

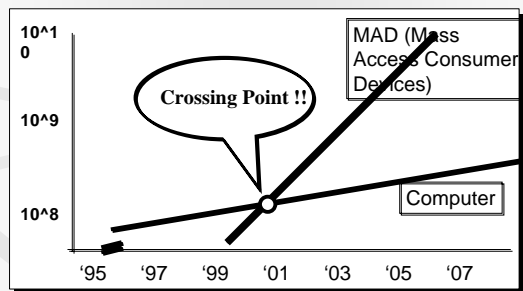
WebサービスとGrid

稚内北星学園大学
丸山不二夫

我々は今どこに立っているか

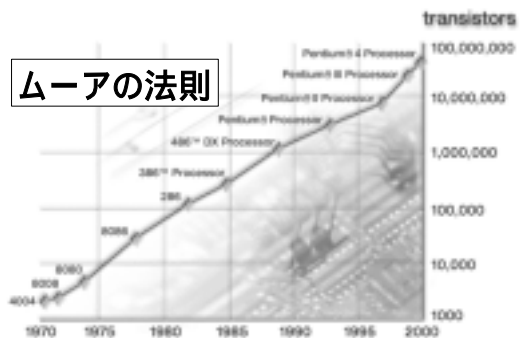
--- いくつかのcrossing point ---

ネットワークデバイスの急成長 --- ユビキタス・ネットワーク ---

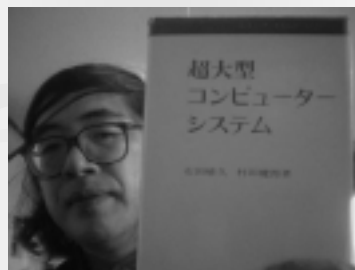


<http://www.intel.com/research/silicon/mooreslaw.htm>

ムーアの法則



1975年の 「超大型コンピュータ・システム」



東大大型計算機センターのHITAC8800/8700

1975年の 「超大型コンピュータ・システム」

- 主メモリー 4MB
- スワップ用ドラム 16MB
- ファイル用ドラム 8MB
- 集団ディスク 2.13GB
- 10プロセッサ 8MB/sec

スーパー
コンピュータ

Cray-1

1976年



Clock period
12.5 nanosecond (= 80MHz)

Address space
4M words(4x8 M Byte)

Memory Size
最大 1M words(1x8 M Byte)

Weight of mainframe
5.25 tons

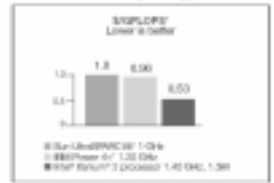
Cooling
Freon



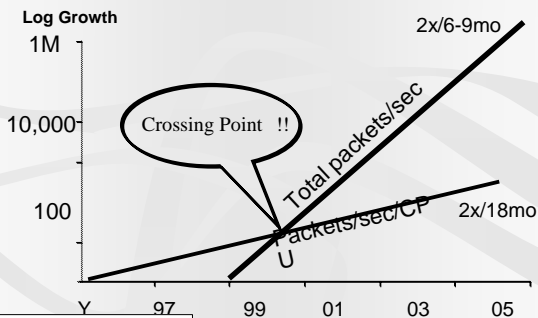
Sept. 8, 2003

Intel® Pentium® 2 Processor 1.40 GHz with
1.5MB L3 Cache for Technical Computing
and Front End Enterprise Applications

10GFLOPs !

