管理、ユーザ･グループ･課金管理、ジョブ管理、資源管理
- PC管理、装置管理、ストレージ管理、ネットワーク機器管理



事例2: VOD基盤

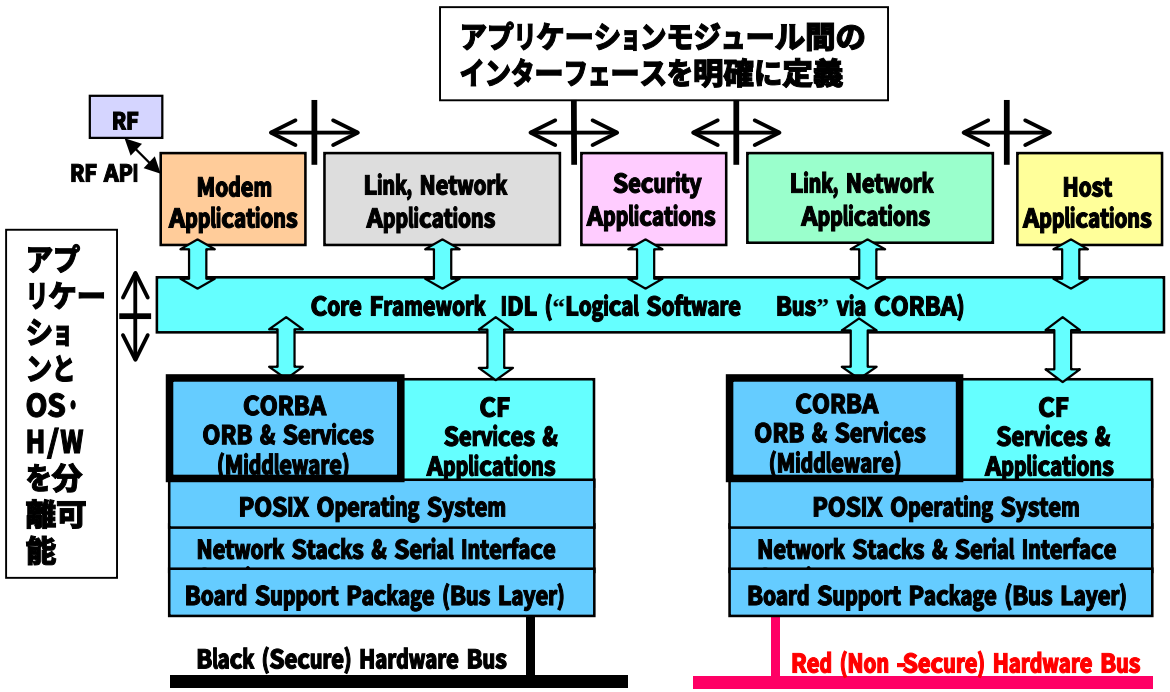
- 香港テレコム、CdS(フランス科学博物館)、その他
- DAVIC準拠仕様
  - ビデオサーバ上の制御オブジェクトに対し、
  - ビデオデータの要求、早送り、巻き戻し等の制御



事例3：ソフトウェア無線機

- ソフトウェア無線機
  - 様々な通信方式に対応して無線端末の機能を状況に応じて書き換えられる
- Software Defined Radio Forum (SDR Forum) で推進
- minimum CORBA、realtime CORBAの適用

ソフトウェア無線機のアーキテクチャ



(SDR Forum資料より転載)

CORBA仕様のサブセッティング

full CORBA
- 動的起動インタフェース
- インタフェースリポジトリ
- POAの一部の機能

minimum CORBA
- ORB間通信機能

realtime CORBA
- プライオリティ制御

NEC

Webサービス技術の台頭

技術は直に陳腐化する？
WebサービスはNirvanaになりうるか？

NEC

CORBA + Java/EJB + Webサービスへ

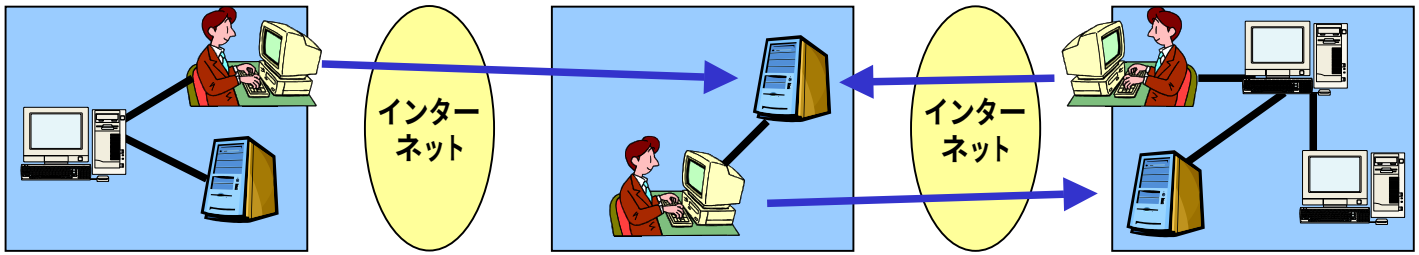
- CORBAの課題
  - トランザクション処理記述、排他制御、DBアクセス記述などの煩雑さ
  - コンポーネントモデルの未成熟さ
  - 疎結合ネットワーク(ファイアウォール)との相性
- Java/EJBの出現
  - Enterprise Java Beans (Session Bean/Entity Beanの導入)
    - コンポーネントモデルの整備
    - トランザクション処理、DBアクセス等の隠蔽
  - デプロイメントの考え方
  - RMI over IIOP の採用 (CORBAからの継承性)
- Web・XML技術の発展
  - XML、XSLT、XML Schema
  - HTTP、WebサーバからSOAP技術へ
    - 遍在するWebサーバ(TCP ポート80/HTTPプロトコル)の活用
  - WSDL：ユニバーサルインタフェース記述
    - トランスポートの束縛からの解放

NEC

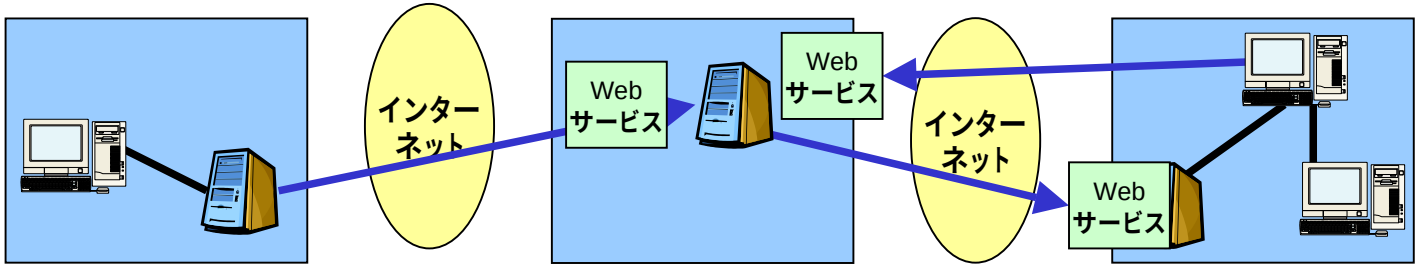
Webサービスとは

Webサービスとは、ビジネス取引の自動化を目指して進化したインターネット技術であり、ひとつのビジネスをネットワーク中に分散した複数のサービスコンポーネントの集合体として提供することを最大の特長とする。

