

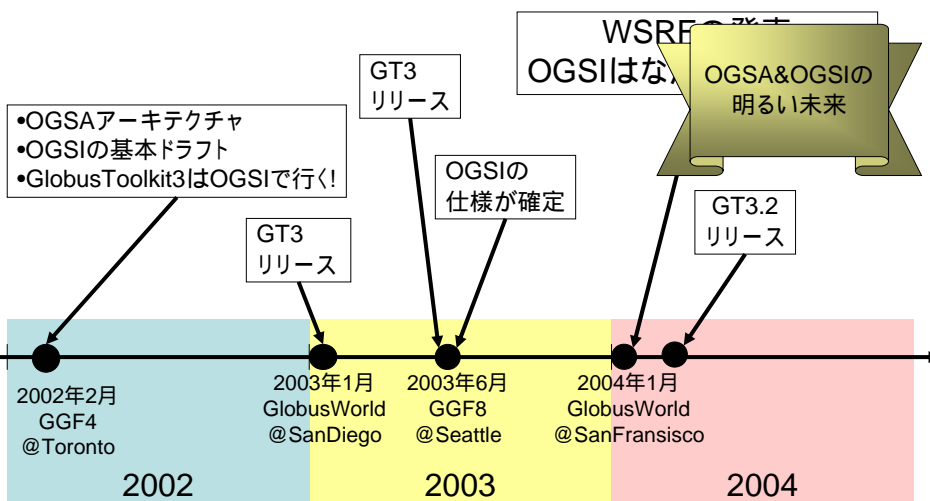
WSRF

- Web Service Resource Framework

産業技術総合研究所 グリッド研究センター
中田秀基



またかよ！ Globus+IBM

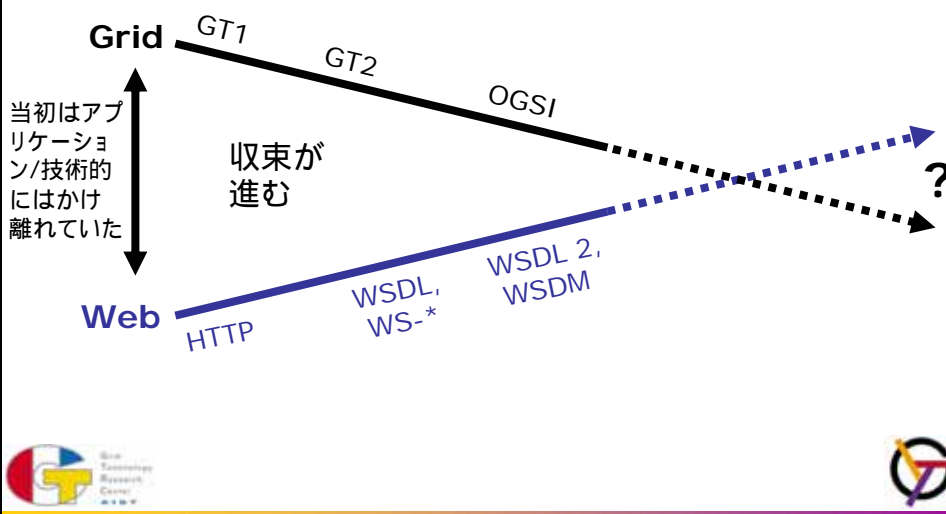


WSRF以降の世界

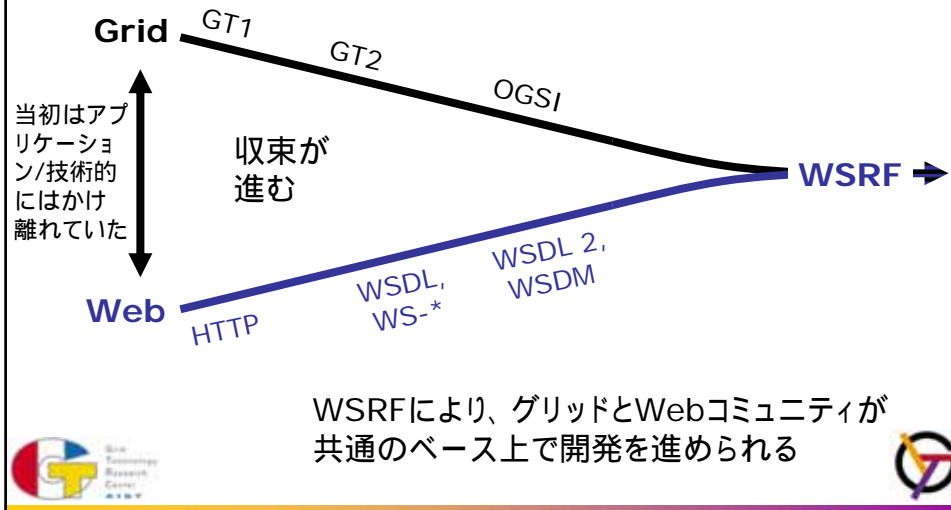
- OGSiはなかったことに
 - ▶ OGSAはWSRF上に実装
- OGSiを6つのWeb Service コンポーネントに分割し、リファクタリングした**WSRF**に再構成
- WSRFがOGSIを完全にリプレイス
- WSRFをベースにしたGT4が登場する
 - ▶ GT3はお払い箱