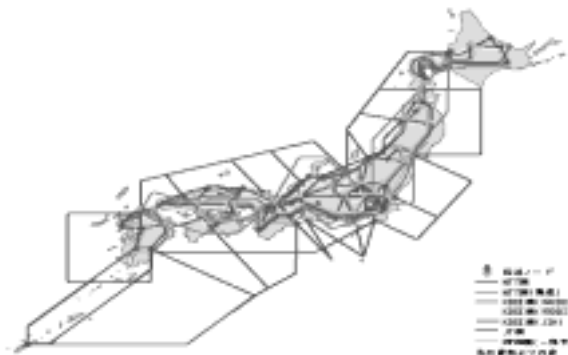


ネットワークの進化が コンピュータの進化を上回る



ギルダールの法則

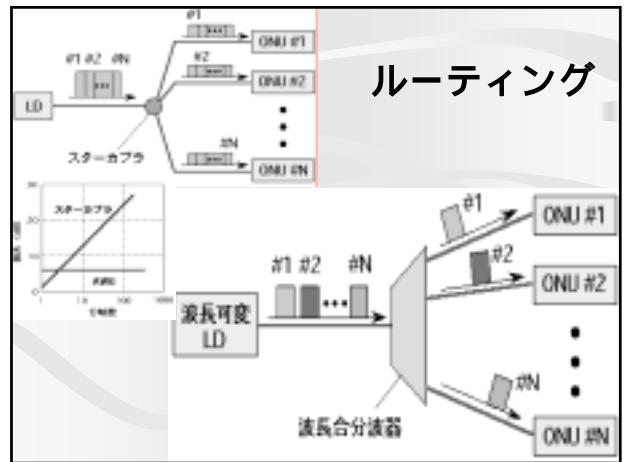
日本国内における基幹網



ネットワークの高速化

TDM(時分割多重)から
WDM(波長分割多重)へ

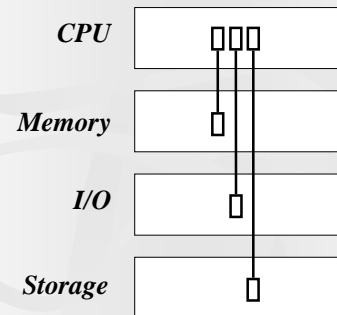
Wavelength Division Multiplexing



ネットワークがコンピュータの内部バスと同じくらい早くなれば、マシンは、特定の目的を持ったデバイスのあつまりへとネットワーク上で分解するだろう。

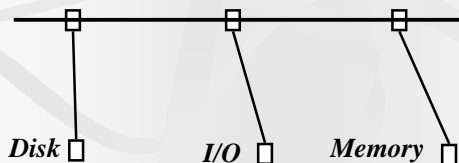
ギルダール

コンピュータの内部バス



コンピュータが
ネットワーク・デバイスに分解する

CPU

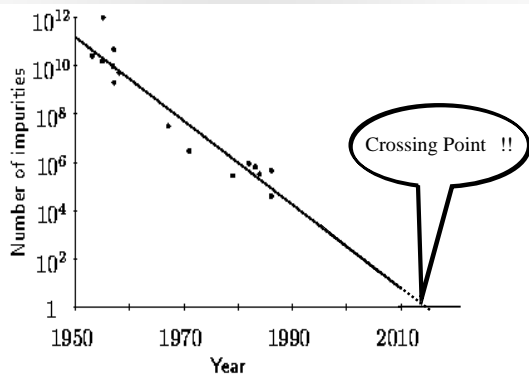


- “A Note on Distributed Computing”
(http://www.sunlabs.com/techrep/1994/sml_i_tr-94-29.pdf)

ローカルなプログラミングとリモートなプログラミングを、はっきりと区別すべきだという立場

Jim Waldo et al.

情報1bitを蓄えるのに必要な電子の数



新しい原理の探求



Church-Turing principle

- *Every 'function which would naturally be regarded as computable' can be computed by the universal Turing machine.*



- *'Every finitely realizable physical system can be perfectly simulated by a universal model computing machine operating by finite means'.*

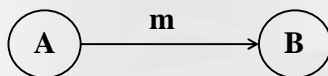
David Deutsch 1985

インスタンスの状態と「観測」

すべてのTuringマシンが状態を持つというトリビアルな話と、その状態が外部から観測可能であるという話は別の話である。

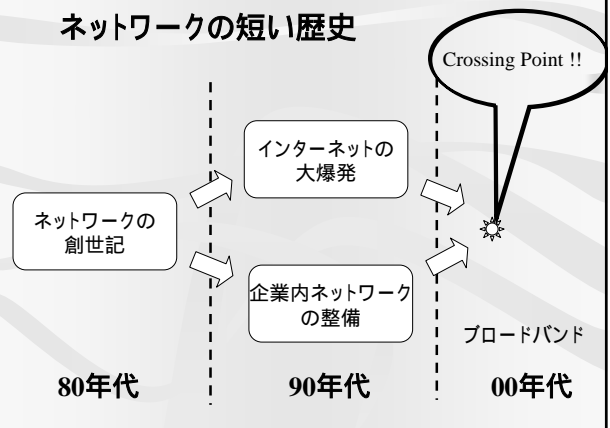
もしも、あるマシンの状態を外部から「観察」可能にしようと思えば、「観測」を、状態の問い合わせの入力に対する、状態の情報の出力としてモデル化すればいい。ただ、こうして「観測」されたマシンの「状態」は、マシン自体の定義が変わっているため、正確には、「観測」しようとしたものと、同じ「状態」とはいえない

インスタンスの「状態遷移」のモデル



あるインスタンスの状態が操作mによって、AからBに変わったとする。上のグラフは、そのことを表している。一方、インスタンスAが、インスタンスBに、メッセージmを発したとする。それも、同じグラフで表現できる。ただし、前者では、mの作用は瞬時に無限遠に伝わる「遠隔作用」とみなされるのに対して、後者では、mの伝播は、明らかに、光速を超えることは無い。

ネットワークの短い歴史



Enterprise Javaの主な動き

- 1995/05 : Java の誕生



- 1998/02 : XML 1.0
- 1999/12 : J2EE 1.2
- 2000/05 : SOAP 1.1
- 2001/03 : WSDL 1.1
- 2002/07 : UDDI
- 2002/11 : J2EE 1.4 beta



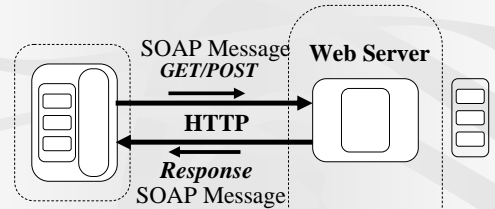
何故、Webサービスが注目されるのか？

Webサービスとは？

- Webサービスは、90年代半ばからのインターネットの爆発的な普及の中、WebとそのプロトコルとしてのHTTPの圧倒的な影響力の中、Webサーバを汎用のサーバとして利用する、サーバ・クライアント型の新しいネットワーク・プログラミングのスタイルとして登場した。