■ **従来のWebによるビジネス取引**



■ **Webサービスによるビジネス統合**



NEC

Webサービスの基本モデル

Webサービス上の役割を、サービスの利用者、提供者、ブローカに区分して、3者間のサービスに対する関係を規定する。

[提供者⇔ブローカ]

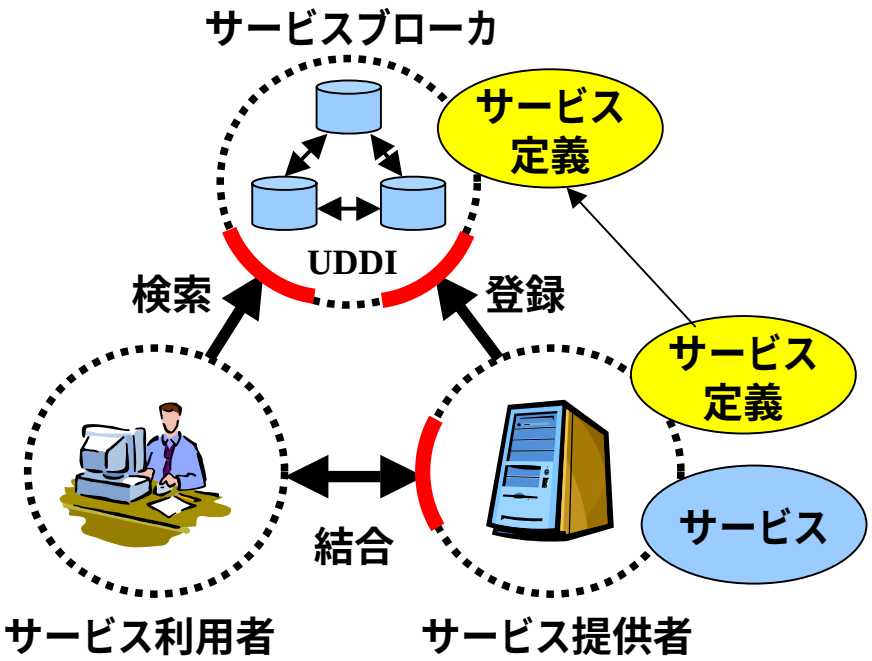
- **提供するサービスの定義を、提供者がブローカに登録する。**

[利用者⇔ブローカ]

- **利用者は、ブローカに対してサービス定義に関する検索を行う。**

[利用者⇔提供者]

- **利用者は、提供者との間で直接サービスの結合(実行)を行う。**



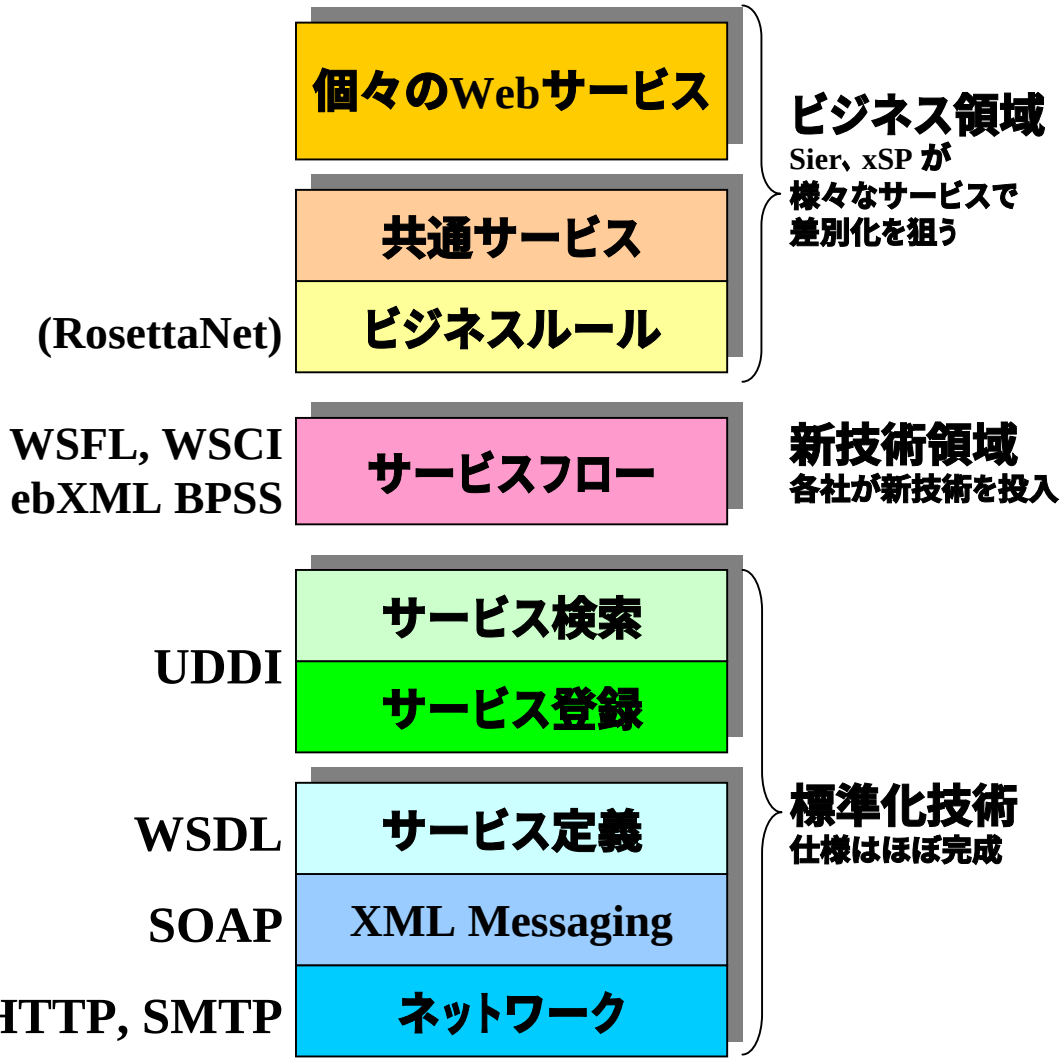
WSDL(Web Service Description Language)

SOAP(Simple Object Access Protocol)

