

ファイルシステム Gfarm の インストールと利用

谷村 勇輔

産業技術総合研究所 グリッド研究センター



本日の実習内容

■ Gfarm の概要

- ▶ Gfarm を使って何ができるのか？
- ▶ アーキテクチャ

■ Gfarm のインストールとセットアップ

- ▶ Gfarm, GfarmFS-FUSE のインストール
- ▶ セットアップと動作確認

■ Gfarm を用いたデータ処理

- ▶ 基本コマンド
- ▶ 並列データ処理

■ Q&A

Gfarm とは

■ Grid Data Farm で提案された仕様の実装

- ▶ 高エネルギー物理のための大規模データ処理
 - ⊗ 1ヶ所で生成されたデータの迅速な配置・展開
 - ⊗ データの所在を生かしたジョブ起動
- ▶ 地球規模の仮想コンピュータ(CPU+ストレージ)の実現

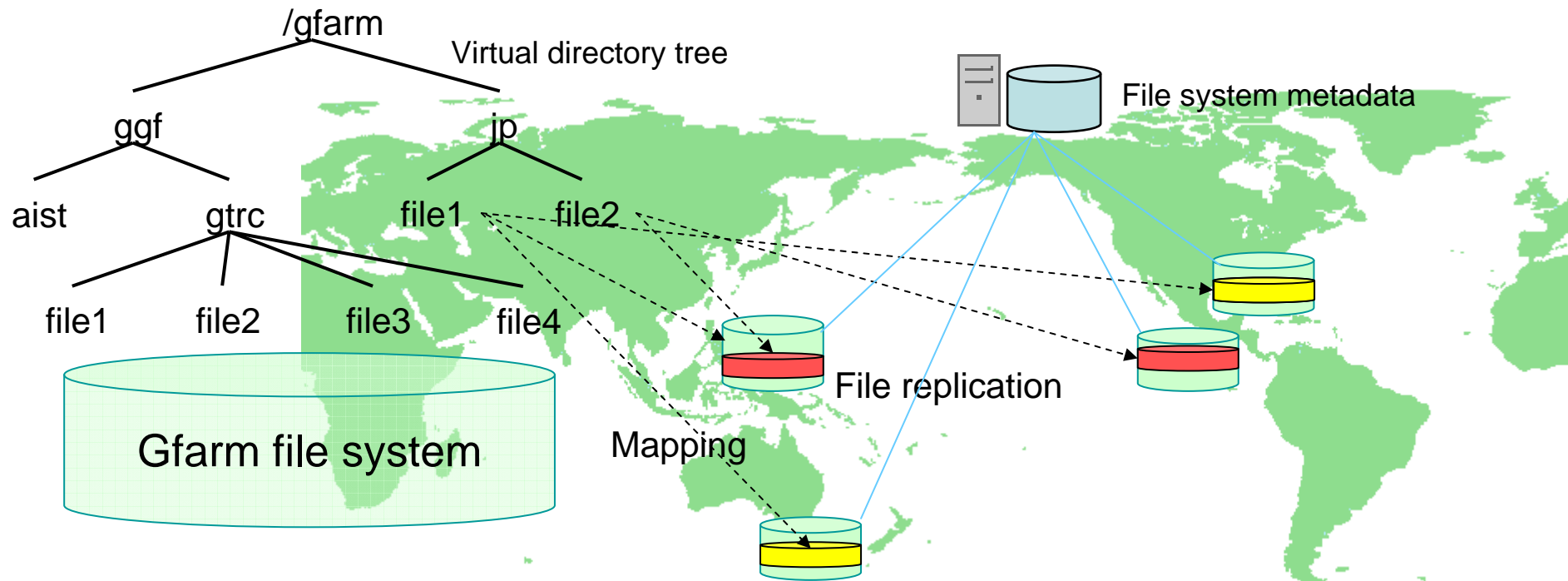
■ 次世代のネットワーク・ファイルシステム

- ▶ スケーラブルなクラスタ・ファイルシステム
 - ⊗ ストライピングによるファイル I/O の高速化ではない
 - ⊗ ファイルの局所アクセスを利用したスケーラビリティの確保
- ▶ 広域ネットワーク上の分散ファイルシステム
 - ⊗ グリッド・ファイルシステム

Gfarm ファイルシステム (1)