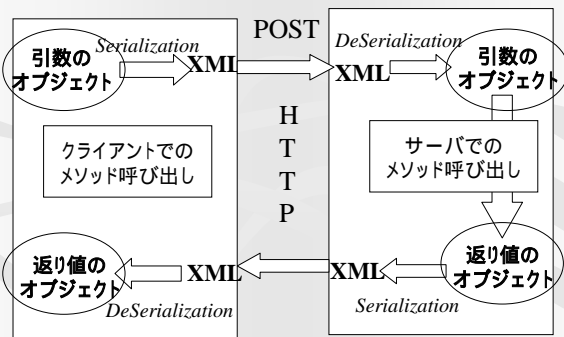
Webサービスとは？

汎用サーバとしてWebサーバを利用するサーバ・クライアント・モデル

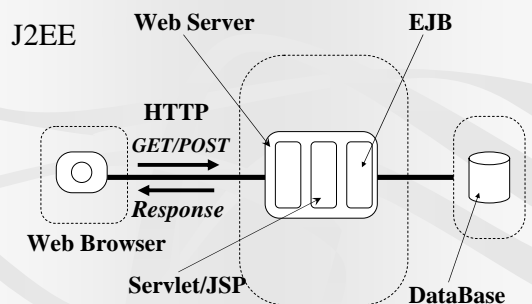


HTTP上でXMLで記述されたSOAPメッセージを交換

Webサービス SOAP-RPC



汎用クライアントとしてのWebブラウザ



HTTPへの変換器としてのミドルウェア

e-ビジネスの現状の問題を考える

- 二つのネットワーク
- 内部と外部の分離
- ネットワーク管理者の背理
- e-ビジネスのパラドックス
(急速な変化とスタンダードの未成熟)
- e-ビジネスの欠けたリング
(ネットワーク上には「市場」が未成立)

Lingua Franca



Webサービスは、まさにe-ビジネスの、「共通の言葉“Lingua Franca”」として、期待を集めている。

“Lingua Franca”の別の解釈



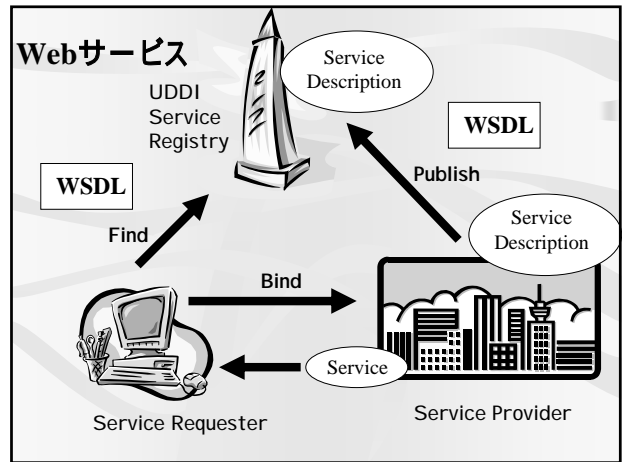
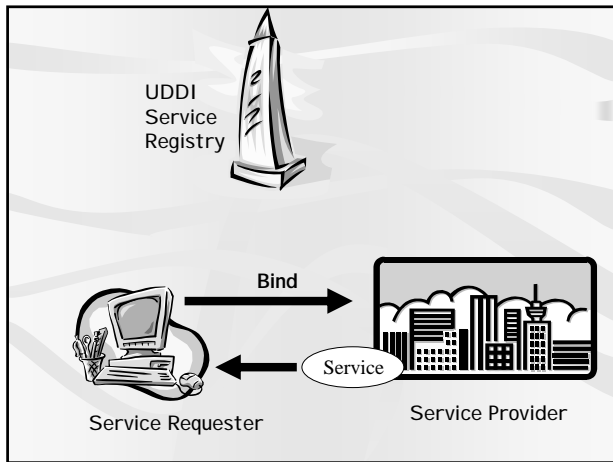
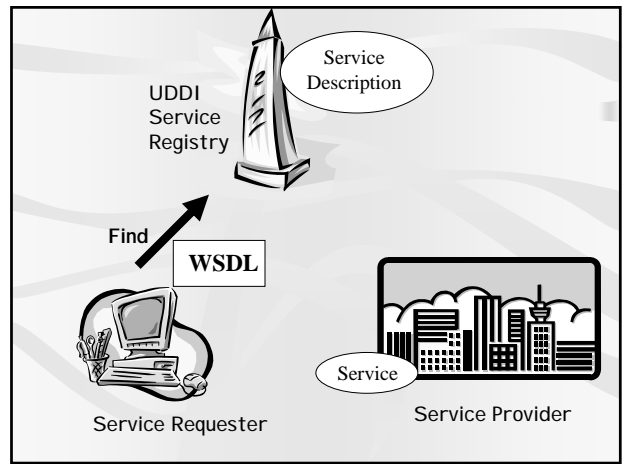
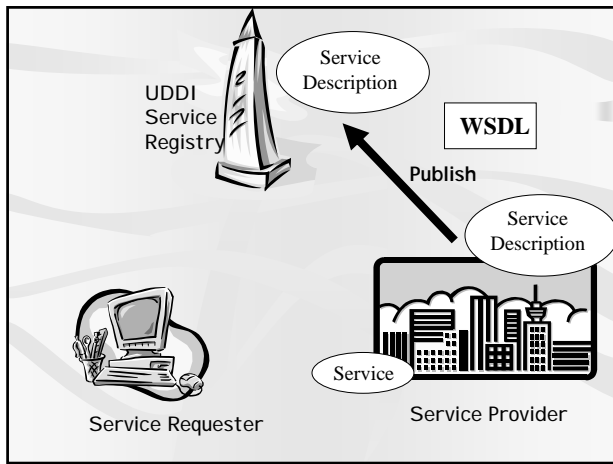
Hardware技術の変化は、革命的。
Software技術は、変化に対して保守的。