UDDI(Universal Description, Discovery and Integration)

NEC

Webサービススタック



用語解説

- UDDI(Universal Description, Discovery and Integration)
  - Web サービスのための情報レジストリ
- WSDL(Web Services Description Language)
  - サービス呼出しのインターフェースを規定
- SOAP(Simple Object Access Protocol)
  - システム間でXML文書情報を交換するための通信規約

NEC

SOAP (Simple Object Access Protocol)

インターネットに分散したシステム間で構造化された型付きメッセージを交換するためのXMLの標準案。Envelope の中に、Header 部と Body部が定義される。

サービス呼出

SOAP Envelope

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV=
    "http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle=
    "http://schemas.xmlsoap.org/soap/encoding/">
```

SOAP Header

```
<SOAP-ENV:Header>
</SOAP-ENV:Header>
```

SOAP Body

```
<SOAP-ENV:Body>
   <m:GetLastTradePrice
         xmlns:m="Some-URI">
      <m:symbol>DIS</m:symbol>
   </m:GetLastTradePrice>
</SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>
```

サービス返り値

SOAP Envelope

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV=
    "http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-ENV:encodingStyle=
    "http://schemas.xmlsoap.org/soap/encoding/">
```

SOAP Header

```
<SOAP-ENV:Header>
</SOAP-ENV:Header>
```

SOAP Body

```
<SOAP-ENV:Body>
   <m:GetLastTradePriceResponse
         xmlns:m="Some-URI">
      <m:price>34.5</m:price>
   </m:GetLastTradePriceResponse>
</SOAP-ENV:Body>
```

```
</SOAP-ENV:Envelope>
```


NEC

WSDL (Web Services Description Language)

WSDLは、サービスのエンドポイント間で交わされる手続き呼出のインタフェースとプロトコルバインディングを規定するための XML フォーマット。

```
<?xml version="1.0"?>
 <definitions
   name="StockQuote“
   targetNamespace="http://example.com/stockquote.wsdl"
   xmlns:tns="http://example.com/stockquote.wsdl"
   xmlns:xsd1="http://example.com/stockquote.xsd"
   xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns="http://schemas.xmlsoap.org/wsdl/">

     <types>
       <schema targetNamespace="http://example.com/stockquote.xsd"
             xmlns="http://www.w3.org/1999/XMLSchema">
         <element name="TradePriceRequest">
           <complexType>
             <all>
                 <element name="symbol" type="string"/>
             </all>
           </complexType>
         </element>
         <element name="TradePriceResult">
           <complexType>
             <all>
                 <element name="price" type="float"/>
             </all>
           </complexType>
         </element>
       </schema>
     </types>
    ...
```

データ型の定義

```
<message name="GetLastTradePriceInput">
   <part name="GetLastTradePrice" element="xsd1:TradePri
</message>

<message name="GetLastTradePriceOutput">
   <part name="GetLastTradePriceResult" element="xsd1:TradePriceResult"/>
</message>

<portType name="StockQuotePortType">
   <operation name="GetLastTradePrice">
     <input message="tns:GetLastTradePriceInput"/>
     <output message="tns:GetLastTradePriceOutput"/>
   </operation>
</portType>

<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
   <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
   <operation name="GetLastTradePrice">
     <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
     <input>
       <soap:body use="literal" namespace="http://example.com/stockquote.xsd"
             encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
     </input>
     <output>
       <soap:body use="literal" names ce="http://example.com/stockquote.xsd"
             encodingStyle="http://sch .xmlsoap.org/soap/encoding/"/>
     </output>
   </operation>
</binding>
```