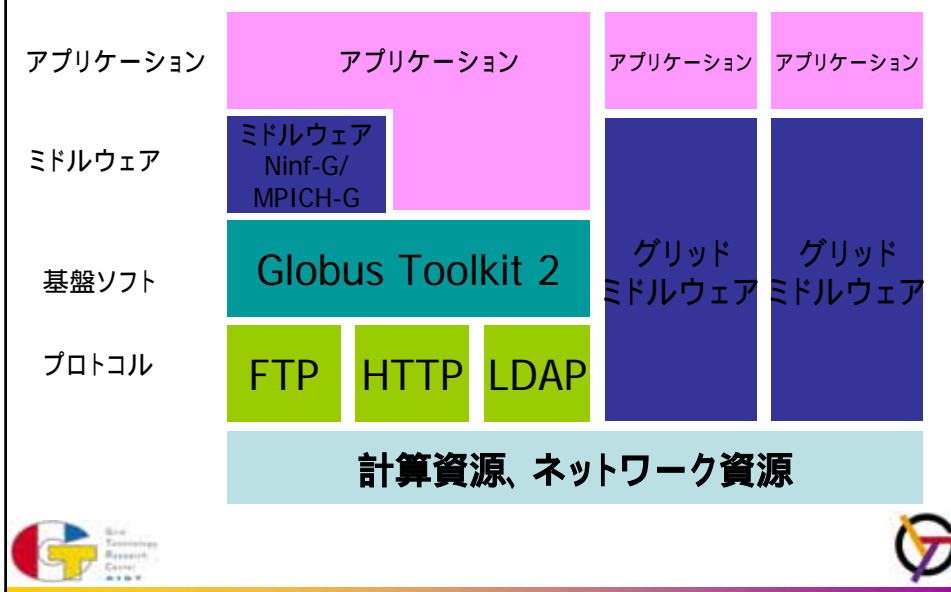
GridとWSRF – Ian Foster のスライドから



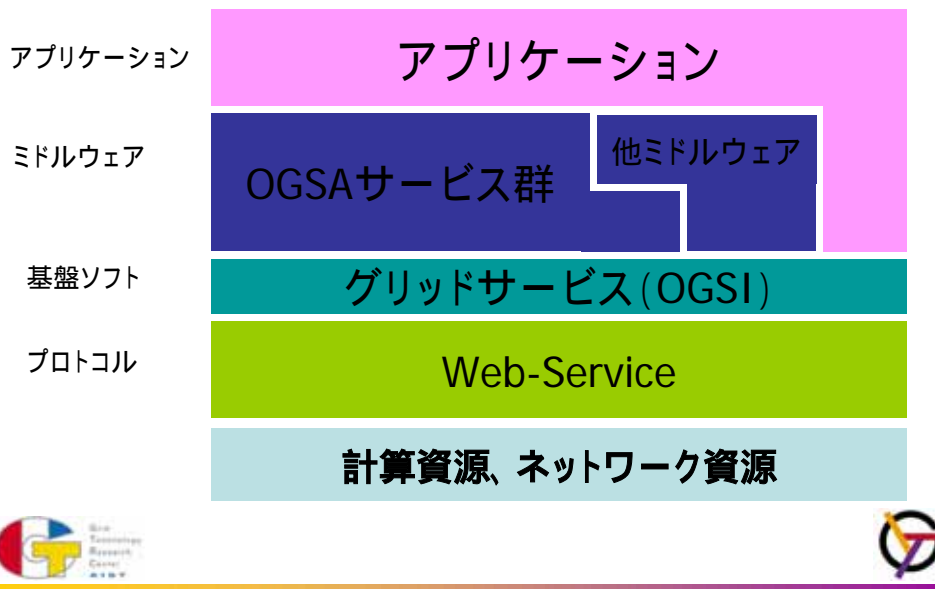
GridとWSRF – Ian Foster のスライドから



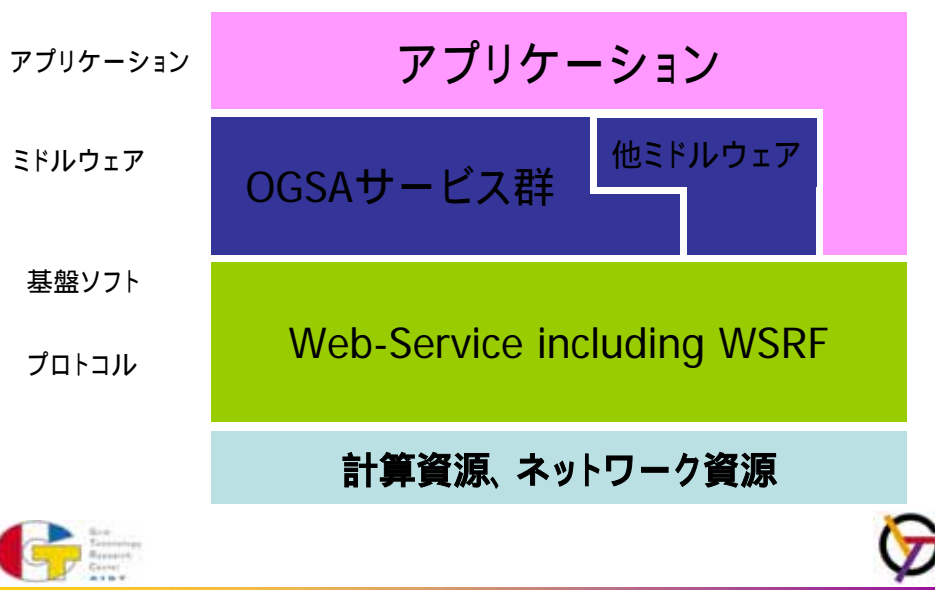
OGSA以前



OGSA以後



OGSA以後 – WSRFの登場



なぜWSRFを定義するのか

● OGSiのなにが気に入らなかったのか？

- ▶ ひとつの仕様なのにでかすぎる
 - WSRF では6つのWS-XXX に分割
- ▶ Webserviceのツール群と相性が悪い
 - Xsd:any を使いすぎ
 - WSRF では控えめになっている
- ▶ オブジェクト指向過ぎ
 - WSRF では「サービス」と、状態を持つ「リソース」を明示的に分離



WSRFの要素

- WS-ResourceLifetime
- WS-ResourceProperties
- WS-RenewableReferences
- WS-ServiceGroup
- WS-BaseFaults
- WS-Notification

WSRF

- WS-Addressing

WSRFとは無関係に
定義された枠組み



グリッドサービスとはなんだったのか – OGSIの本質

● Webサービスはステイトレス

- ▶ 状態がもてないのではコンポーネントとしてお話にならない



● グリッドサービスはステイトフル

- 状態の表現
 - グリッドサービスデータ
- 資源を使いつぶさないためには生成と破棄の概念が必要
 - Factoryとlifetime management
- 生成されたものを指し示すものが必要
 - GSHとGSR