Webサービスが描く B2Bのシナリオ

B2Bでの期待

- Webサービスの相互運用性が、もっとも期待されるのは、企業間の電子的なビジネス取引（B2B）の展開においてである。
- 企業と企業のネットワーク同士を安全に結ぶには、80番のポートを使うしかないというセキュリティ上の歴史的に形成された特殊な事情も、現実的に相互運用可能なB2BシステムとしてのWebサービスへの期待を高めている。





JAX-RPCの登場

2002/01/11

Webサービス開発手法としての
JAX-RPC

JAX-RPCは、J2EEとWebサービスを結びつける手法として優れているだけでなく、一般的で強力なWebサービス開発手法である。

JAX-RPCは、Gridに技術的な基礎を提供した。

SOAP-RPCからJAX-RPCへ

JAX-RPC = RMI / SOAP



Rahul Sharma

JAX-RPC の特徴

- JavaとWSDLの対応付け
- Javaの型とXMLの型の対応付け
- JavaでのWebサービスの記述
- 多様なクライアントモデル

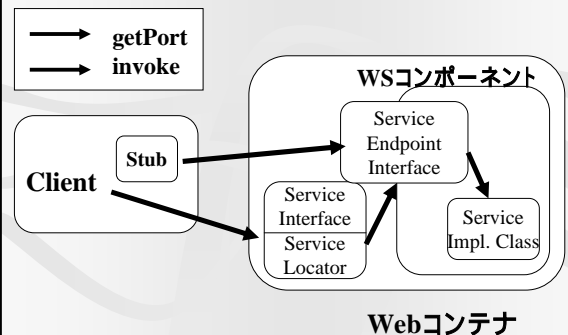
Java Remote Interface ⇔ WSDL

WSDL ⇔ Java クラス群

Webサービスの新しい定義

- Webサービスは、HTTP上のSOAPのような標準的なネットワーク・プロトコルを通じてアクセス可能な、WSDLによって記述されるソフトウェア・コンポーネントである。

JAX-RPCがWSDLから生成する クラスとコンテナ



J2EE1.4でのWebサービスの実装

2003/11/18

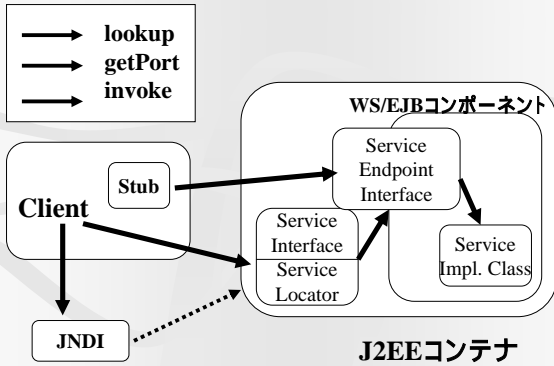
EJB層でのWebサービスとの統合の課題

Web層でJ2EEとWebサービスとを統合することはそれほど難しくはない。問題はEJB層。

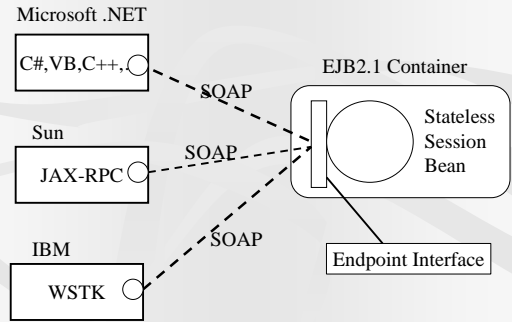
EJB層のEntityビーンもSessionビーンも、コンテナの外部とは、基本的にはRMIを使って通信しなければならない。