■ Gfarm が提供するファイルシステム

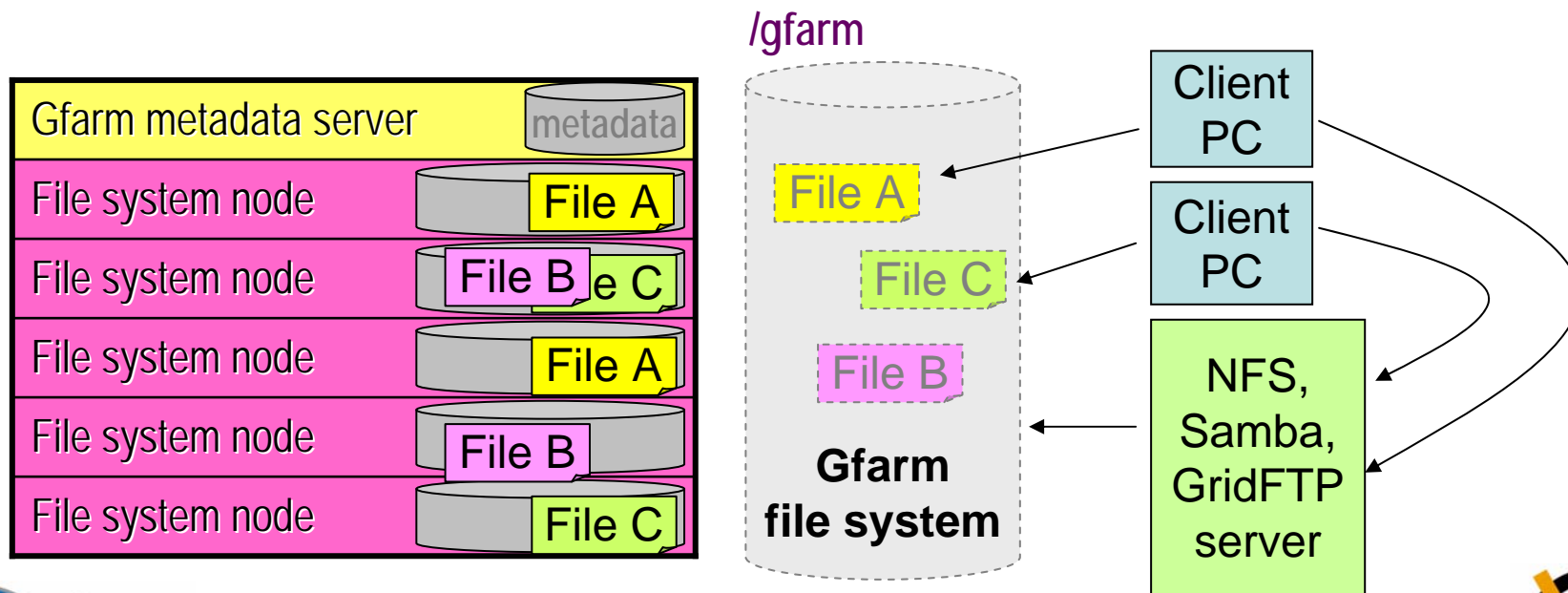
- ▶ グローバル名を利用したファイルへの透過的なアクセス
- ▶ ファイルを複製して持つことで、耐故障性を得たり、ファイルアクセスの負荷分散を行ったりすることが可能



Gfarm ファイルシステム (2)

■ アプリケーション・ユーザから見た利点

- ▶ ファイルの所在, 複製の有無を気にしなくて良い
- ▶ クライアントを含む全ノードでファイルが共有されている
- ▶ 既存のアプリケーションをそのまま利用できる



Gfarm の歴史

- 2001** ペタバイト規模の Data Intensive Computing のためのフレームワーク「Grid Data Farm」を提案(現筑波大 建部ら)
- 2002** SC の High-performance Bandwidth Challenge にて「データ複製により 2.286 Gbps のバンド幅を利用」を達成
- 2003** 4月に Gfarm v1.0 beta1, 11月に v1.0 を正式にリリース
SC の High-performance Bandwidth Challenge にて「Distributed Infrastructure Award」を受賞
- 2004** Gfarm v1.1 をリリース
- 2005** Gfarm v1.2 をリリース
SC の StorCloud Challenge にて「Most Innovative Use of Storage in Support of Science Award」を受賞
- 2006** Gfarm v1.3 をリリース
- 2007** Gfarm v1.4.1 をリリース (3月末)

Gfarm の特徴

- **コモディティ PC ベースのアーキテクチャ**
 - ▶ 特別なハードウェアが不要
 - ▶ HDD 付きのノードを追加して、容易にディスクスペースを増設可能
 - ▶ 主に Linux をサポート
 - ▶ オープンソース

- **大規模な並列データ処理をサポート**
 - ▶ ファイルの所在を考慮したジョブ起動の仕組みを提供
 - ▶ 既存のアプリケーション・プログラムの変更が不要
 - ▶ 手動, あるいはオンデマンドでのファイル複製機能を提供