ステイトレス vs. ステイトフル

● ステイト – サービスの内部状態

● ステイトフル

- ▶ まったく同じようにアクセスしても内部状態に応じて挙動が異なる可能性がある
- ▶ ごく普通のオブジェクトのような概念
- ▶ 概念的に直感的で構成が容易

● ステイトレス

- ▶ 同じようにアクセスしたら同じ挙動を示す
- ▶ 実装が容易 – 耐故障性上有利
- ▶ 概念的には若干わかりにくい



グリッドサービス ウェブサービス+リソース

● グリッドサービス

- ▶ オペレーション + ステイト



● ウェブサービス

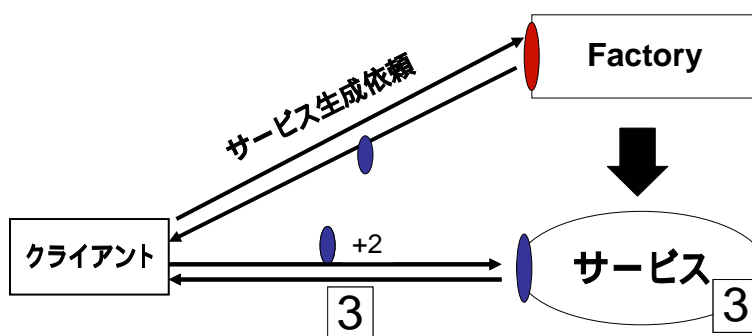
- ▶ オペレーションのみ

● リソース

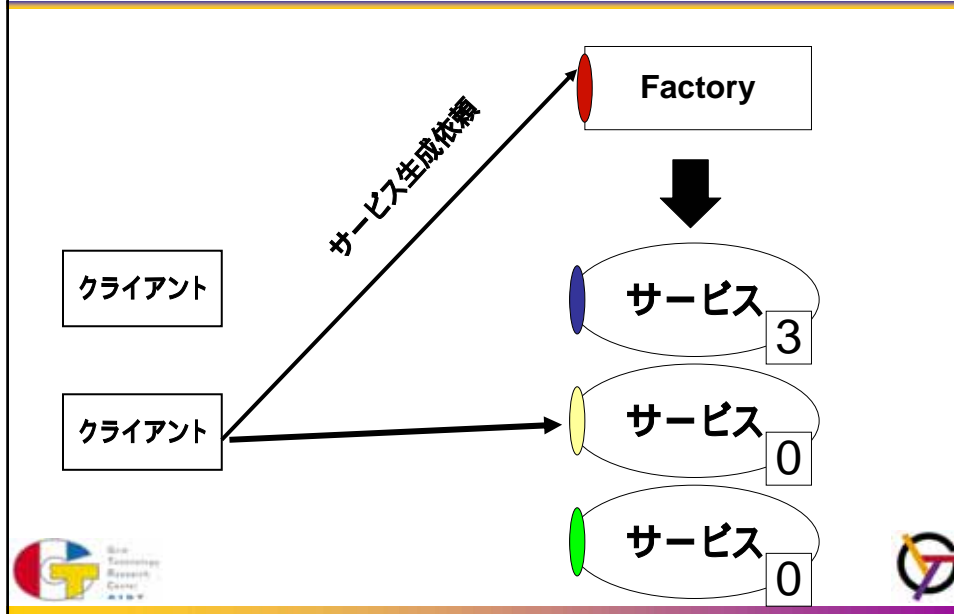