一方、Webサービスは、SOAPのプロトコルを使わなければならない。

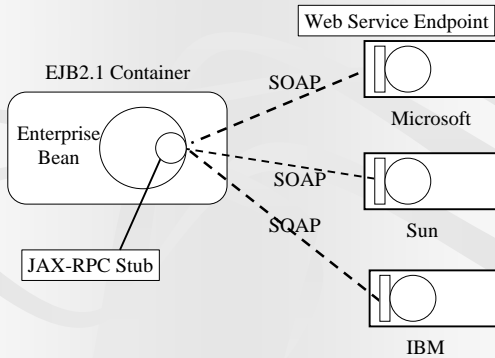
J2EE1.4のWebサービス対応



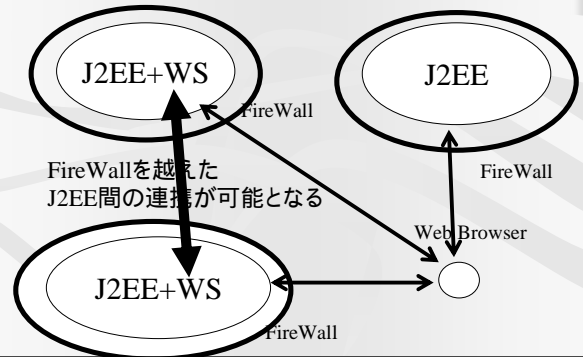
SOAP互換であれば、誰でも、セッション・ビーンの方法を呼び出せる



J2EEのビーンの中のJAX-RPCのStubからどんなWebサービスも呼び出せる



FireWallを越えたシステム間の連携 Enterprise Gridの基礎



Webサービスの発展段階

企業内のシステム統合

- これまでのシステム統合は、高価で複雑。かつ技術の転用が利かない。Webサービスを利用したシステム統合は、大きくコストを減らすことができる。
- Webサービスは、メインフレームからPCにいたる異機種混合環境でのシステム統合を可能とするので、ビジネスの必要に応じた柔軟なシステム構成によって、システムのトータルコストを削減出来る。

第一段階

企業間のネットワーク統合

- 企業内・系列内の範囲を超えて、グローバルな規模でビジネスを拡大しようとするのは、企業にとっては当然の指向である。
- しかし、企業内のネットワーク・システムを、企業の範囲を超えて拡大することは、簡単に出来ることではない。そこには、大きな断絶がある。

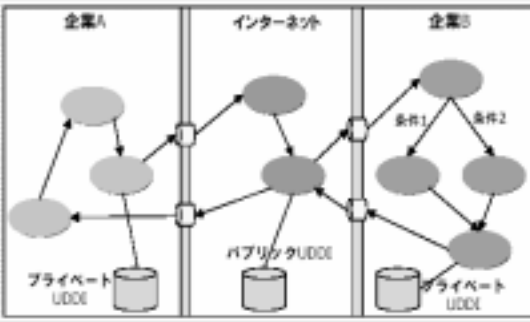
第二段階

ネットワーク上のサービス統合

- トランザクション(transaction)
- ワークフロー(workflow)
- オーケストレーション(orchestration)
- コレオグラフィ(choreography)
- コーディネーション(coordination)
- フェデレーション(federation)