- **グリッド・ファイルシステムとしての機能をサポート**
 - ▶ グローバルな名前空間でファイルにアクセス可能
 - ▶ GSI 認証, 通信経路の暗号化をサポート
 - ▶ WAN 向けの通信パラメータのチューニングが可能

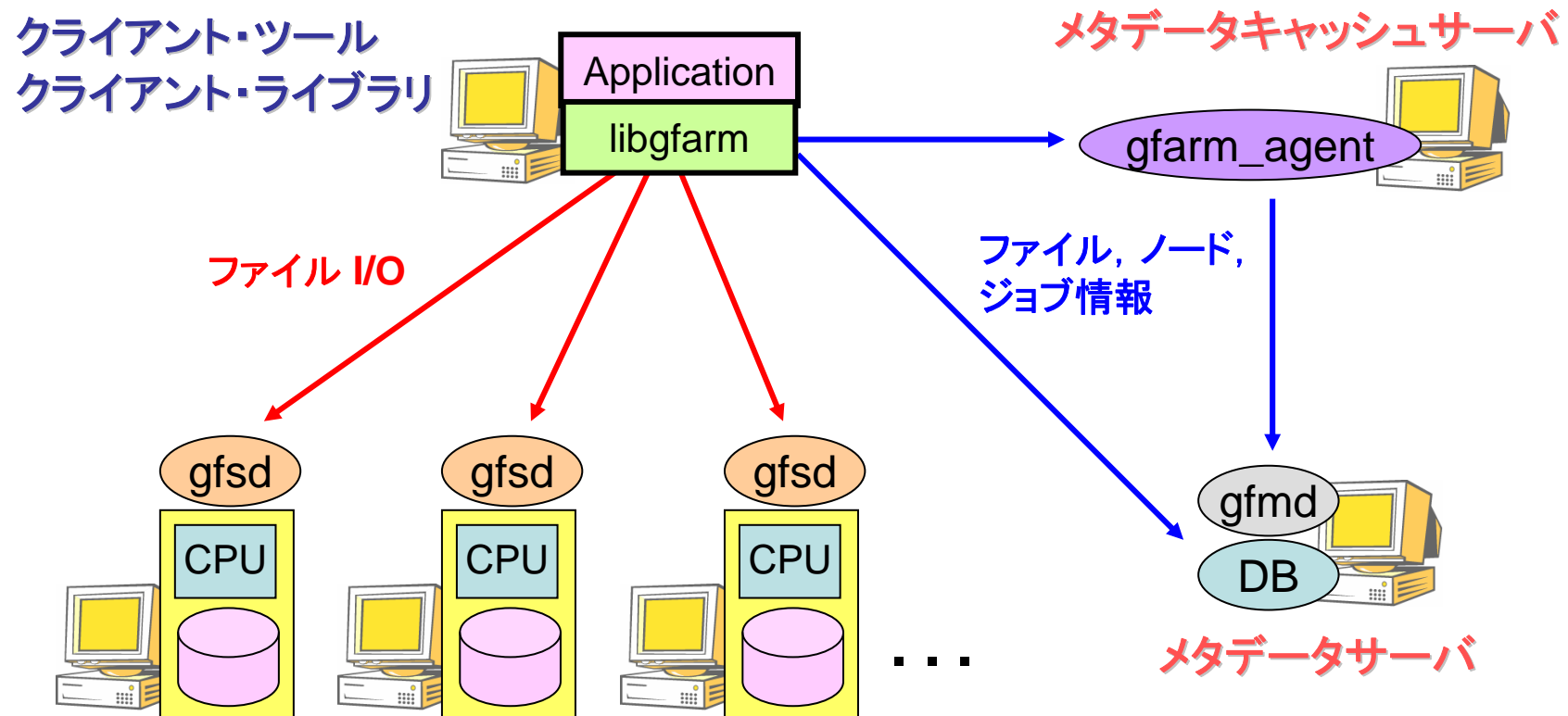
Gfarm の利用シナリオ

- NFS の代替としての共有ファイルシステム
 - ▶ 安価なクラスタファイルシステム
- ファイル複製機能を利用したデータのバックアップ
 - ▶ ディザスタ・リカバリ
- 大規模な並列データ処理, ワークフローを伴う計算のためのストレージシステム
 - ▶ 科学技術計算向けのストレージ
- ペタバイト規模のデータの単一リポジトリ
 - ▶ デジタル・ライブラリ

Gfarm のアーキテクチャ (1)