- ▶ ステイトのみ



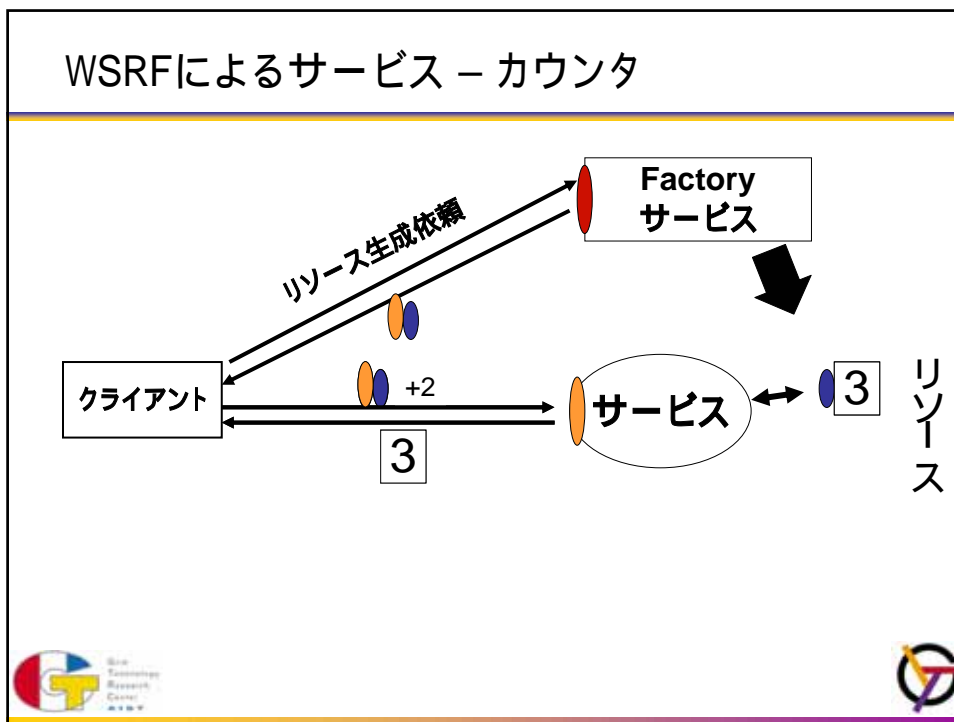
OGSIのグリッドサービス – カウンタ



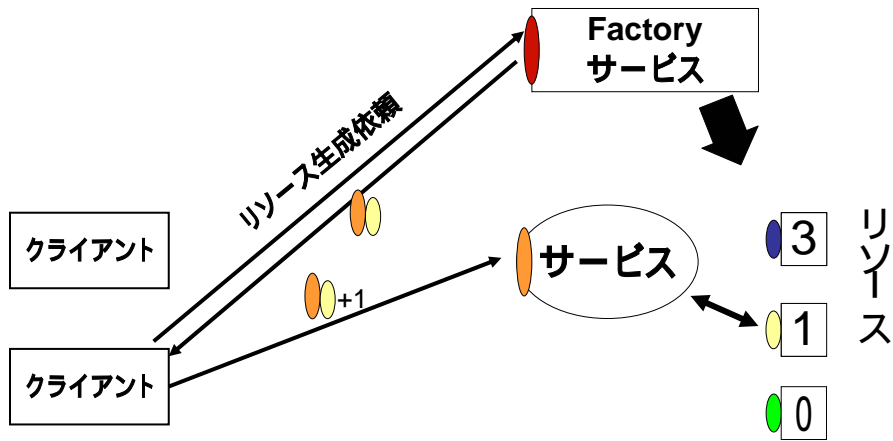
OGSIのグリッドサービス - カウンタ



WSRFによるサービス - カウンタ



WSRFによるサービス – カウンタ



サービスは永続的で、リソースだけが作られる



WS-Addressing

- **WSRFの一部ではない**
 - ▶ 既存のWS規格のひとつ
 - ▶ アドレスに付随する情報を相互運用可能な形で受け渡すためのプロトコル
 - ▶ 具体的にはSOAPのヘッダに情報を入れる
- **サービスのURLとプロパティとをペアにして保持するために利用**