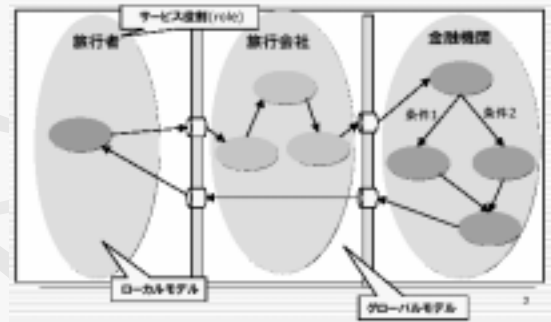
第三段階

日本IBM東京基礎研究所 浦本直彦
「サービスのフェデレーション」より

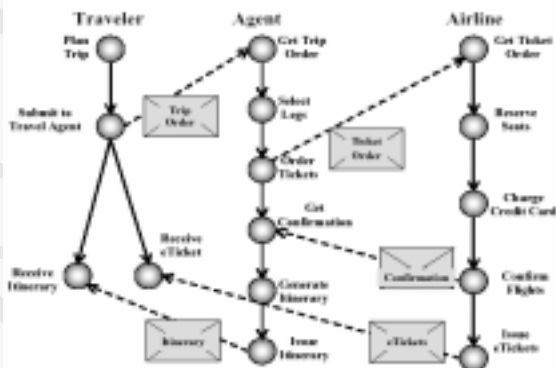


<http://www.c-sq.com/jump/maru.html>

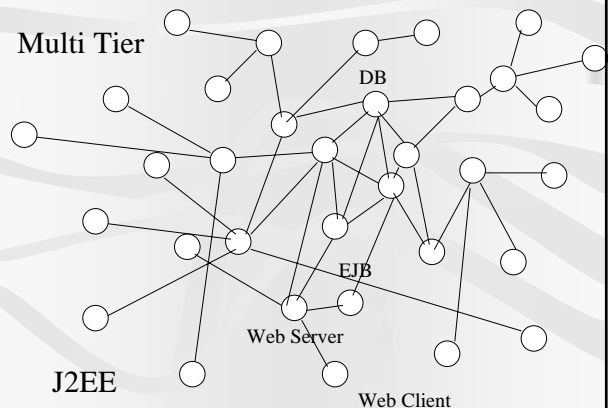
日本IBM東京基礎研究所 浦本直彦
「サービスのフェデレーション」より

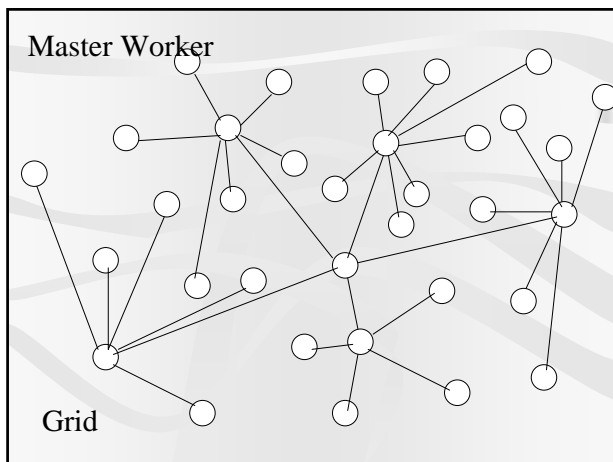


<http://www.c-sq.com/jump/maru.html>



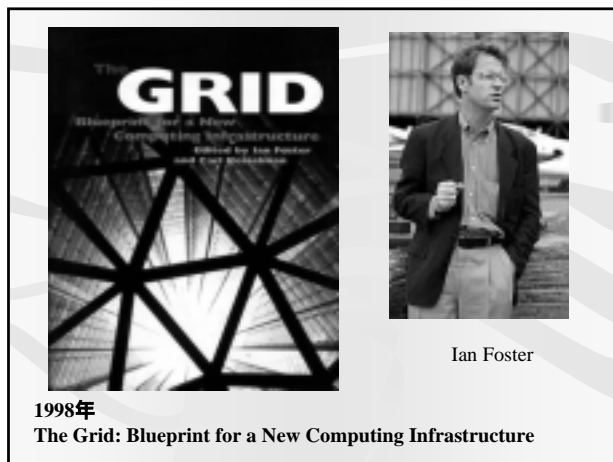
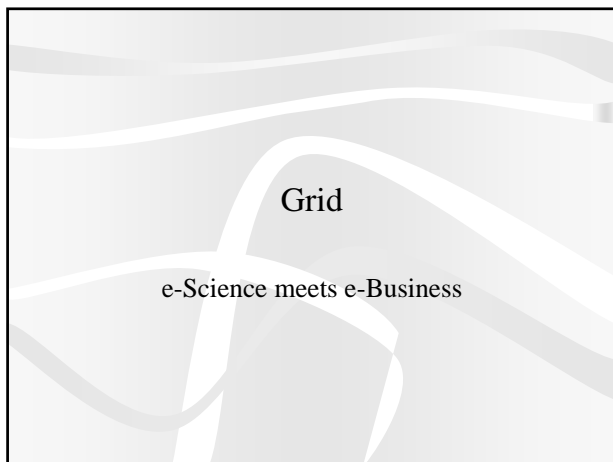
<http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf>





How to Find, Discover and Lookup Services on Network

	Jini	Web Service	JXTA
	Lookup	Find	Discover
Portal	Lookup Server	UDDI Server	なし
Active / Passive	(Active) (Multicast)	Passive	Active Propagation
探索対象	Service Template (Java Object)	WSDL (XML)	Advertisement (XML)



「電力グリッド」とのアナロジー

- 電気を使う時、我々は、電気がどこで作られたものなのかを意識する必要はない。正確に言えば、それを知ることは出来ない。
- 大事なポイントは、我々は、電力会社に対して電気供給という「サービス」を要求しているということ。
- 電力グリッドは、発電や送電の細部を利用者には隠しながら、同時に、簡単なインターフェースで、安定した電力サービスを提供することを可能にしている。

ネットワークとサービス

- 電力・通信・放送（ラジオ、テレビ）といった昔からの「ネットワーク産業」で、サービス概念が発達していた。
- こうした古いタイプのネットワーク産業は、サービスを流通させるネーション・ワイド（あるいは地球規模）の巨大なネットワークを構築してきた。
- コンピュータでのサービス概念の発生も、歴史的には、コンピュータのネットワーク利用の展開と結びついていた。

Gridとは？

- ネットワーク上で複数のマシンが仮想的な組織を形成して協調動作し、ユーザ（主体）にネットワークを通じて一つのまとまりのあるサービスを提供する時、その提供者をGridと呼ぶ。
- Gridサービスの利用者は、Gridにサービスの提供を要求するだけで、Grid内部での処理に関心を持たない。

新しいGrid概念の登場

GT3とOGSA/OGSI

Globus Toolkit 3.0



GT2
から
GT3
へ

<http://www.globus.org/>

GGF OGSA/OGSI

Open Grid Service Architecture / Open Grid Service Infrastructure



<http://www.gridforum.org/>

GT3でのGrid概念の新しさ

- マシンを、PlatformやOSの違いを超えて、ネットワーク上で、結合・連携させる。
- その結合・連携には、汎用的なネットワークのリソースが利用可能なWebサービスを利用して、マシンを「疎結合」で結ぶ。
- 従来のコンピューティング・パワー指向のGridに対して、リソース共有を指向するGrid

Internet Computing と
Grid Computing

I.Foster

「インターネット・コンピューティングとGridの出現」
ネイチャー誌
(<http://www.nature.com/nature/webmatters/grid/grid.html>)