■ Gfarm v1.x の設計仕様

- ▶ ただし、メタデータキャッシュサーバは v1.3 以降



Gfarm のアーキテクチャ (2)

■ クライアント・ツール

- ▶ `gfarm-client`: Gfarm の基本コマンド
- ▶ システムコールフック, `GfarmFS-FUSE`: `libgfarm` の上位ツール

■ クライアント・ライブラリ: `libgfarm`

- ▶ Gfarm API ライブラリ

■ メタデータキャッシュサーバ: `gfarm_agent`

- ▶ メタデータをキャッシュし, 複数クライアントで共有する仕組みを提供

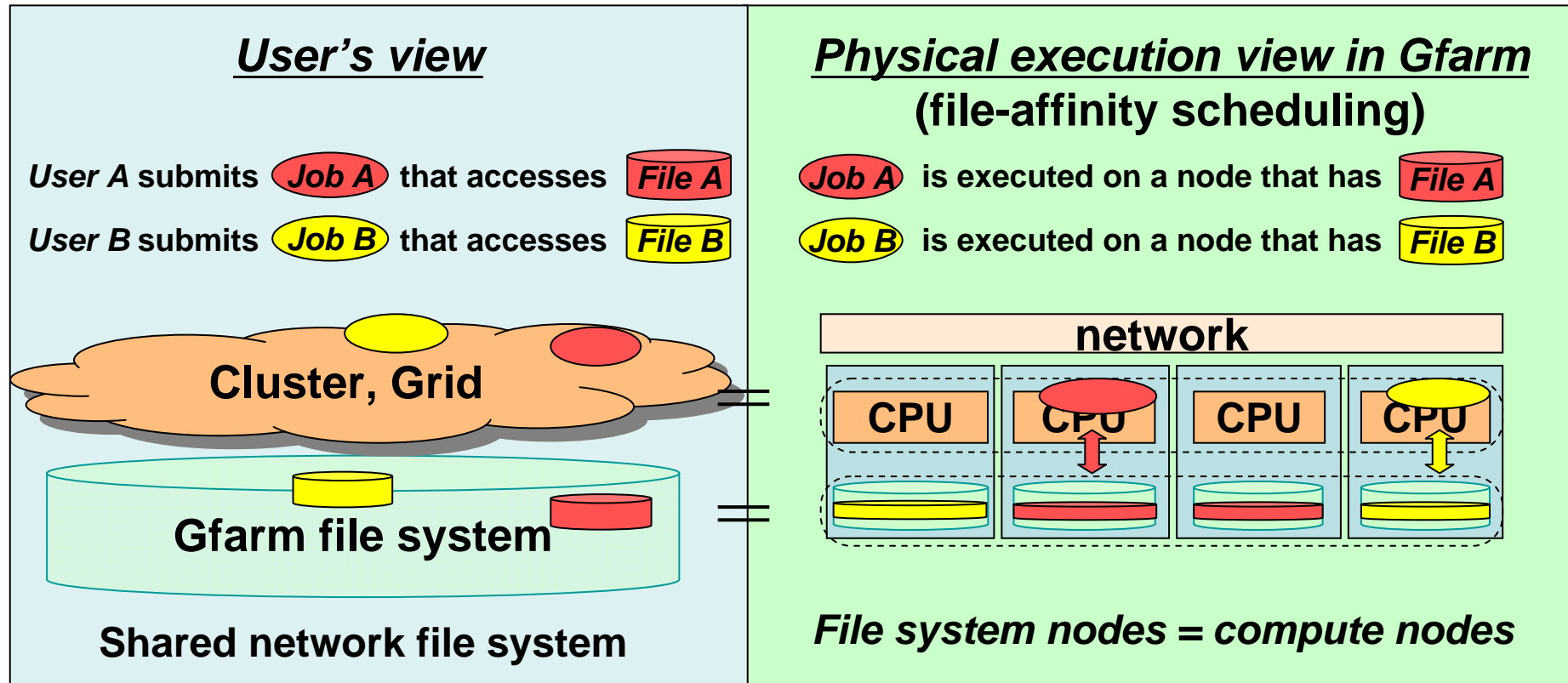
■ メタデータサーバ

- ▶ バックエンドDB (PostgreSQL または LDAP): ファイル, ノード情報を管理
- ▶ `gfmd`: ジョブ情報の管理

■ ファイルシステムノード: `gfsd`

- ▶ ファイルを格納しておくノードで動作

スケーラブル I/O の仕組み



CPU とストレージが分離されていない

データが格納されているノードでプログラムが起動される

並列化による高性能な I/O を実現

既存のアプリケーションから Gfarm FS を利用するには

■ システムコールフック: libgfs_hook.so

- ▶ open(2), read(2), write(2)などをフックして, /**gfarm** に Gfarm FS をマウントしているかのように見せる
 - Ⓜ /**gfarm** にアクセスすると, それに対応した Gfarm API が呼ばれる.
 - Ⓜ /**gfarm** 以外へのアクセスでは通常のシステムコールが呼ばれる.
- ▶ libgfs_hook を利用するために **LD_PRELOAD** を利用する.