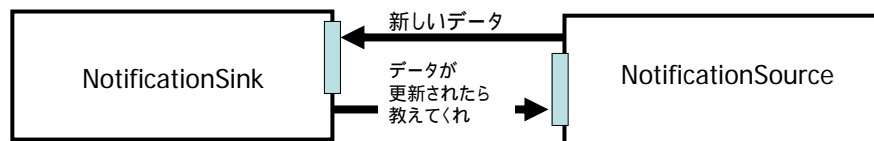


OGSIとWSRFの対応

OGSI	WSRF
Grid Service Reference	<i>WS-Addressing</i> Endpoint Reference
Grid Service Handle	<i>WS-Addressing</i> Endpoint Reference
HandleResolver ポートタイプ	WS-RenewableReferences
サービスデータ	WS-ResourceProperties
GridService 生存期間管理	WS-ResourceLifeCycle
Notification ポートタイプ	WS-Notification
Factory ポートタイプ	Treated as a pattern
ServiceGroup ポートタイプ	WS-ServiceGroup
Base fault type	WS-BaseFaults

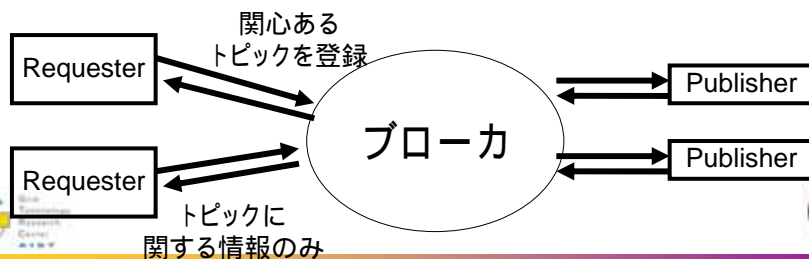
WS-Notification

- OGSIのNotification **ポートタイプ群**よりも機能が増えている
- OGSIでは NotificationSourceとNotificationSinkだけ



- WS-Notification ではブローカを導入可能

▶ トピックによるフィルタリング



WSRFとOGSIはどのくらい違うのか

● 機能的にはほとんど変わらない

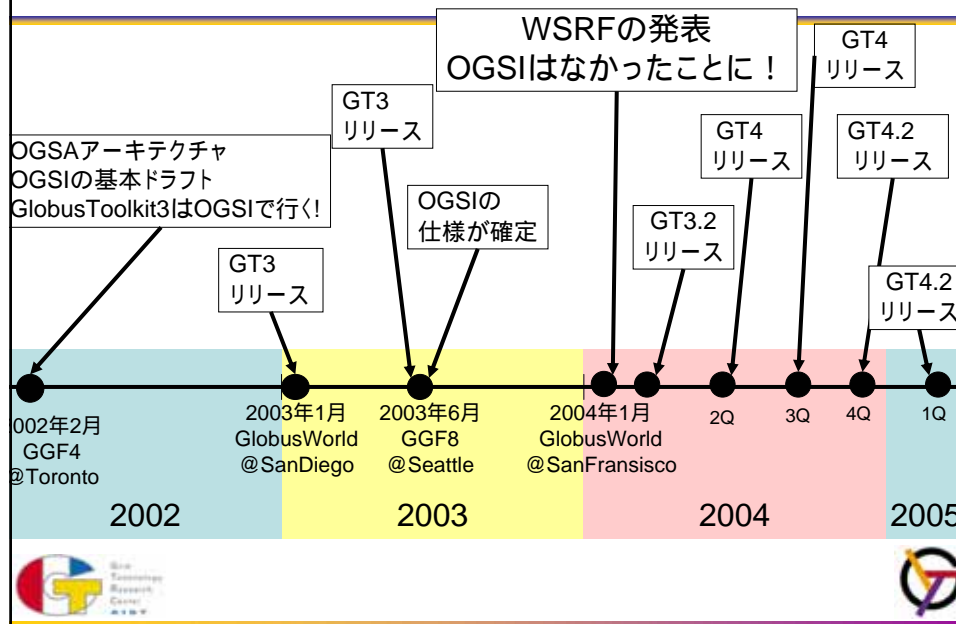
- ▶ OGSIでできたことは(ごく一部の例外を除くと)そのままWSRFでもできる
- ▶ 特にクライアント側ではAPIを変更しないことも可能かもしれない
- ▶ サーバ側にもOGSI互換のコンテナを作ることもできるかもしれない