メッセージの定義

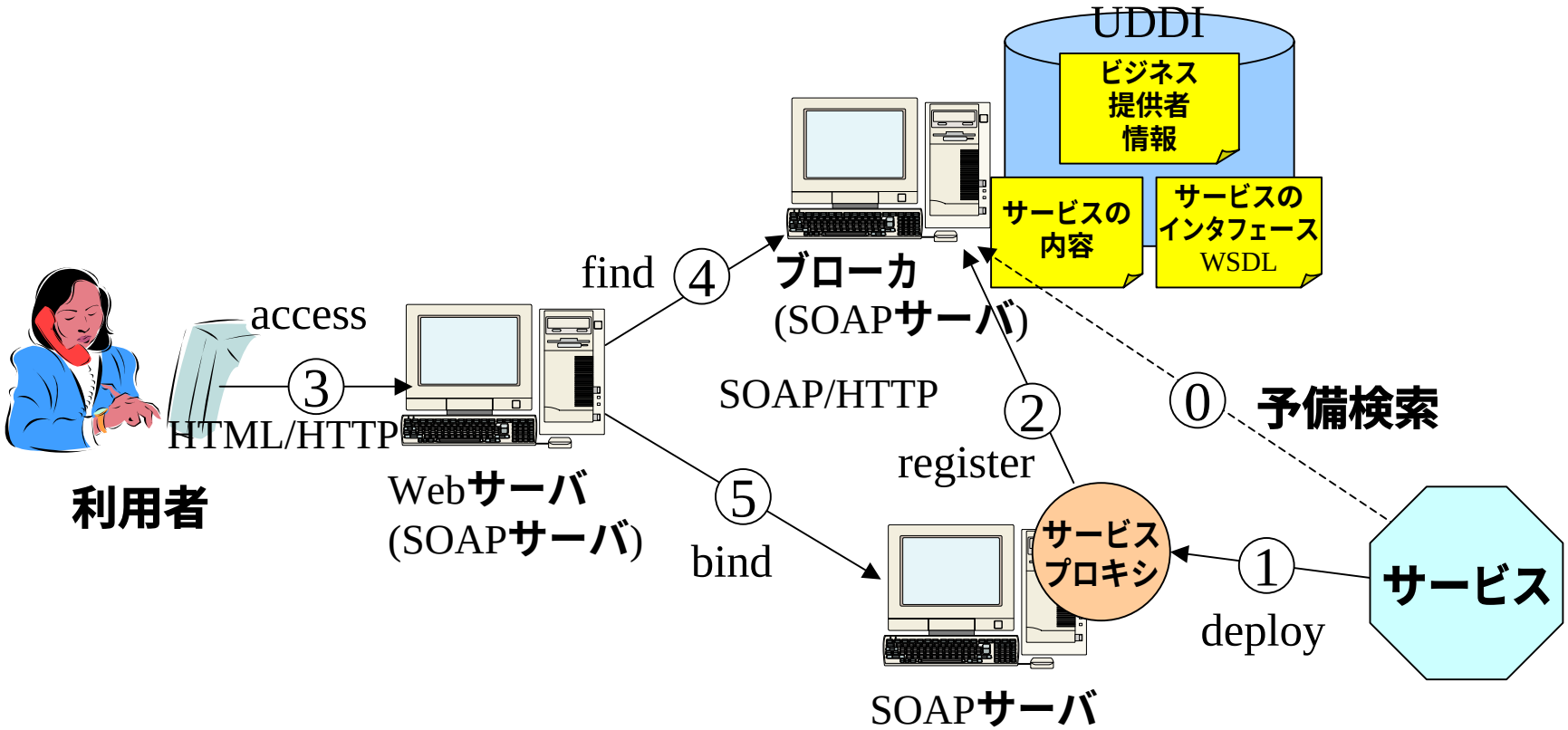
操作の定義

プロトコルバインディング

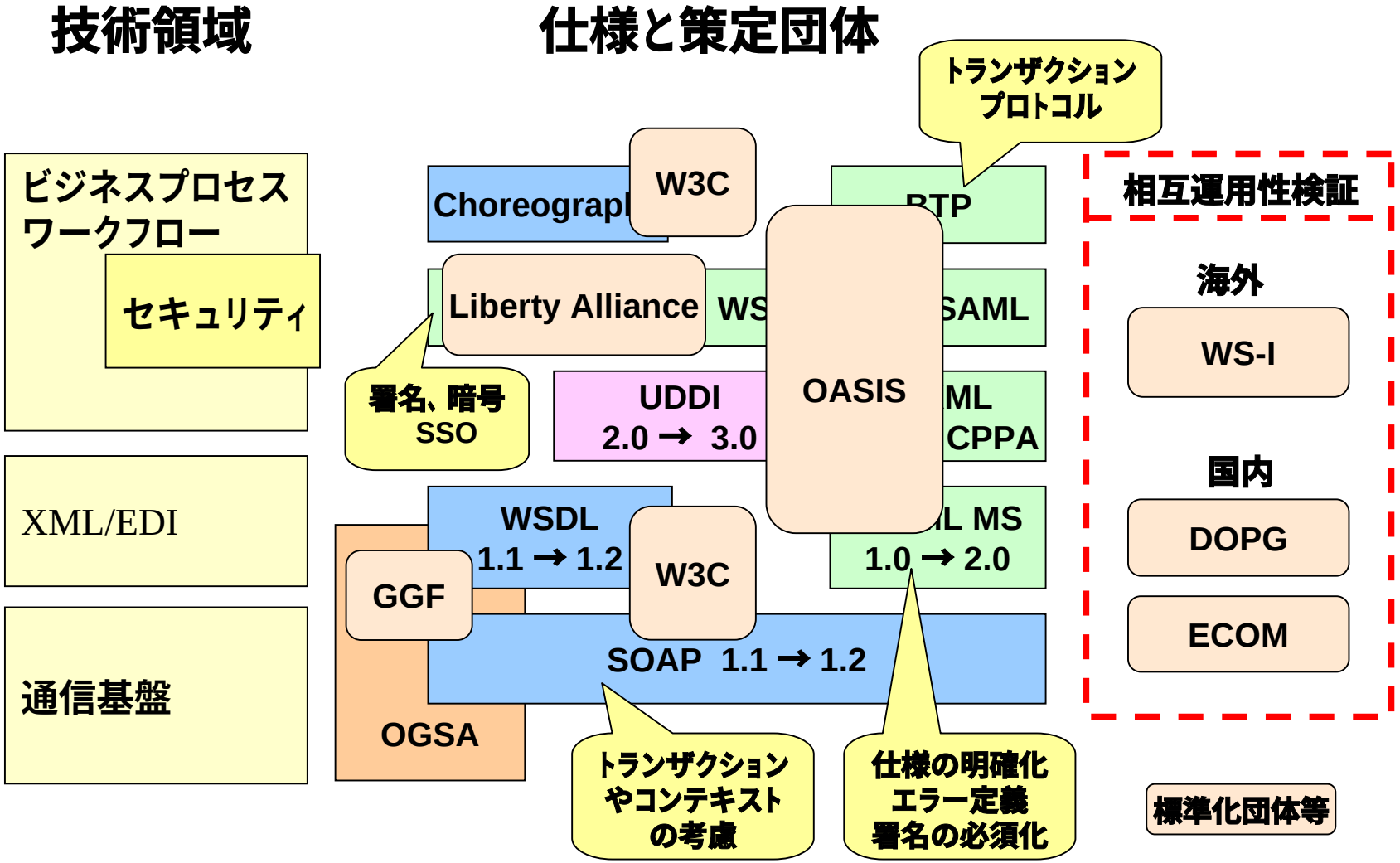
NEC

UDDI (Universal Description, Discovery and Integration)

- UDDIは、SOAPサーバを連携させたWebサービス提供アーキテクチャにおけるサービスディレクトリを提供
- ビジネス/サービス/接続方法に関する文書を登録し、利用者からに検索機能を提供
  - 登録、検索のインタフェースもSOAPから利用できる。