■ ユーザ権限でのマウント

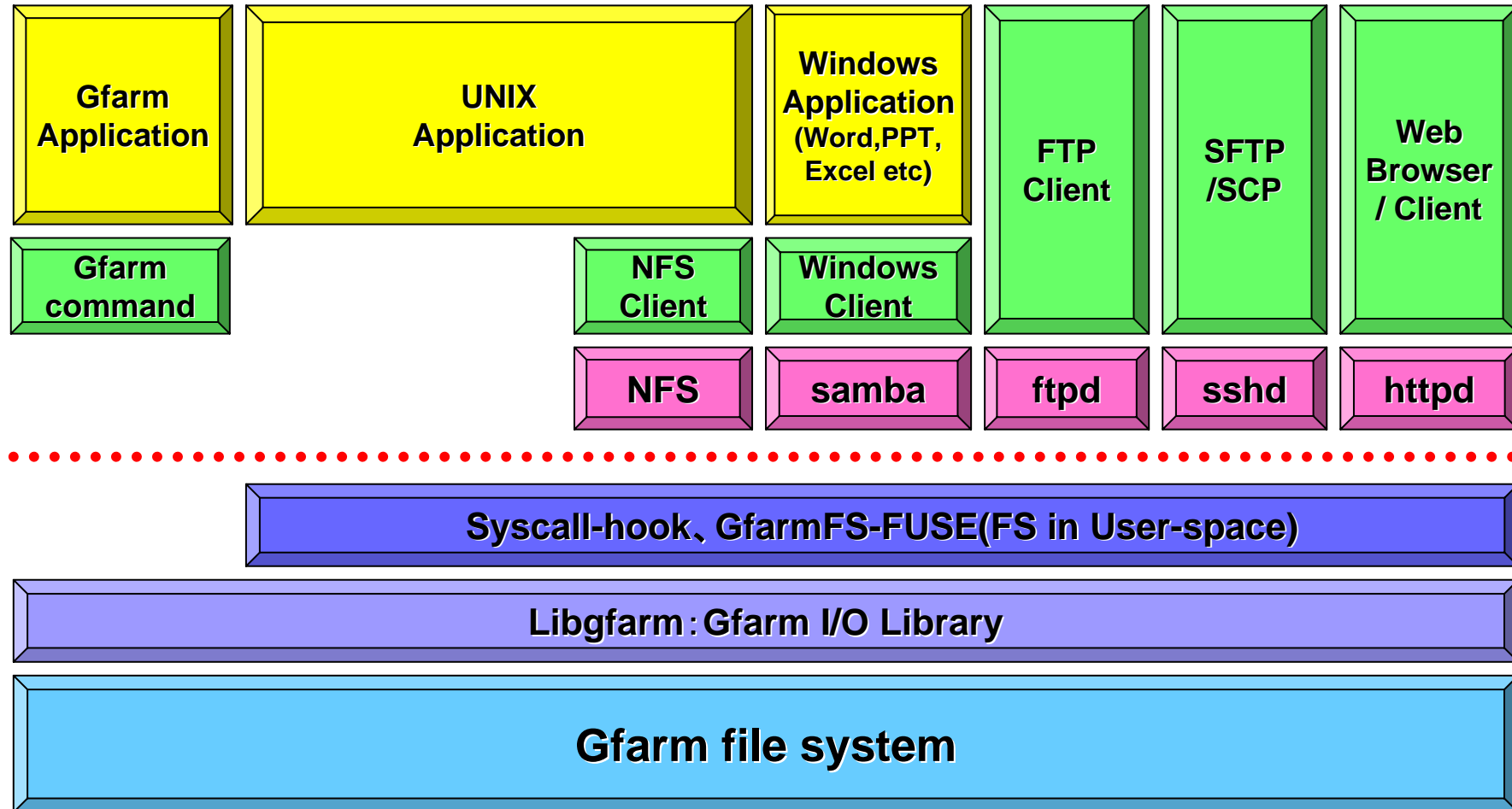
- ▶ Linux において, FUSE の機構を使って **GfarmFS-FUSE** で Gfarm FS をマウントする.
 - Ⓜ 利用できるシステムコール機能に制限がほとんどない
- ▶ Linux 以外では FreeBSD でも FUSE が動く