● 一番変わったのは名前

- ▶ OGSI 2 と言われたら納得するような内容
- ▶ 名前を変更することによるメッセージ



今後の予定



WSRFのインパクト (ネガティブ編)

- 「グリッドサービス」という概念が消えてなくなった
 - ▶ 言葉の問題に過ぎないが象徴的
- 今後の標準化は(おそらく)OASISで行われる
 - ▶ OGSII-WG でも議論はするが、標準化はOASIS(多分)
 - ▶ 「The development of WSRF specifications demanded expertise, particularly in Web services standards, that was not represented in the OGSII WG. 」
FAQより
 - ▶ OGSII-WGっていったい。。。 Primerも作ったのに。。。
- Globus+IBMの思惑に振り回されて不愉快
 - ▶ IBM内部でのグリッド派とWS派の対立の余波？
- 使い物になるベースソフトウェア(GT3.2/GT4.2)の登場が一年先延ばし延期
 - ▶ それも、スケジュールどおりに出るならば。。。



WSRFのインパクト (ポジティブ編)

- テクニカルには(たぶん)正しい方向性
 - ▶ たしかにOGSIはやりすぎだったかも
 - ▶ 既存ツールがどこまで簡単に利用できるのか見当がつかないが、OGSIを実装するよりはWSRFを実装するほうが楽そう
- (ナイーブな実装の場合には)OGSIよりも軽量であることが期待できる
 - ▶ インターフェイスをOGSIにしたままでも、同様な軽量実装がおそらく可能だが、プログラミングは大変。



WSRFのインパクト (ニュートラル編)

● OGSAへの影響はない

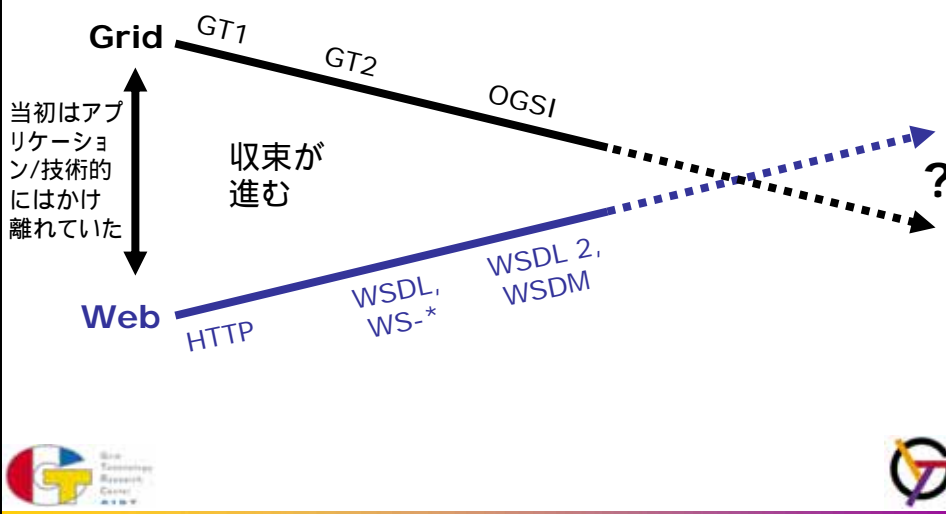
- ▶ OGSAは下部インフラに依存しない(by 岸本さん)
 - Ⓜ まだモノがないから、別に今すぐ実装できなくても困らない?

● ユーザへの影響

- ▶ GT3からGT4へ
 - Ⓜ プログラミングモデルにはそれほど大きな影響はでないのではないか、と推察
- ▶ GT3でのプロジェクトを考慮中の場合は、とりあえずGT3で作って、GT4が出た時点で移行を考えてほしい、とのこと。



GridとWSRF – Ian Foster のスライドから



GridとWSRF – 本当は？