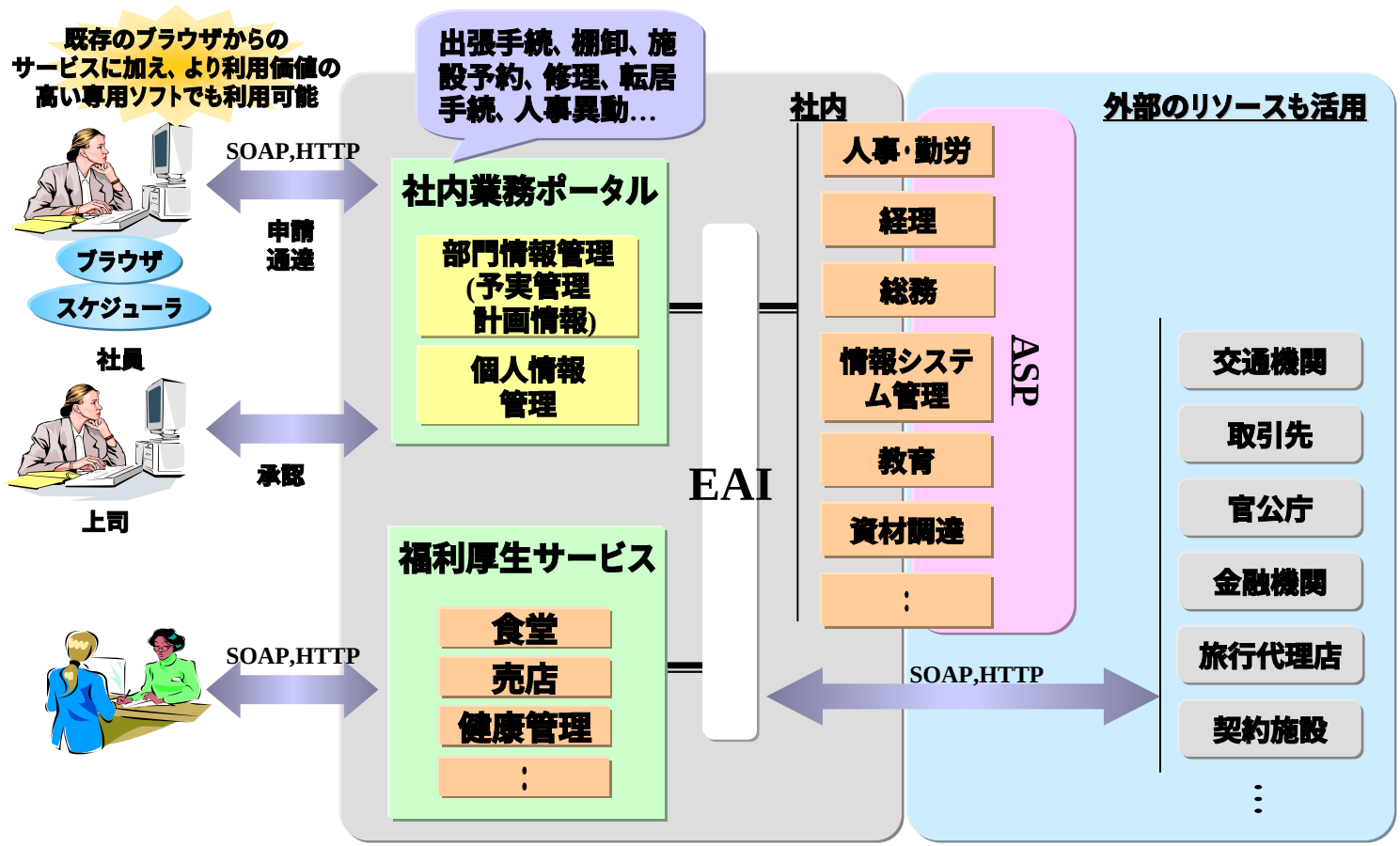


Webサービス関連標準規格と団体



Webサービス導入例

■ 外部リソースを含むプロセスを統合し、社内業務がワンストップで行えるポータルサービス

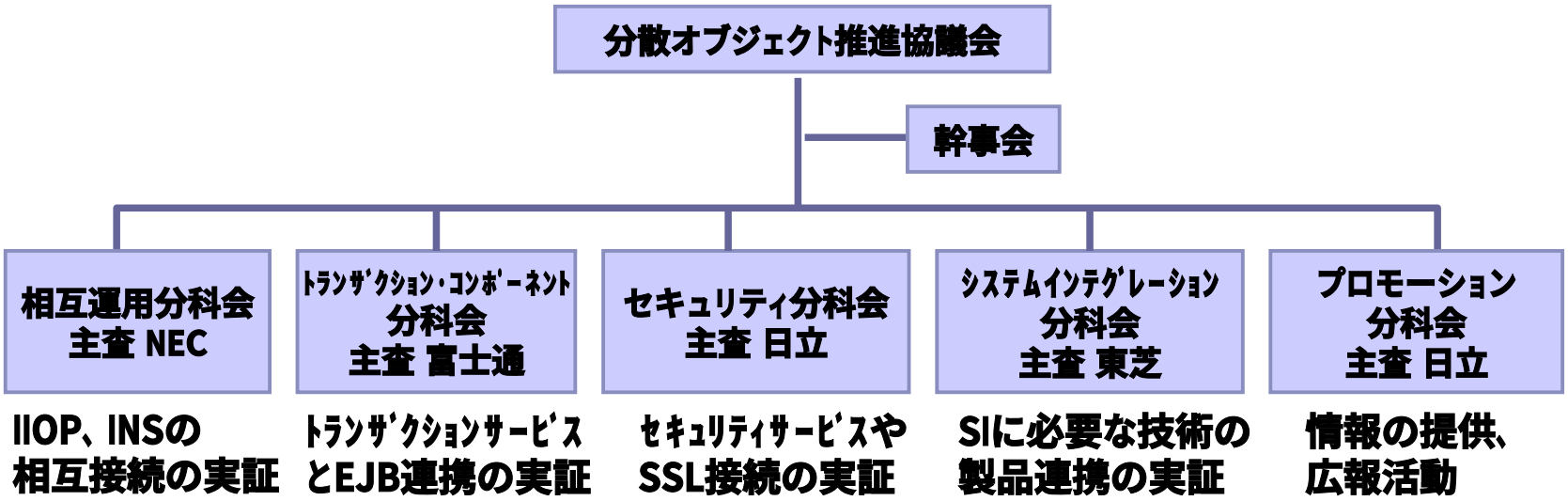


相互接続実証実験の試み

IIOPやSOAP製品の相互運用性を
実証する試みについて
(ソフトウェア製品の相性？)

分散オブジェクト推進協議会
(The Distributed Object Promotion Group)

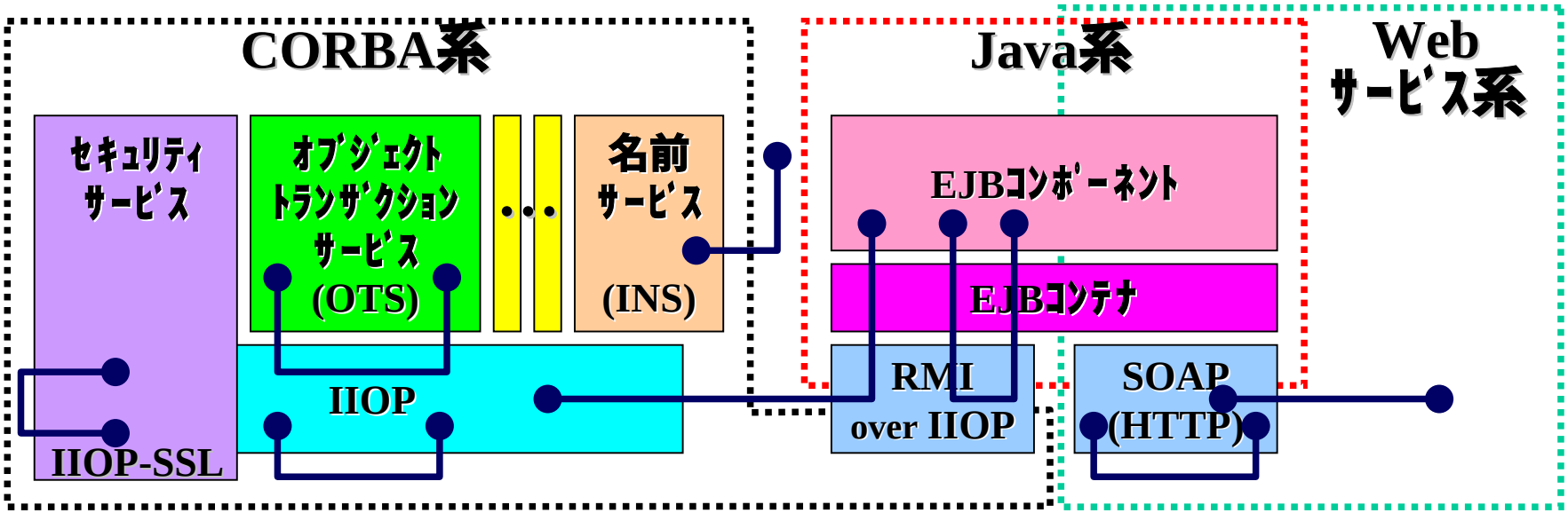
- 日本市場における分散オブジェクト技術の発展と普及を図るため、分散オブジェクトの普及、啓蒙と関連製品の相互接続性の実証を目的とする非営利団体。
- 1997年10月設立。
- 会員企業16社：沖電気、日本HP、サイベース、サン、東芝、NEC、日本IBM、日本アイオナ、日本オラクル、日本Tmax、日本BEA、日本ユニシス、日立、富士通、ボーランド、三菱電機