Ⓜ FreeBSD では GfarmFS-FUSE の動作確認中



Gfarm ファイルシステムの利用形態



Gfarm の動作環境（v1.4.1 リリース時）



■ 実アプリケーションレベルで Gfarm の動作を確認している環境

- ▶ RedHat 系の Linux

 - ⊗ Fedora Core 5-6

 - ⊗ CentOS 4

■ 動作実績は少ないが、簡単に Gfarm の動作テストを行っている環境

- ▶ Debian 系の Linux

- ▶ Solaris

- ▶ FreeBSD, NetBSD

実習における注意点

- 分からない点があれば挙手して下さい。
 - ▶ 講師＋サポートが対応します。

- 現在のユーザを確認しながら作業して下さい。
 - ▶ 一般ユーザ (griduserX), 管理ユーザ (root) のどちらで作業をしているのかを確認するようにして下さい。
 - Ⓢ 以降, X には各自に割り当てられた 0~27 の値を入れて下さい。

- 2種類のマシンを利用します。
 - ▶ インストール, セットアップの実習は会場の PC を利用します。
 - ▶ データ処理の実習は産総研のクラスタを利用します。
 - Ⓢ SSH でリモートログインを行います。
 - Ⓢ 実習終了後はアクセスはできなくなります。

インストールとセットアップ

1. 実習マシン@会場へのログイン
2. Gfarm 本体のインストールとセットアップ
3. GfarmFS-FUSE のインストール
4. 動作確認

Gfarm 本体のインストール (1)

■ 実習マシン@会場へログインします.

▶ griduserX でログインして下さい.

■ インストールするパッケージ(バイナリ)の確認をします.

```
$ cd /var/tmp/gfarm/gfarm
```

```
$ ls
```

```
gfarm-agent-1.4.1-0.i386.rpm
```

→ メタデータキャッシュサーバ

```
gfarm-doc-1.4.1-0.i386.rpm
```

→ man ページなどのドキュメント

```
gfarm-libs-1.4.1-0.i386.rpm
```

→ Gfarm ライブラリ

```
gfarm-client-1.4.1-0.i386.rpm
```

→ クライアントツール

```
gfarm-fsnode-1.4.1-0.i386.rpm
```

→ ファイルシステムサーバ

```
gfarm-server-1.4.1-0.i386.rpm
```

→ メタデータサーバ

```
gfarm-devel-1.4.1-0.i386.rpm
```

→ 開発用パッケージ(gfarmfs-fuse のビルドに必要)

```
gfarm-gfptool-1.4.1-0.i386.rpm
```

→ 並列データ処理のためのクライアントツール

Gfarm 本体のインストール (2)

■ パッケージ(バイナリ)をインストールします.

- ▶ root 権限で行う必要があります. 本実習では sudo を利用します.
- ▶ エラーが出なければ, 以下でインストールは完了です.

```
$ sudo rpm -ivh *
Preparing...      ##### [100%]
 1:gfarm-libs     ##### [ 13%]
 2:gfarm-client   ##### [ 25%]
 3:gfarm-agent    ##### [ 38%]
copy /etc/gfarm.conf from metadata server and
run /usr/bin/config-agent to configure Gfarm metadata cache server
 4:gfarm-devel    ##### [ 50%]
 5:gfarm-doc      ##### [ 63%]
 6:gfarm-fsnode   ##### [ 75%]
copy /etc/gfarm.conf from metadata (cache) server and
run /usr/bin/config-gfsd
 7:gfarm-gfptool  ##### [ 88%]
 8:gfarm-server   ##### [100%]
run /usr/bin/config-gfarm to configure Gfarm file system
```

Gfarm のセットアップ (1)

■ メタデータサーバをセットアップします。

▶ config-gfarm コマンドを利用します。

```
$ sudo config-gfarm -U griduserX -t → -t オプションで設定を確認
prefix [--prefix]:
metadata backend [-b]: postgresql → PostgreSQL をバックエンドに利用
(available backend: postgresql ldap)
metadata directory [-l]: /var/gfarm-pgsq
metadata log directory [-L]: /var/gfarm-pgsq/pg_xlog
postgresql schema version [-S]: 2
postgresql admin user [-U]: griduserX
postgresql admin password [-W]: (auto generated)
postgresql user [-u]: gfarm
postgresql password [-w]: (auto generated)
postgresql prefix [-P]: /usr
postgresql version [-V]: 7.4
metaserver hostname [-h]: compute-0-X.local
portmaster port [-p]: 10602
gfmd port [-m]: 601
gfsd port [-s]: 600
auth type [-a]: sharedsecret → 認証方式を sharedsecret に設定
$ sudo config-gfarm -U griduserX → セットアップを実行
```

Gfarm のセットアップ (2)

■ メタデータキャッシュサーバをセットアップします.