インターネット・コンピューティングは、そうでもしなければ何の仕事もしない、PCやワークステーションを探し出して利用して、グローバルな到達範囲とスーパーコンピュータの能力を持つ強力な分散コンピューティングシステムを作り上げるのだ。

David Anderson

SETI@home プロジェクト

(<http://setiathome.ssl.berkeley.edu/>)



Scott Kurowski

Entropia社

(<http://www.entropia.com/>)



<http://fightaidsathome.scripps.edu/>



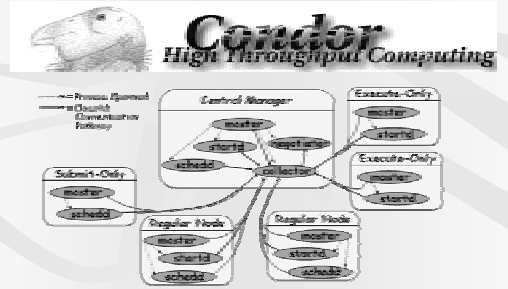
<http://www.mersenne.org/prime.htm>



<http://www.parabon.com/cac.jsp>

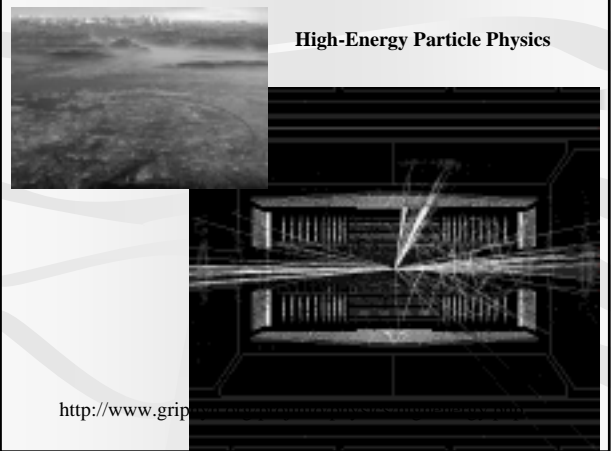
e-ScienceとGrid

Miron Livny
「コンドル・プロジェクト」
(<http://www.cs.wisc.edu/condor/>)



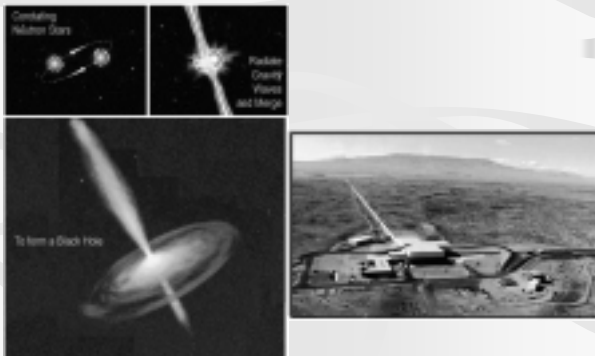
<http://www.griphyn.org/index.php>

High-Energy Particle Physics



<http://www.griphyn.org>

LIGO: Detecting Einstein's Gravitational Waves



<http://www.griphyn.org/projinfo/physics/ligo.php>



<http://www.ipg.nasa.gov/>



<http://eu-datagrid.web.cern.ch/eu-datagrid/>

e-Science meets e-Business

The Anatomy of the Grid (2001)
 The Physiology of the Grid (2002)
 Globus Toolkit 3.0 (2003)

“The Anatomy of the Grid” 「Gridの解剖学」

<http://www.globus.org/research/papers/anatomy.pdf>
 2001

- “Enabling Scalable Virtual Organizations”
 「スケーラブルな仮想組織体を可能にする」というサブタイトルを持つ。
- Gridの重要なコンセプトの一つである仮想的な組織 (“Virtual Organizations”) という考えを初めて定式化した論文。
- このVO (“Virtual Organizations”) という視点の発見が、今日のGridへの飛躍の最初の一步だった。

OGSAの提案

GridへのWebサービスの全面的な導入

「Gridの生理学」

“The Physiology of the Grid”

<http://www-nix.globus.org/ogsa/docs/alpha/physiology.pdf>

2002/06/22

「Grid技術にたいする必要性」

- 「エンタープライズのコンピュータ技術の進化」
- 「サービスプロバイダとB2Bコンピュータ技術」

“The Physiology of the Grid”

- この論文が画期的だったのは、Gridサービスの実現に、e-Businessの世界で急速に標準的な地位を占めつつあるWebサービスを利用することを明確に宣言したこと。
- GridとWebサービスの提携と、それによる双方の能力の拡大を、“Open Grid Services Architecture (OGSA)” と呼ぶことが提案される。