DOPG資料より転載

相互運用のシステムモデル

- CORBAの基本通信とオブジェクトサービス
- 様々な通信基盤同士の相互運用
  - CORBA、Java EJB、Webサービス
- 様々な通信基盤を強化･補完する機能
  - セキュリティ



DOPG資料より転載

DOPGの活動状況

IIOP
相互接続実証

▲9社間　▲10社間

Interoperable
Naming
Service
(相互接続実証実験)

▲3社間

COST
プロジェクト
テストキット
公開

SOAP
データ網羅性
(相互接続実証実験)

▲11社間　▲14社間

Security Service
(相互接続実証実験)

▲4社間IIOP-SSL

トランザクション相互接続実証

▲4社間
(1フェーズ)　▲4社間
(2フェーズ)

▲5社間
(Java/2フェーズ)

SOAP
アプリケーション間
連携接続
(相互接続実証実験)

▲12社間　▲14社間

コンポーネント
(CORBA/EJB相互接続実証実験)

▲5社間　▲7社間　▲8社間

DOPG資料より転載

NEC

IIOP相互接続実験結果

◎接続成功

(1999/5/18)

製品名	クライアント		Java 2	Orbix		WebOTX		OAS 4.0	SYSTEM V [nju:]	TPBroker		TIB/ ObjectBus	INTERSTAGE	
サーバ	社名		Sun	TIS/NEC		NEC		Oracle	ユニシス	日立		TIBCO	富士通	
		言語	Java	C++	Java	C++	Java	Java	C++	C++	Java	Java	C++	Java
Java 2	Sun	Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
Orbix	TIS/ NEC	C++	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
		Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
WebOTX	NEC	C++	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
		Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
WebSphere Application Server	IBM	Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	TIBCO社 第7回不参加 (TBD)	◎	◎
Oracle Application Server 4.0	Oracle	Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
Oracle8*i*		Java	◎	TBD	◎	TBD	◎	◎	TBD	TBD	◎	◎	TBD	◎
BEA WebLogic Enterprise	BEA	C++	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
SYSTEM V [nju:]	ユニシス	C++	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
TPBroker	日立	C++	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
		Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
TIB/ObjectBus	TIBCO	Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
INTERSTAGE	富士通	C++	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎
		Java	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎

DOPG資料より転載

NEC

SOAP相互接続実験結果

◎: 接続成功 (2002年9月25日現在)　　(2002年4月実験済)　　2002年8月～　　2002年9月～

製品名	クライアント	WebLogic Server	WebSphere	WebOTX	Orbix E2A XMLBus	Oracle9*i*AS Release2	JEUS	Cosminexus	Interstage	NonStop SOAP for Java	J2EE SDK1.3 +WSDP	Sun ONE AS7	EAServer
サーバ	社名	沖・BEA・ユニシス	東芝・IBM	NEC	アイオナ	オラクル	TmaxSoft	日立	富士通	コンパック	サン	サン	サイベース
WebLogic Server	沖・BEA・ユニシス	◎	◎	◎	◎	◎	◎	◎	◎	※	※	※	※
WebSphere	東芝・IBM	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	※	※
WebOTX	NEC	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	※
Orbix E2A XMLBus	アイオナ	◎	◎	◎	◎	◎	◎	◎	◎	◎	※	※	※
Oracle9*i*AS Release2	オラクル	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	※
JEUS	TmaxSoft	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	※
Cosminexus	日立	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	※
Interstage	富士通	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	◎	※
NonStop SOAP for Java	コンパック	※	◎	◎	◎	◎	◎	◎	◎	◎	◎	※	※
J2EE SDK 1.3 + WSDP	サン	※	◎	◎	※	◎	◎	◎	◎	◎	◎	◎	※
Sun ONE AS7	サン	※	※	◎	※	◎	◎	◎	◎	※	◎	◎	※
EAServer	サイベース	※	※	※	※	※	※	※	※	※	※	※	※

※未実施(今後検証予定)