▶ config-agent コマンドを利用します.

```
$ sudo config-agent -t → -t オプションで設定を確認
```

```
$ sudo config-agent -t  
prefix [--prefix]:  
hostname [-h]: compute-0-X.local  
port [-p]: 603
```

```
$ sudo config-agent → セットアップを実行
```

Gfarm のセットアップ (3)

■ ファイルシステムサーバをセットアップします。

▶ config-gfsd コマンドを利用します。

② 引数でストレージ領域を指定することができます。

✦ 本実習ではディスク容量の大きい /state/partition1 を指定します。

✦ Gfarm では、ローカルデバイス(ディスク)の1つのパーティションを Gfarm 専用としてストレージ領域に割り当てるのが推奨されています。

▶ architecture は実行権限のあるファイルを Gfarm で共有する際に利用します。

```
$ sudo config-gfsd -t /state/partition1/gfarm → -t オプションで設定を確認
```

```
prefix [--prefix]:
```

```
hostname [-h]: compute-0-X.local
```

```
listen address [-l]: (all local IP addresses)
```

```
architecture [-a]: i386-rocks4.2.1-linux
```

```
ncpu [-n]: 1
```

```
pool directory : /state/partition1/gfarm → Gfarm のストレージ領域を指定
```

```
rc script name : /etc/init.d/gfsd
```

```
$ sudo config-gfsd /state/partition1/gfarm → セットアップを実行
```

Gfarm 本体の動作確認 (1)

■ Gfarm の設定ファイルを確認します。

▶ Gfarm の各種設定は /etc/gfarm.conf に書き込まれています。

Ⓜ gfarm.conf は gfmd や gfsd などのサーバプログラム, Gfarm のクライアントがデフォルトで参照します。

✦ サーバプログラムが参照するパラメータの値を変更した場合は, 該当サーバを再起動する必要があります。

```
$ cat /etc/gfarm.conf
spool_serverport 600
metadb_serverhost compute-0-X.local
metadb_serverport 601
postgresql_serverhost compute-0-X.local
postgresql_serverport 10602
postgresql_dbname gfarm
postgresql_user gfarm
postgresql_password "X0j69n*****yPFoCjgVZy0"
auth enable sharedsecret *

sockopt keepalive
agent_serverhost compute-0-X.local
agent_serverport 603
```

Gfarm 本体の動作確認 (2)

■ Gfarm のサーバプログラムが動作していることを確認します。

- ▶ PostgreSQL サーバ(postmaster), gfmd, gfarm_agent, gfsd

```
$ ps -ef | grep gfarm
```

```
griduserX 1441 1 0 14:20 pts/1 00:00:00 /usr/bin/postmaster -D /var/gfarm-pgsql
```

```
root 1459 1 0 14:20 ? 00:00:00 /usr/sbin/gfmd -P /var/run/gfmd.pid -f /etc/gfarm.conf
```

```
root 1638 1 0 14:37 ? 00:00:00 /usr/sbin/gfsd -P /var/run/gfsd.pid -f /etc/gfarm.conf -r  
/state/partition1/gfarm
```

```
griduserX 1711 1441 0 14:43 pts/1 00:00:00 postgres: gfarm gfarm 10.255.255.2XX idle
```

```
root 1712 1638 0 14:43 ? 00:00:00 /usr/sbin/gfsd -P /var/run/gfsd.pid -f /etc/gfarm.conf -r  
/state/partition1/gfarm
```

```
griduserX 1713 1441 0 14:43 pts/1 00:00:00 postgres: gfarm gfarm 10.255.255.2XX idle
```

```
root 1714 1 0 14:43 ? 00:00:00 /usr/bin/gfarm_agent -P /var/run/gfarm_agent.pid -f /etc/gfarm.conf
```

Gfarm 本体の動作確認 (3)

■ ファイルシステムサーバへアクセスができることを確認します。

▶ 認証手段

- Ⓧ x: 認証に失敗
- Ⓢ s: 共通鍵認証を利用
- ⓖ g: GSI 認証を利用
- Ⓜ G: GSI 認証および通信データの暗号化を利用

```
$ gfhost -lv
```

```
0.17/0.09/0.03 s i386-rocks4.1-linux 1 compute-0-X.local(10.255.255.2XX)
```

↑ CPU 負荷 ↑ 共通鍵認証を利用 ↑ アーキテクチャ名
 ↑ ↑ CPU 数 ↑ ファイルシステムノード名

■ gfmd へアクセスができることを確認します。

```
$ gfps -v ← エラーメッセージが出なければ OK
```

GfarmFS-FUSE のインストール (1)

■ FUSE の SRPM を用いて, FUSE をコンパイルします.