OGSI とGWSDL

portTypeの継承機能

すべてのGridサービスは、
portType GridServiceを
継承したものである。



すべてのクラスは、
Class objectを
継承したものである。

基本的なportTypeと portTypeの継承

```
<gwsdl:portType name="FactoryServiceGroup"
  extends="ogsi:Factory ogsi:ServiceGroup"/>
.....
</gwsdl:portType>
<gwsdl:portType name="NotificationFactory"
  extends="ogsi:NotificationSource ogsi:Factory ogsi:ServiceGroup"/>
.....
</gwsdl:portType>
<gwsdl:portType name="NotificationServiceGroup"
  extends="ogsi:NotificationSource ogsi:ServiceGroup"/>
.....
</gwsdl:portType>
<gwsdl:portType name="ServiceGroupRegistrationNotification"
  extends="ogsi:ServiceGroupRegistration ogsi:NotificationSource"/>
.....
</gwsdl:portType>
```

すべてのGridサービスがそれを
継承する portType GridServiceは、
どのようなserviceDataを持っているか



GenericなObjectが
持つべき性質・Fieldを
設計する

基本的なportTypeのserviceData

```
<gwsdl:portType name="Factory" extends="ogsi:GridService">
.....
  <sd:serviceData name="createServiceExtensibility" ..../>
</gwsdl:portType>

<gwsdl:portType name="NotificationSource"
  extends="ogsi:GridService">
.....
  <sd:serviceData name="notifiableServiceDataName" ..../>
  <sd:serviceData name="subscribeExtensibility"
    type="ogsi:OperationExtensibilityType"..../>
  <sd:staticServiceDataValues>....</sd:staticServiceDataValues>
</gwsdl:portType>
```

すべてのGridサービスがそれを
継承する portType GridServiceは、
どのようなoperationを持っているか



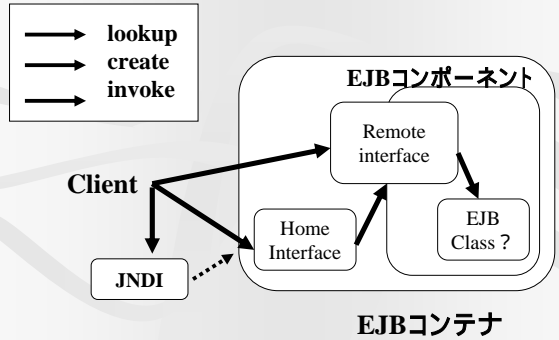
GenericなObjectが
持つべきmethodを
考える

```
<gwsdl:portType name="GridService">
  <operation name="findServiceData" .... />
  <operation name="setServiceData" .... />
  <operation name="requestTerminationAfter" .... />
  <operation name="requestTerminationBefore" .... />
  <operation name="destroy" .... />
  <sd:serviceData name="interface".... />
  <sd:serviceData name="serviceName" .... />
  <sd:serviceData name="factoryLocator" .... />
  <sd:serviceData name="gridServiceHandle" .... />
  <sd:serviceData name="gridServiceReference" .... />
  <sd:serviceData name="findServiceDataExtensibility" .... />
  <sd:serviceData name="setServiceDataExtensibility" .... />
  <sd:serviceData name="terminationTime" .... />
  <sd:staticServiceDataValues> ....
</sd:staticServiceDataValues>
</gwsdl:portType>
```

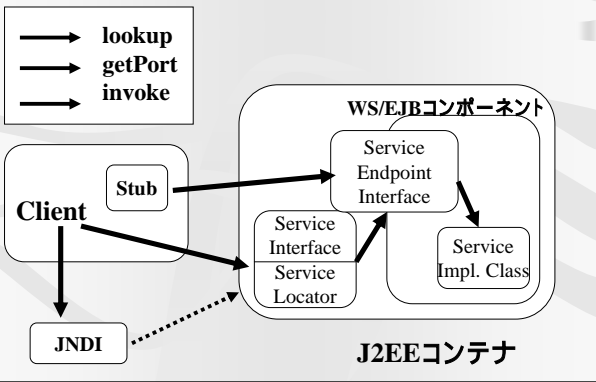
GridService

GridでのContainerの導入 --- GT3のHosting Environment概念 ---

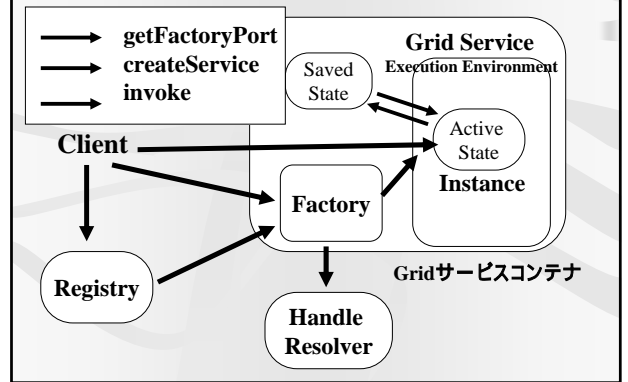
J2EEのコンテナと EJBの三つの定義ファイル



J2EE1.4のWebサービス対応



Gridサービスコンテナ



J2EE, Web Service, Grid の コンテナ / コンポーネントの比較

	J2EE	Web Service	Grid Service
Containerの発見	JNDI	UDDI / WSIL	Registry
Containerのインターフェース	Home Intf.	Service Intf.	Factory
Componentの生成	create	get<Port>	createService
Componentのインターフェース	Remote Intf.	Remote Endpoint	Handle / Reference

OGSA / OGSi