DOPG資料より転載

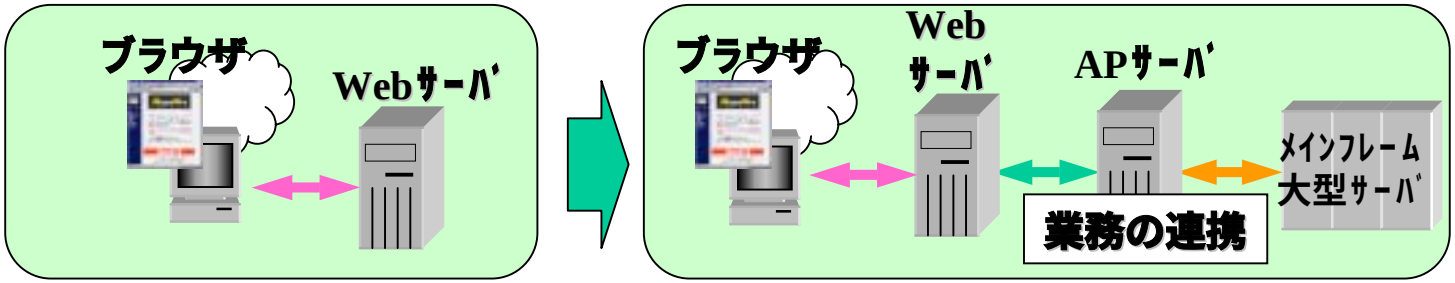
NEC

アプリケーションサーバ製品への組込み

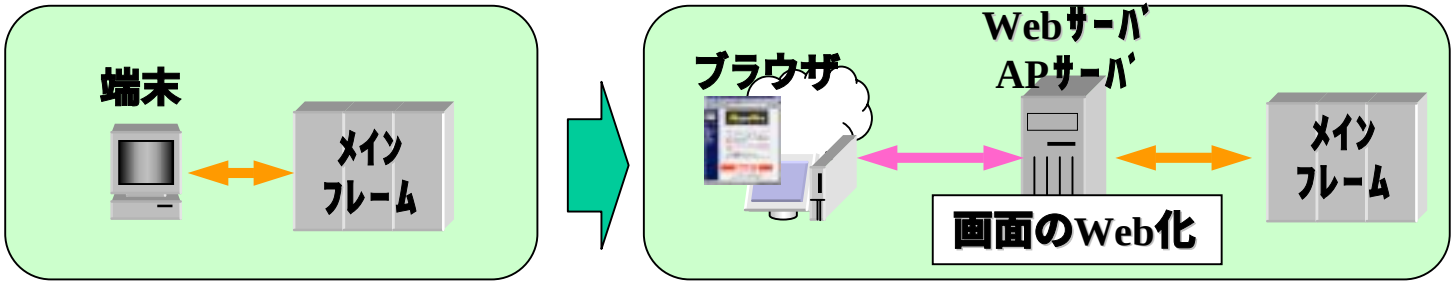
CORBA、Java EJB、SOAP、…

アプリケーションサーバの適用パターン例

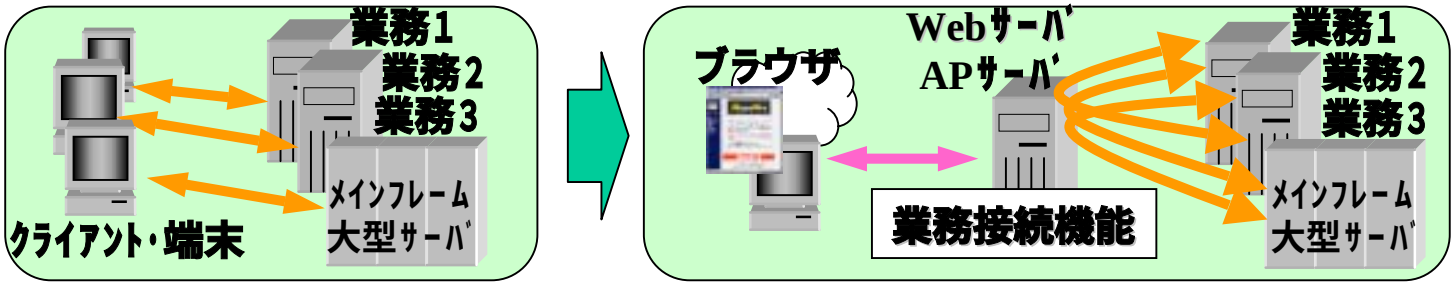
情報発信と業務の連携



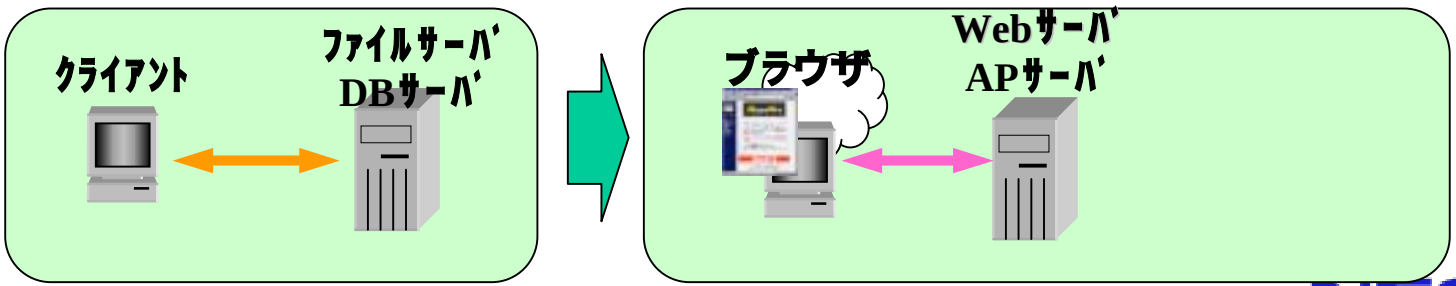
既存業務画面Web化・thin化



業務の統合・連携



クライアント サーバ 相当



NEC

WebOTX Application Server 機能概要

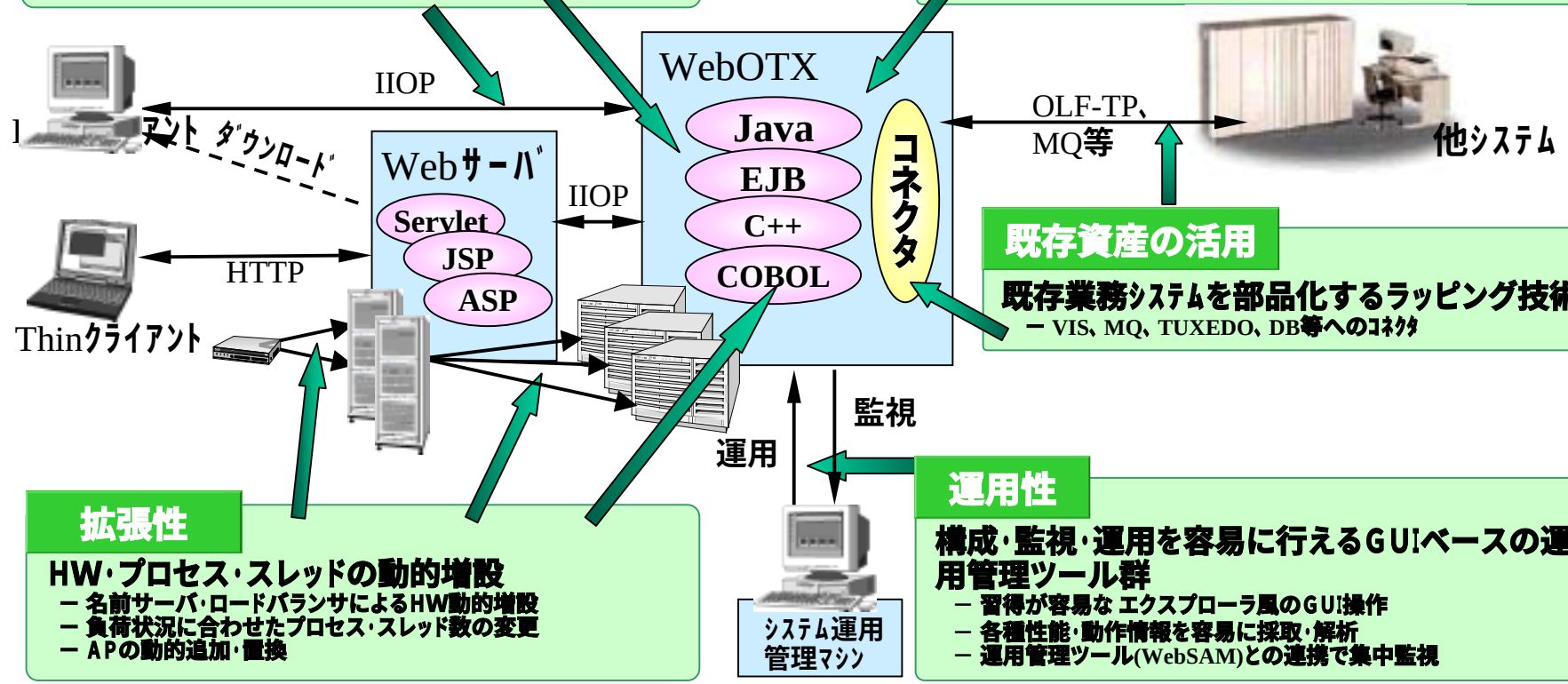
高性能・高信頼 | **高い接続性** | **最新の標準技術**

オープン性

デファクト技術に準拠した分散オブジェクトトランザクション基盤
- 実証された相互接続性 (最新CORBA仕様準拠)
- 実証された移植性 (J2EE準拠、CORBAポータビリティ準拠)
- 多様なビジネスロジック記述言語(Java,C++,COBOL)

ロバスト性

基幹業務に耐える可用性、信頼性、性能
- HW装置の冗長構成 (二重化、クラスタ)、停止時間の短縮
- 各種障害の局所化・閉塞、自動リトライ
- 世界最高レベルの通信性能、資源事前割当による高速化



既存資産の活用

既存業務システムを部品化するラッピング技術
- VIS、MQ、TUXEDO、DB等へのコネクタ

拡張性

HW・プロセス・スレッドの動的増設
- 名前サーバ・ロードバランサによるHW動的増設
- 負荷状況に合わせたプロセス・スレッド数の変更
- APの動的追加・置換

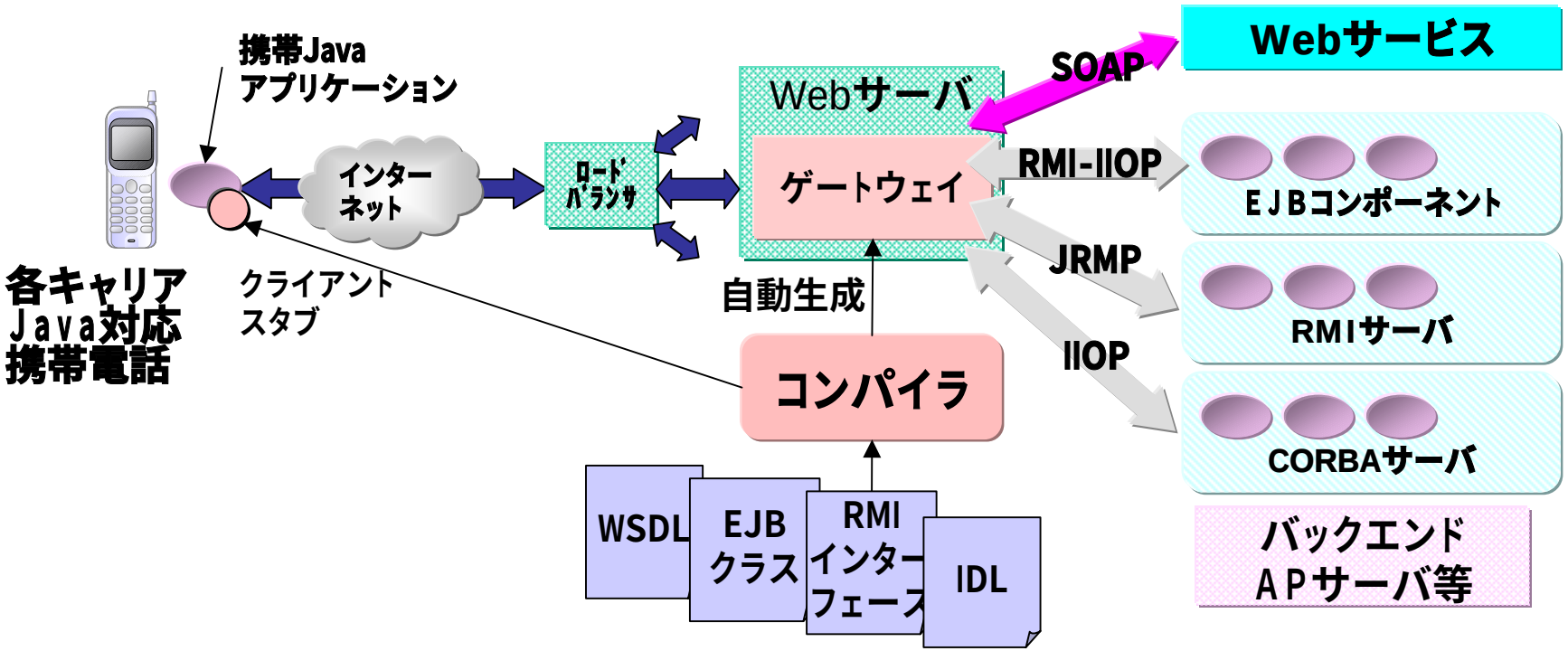
運用性

構成・監視・運用を容易に行えるGUIベースの運用管理ツール群
- 習得が容易な エクスプローラ風のGUI操作
- 各種性能・動作情報を容易に採取・解析
- 運用管理ツール(WebSAM)との連携で集中監視

NEC

Java対応携帯電話向けSOAP技術

- 携帯電話からバックエンドへ様々なプロトコルでアクセス可能
  - 重い標準プロトコルの処理をサーバ側で行い、クライアントを軽量化
  - ゲートウェイによりさまざまな標準プロトコルに変換可能
  - SSLの採用等による、セキュリティの確保



標準準拠性と差異化のバランス

■ 標準準拠性

- 標準プロトコルの採用と相互運用性の確保
- 周辺コンポーネントとの接続性
  - 特に最近はOSS(Open Source Software)の活用がキー
- 次の標準化すべき技術は何か？

■ 差異化ポイント

- 性能
- 高可用機能、耐障害性
- セキュリティ
- 開発ツール・運用ツール
- ビジネスコンサルテーション
- ビジネステンプレート
- SI・サポート
- 運用・トラブル対応体制

顧客満足度の指標例
(日経コンピュータ2002アンケート項目より)

■製品に関する満足度
- 導入や初期設定の容易さ
- 性能
- 機能
- 信頼性
- アプリケーション開発の生産性
- 運用管理の容易さ
- 価格
- 製品の将来性

■サポートに関する満足度
- バグ対応の迅速さ
- 旧バージョンのサポート期間
- 問い合せの対応
- 保守サービスの料金