- ▶ FUSE は CentOS には標準パッケージとして含まれていません.
- ▶ 本実習で利用する SRPM は利用環境のカーネルに対応したカーネルモジュールを作成する必要があります.

```
$ cd /var/tmp/gfarm/gfarmfs-fuse
```

```
$ ls
```

```
fuse-2.6.5-2.src.rpm → FUSE のソースパッケージ
```

```
gfarmfs-fuse-2.0.0-1.centos4.i386.rpm → GfarmFS-FUSE のバイナリパッケージ
```

```
$ sudo rpm -ivh fuse-*
```

```
$ sudo rpmbuild -bb /usr/src/redhat/SPECS/fuse.spec
```

```
$ ls /usr/src/redhat/RPMS/i386/
```

```
fuse-2.6.5-2.i386.rpm → FUSE コマンド
```

```
fuse-libs-2.6.5-2.i386.rpm → FUSE ライブラリ
```

```
fuse-kernel-2.6.5-2.i386.rpm → FUSE カーネルモジュール
```

```
fuse-devel-2.6.5-2.i386.rpm → FUSE 開発用パッケージ
```

GfarmFS-FUSE のインストール (2)

- 作成した FUSE パッケージ(バイナリ)をインストールし, FUSE カーネルモジュールをロードします.

- ▶ root 以外のユーザが /dev/fuse を読み書きできるようにパーミッションが設定されている必要があります.

```
$ sudo rpm -ivh /usr/src/redhat/RPMS/i386/fuse-*
Preparing...      ##### [100%]
 1:fuse-libs      ##### [ 25%]
 2:fuse           ##### [ 50%]
 3:fuse-devel     ##### [ 75%]
 4:fuse-kernel    ##### [100%]

$ /sbin/lsmmod | grep fuse → FUSE のカーネルモジュールがロードされていることを確認
fuse                50708 0

$ ls -l /dev/fuse → others に対して書き込み権限が設定されていることを確認
crw-rw-rw- 1 root fuse 10, 229 Oct  4 15:49 /dev/fuse
```

GfarmFS-FUSE のインストール (3)



- GfarmFS-FUSE パッケージをインストールし, Gfarm ファイルシステムへのマウントポイントに用いるディレクトリを用意します.

```
$ sudo rpm -ivh gfarmfs-fuse-*
Preparing...      ##### [100%]
 1:gfarmfs-fuse   ##### [100%]
$ sudo mkdir -p /gfs/home
$ sudo chmod 1777 /gfs/home → Sticky bit を /gfs/home ディレクトリに設定しておく
$ ls -l /gfs
drwxrwxrwt  3 root root 4096 Oct  4 16:13 home
```

動作確認 (1)

- GfarmFS-FUSE を用いて Gfarm ファイルシステムにアクセスし, 作成した Gfarm 環境に問題がないかを確認します.
- 初めて Gfarm を利用する際に, ユーザが最初にやるべきことは次の2点です.
 - ▶ Gfarm コマンドへのパスが設定されているかを確認します.
 - Ⓜ RPM でデフォルトの場所に Gfarm をインストールしている場合は, Gfarm コマンドへのパスを特別に設定する必要はありません.
 - Ⓜ gfarm.conf は gfsd, gfarm_agent のセットアップで設定済みです.
 - ▶ Gfarm ファイルシステム上に, 自分 (griduserX) のホームディレクトリを作成します.

```
$ gfmkdir gfarm:~
```

動作確認 (2)

- 一般ユーザ (griduserX) で Gfarm ファイルシステムにアクセスします.

(FUSE の機構を用いて, Gfarm ファイルシステムをマウント)

```
$ mount.gfarmfs ← /gfs/home/griduserX にマウントされます
```

(Gfarm ファイルシステムに色々な形でアクセスしてみてください)

```
$ cd /gfs/home/griduserX
```

```
$ mkdir aaa ← ディレクトリの作成
```

```
$ emacs bbb ← エディタで新規ファイルを作成
```

```
$ cp /boot/vmlinuz-2.6.9-42.0.2.EL . ← linux のカーネルイメージをコピー
```

(Gfarm ファイルシステムをアンマウント)

```
$ cd ~
```

```
$ umount.gfarmfs
```

Gfarm を用いたデータ処理

1. 実習クラスター@産総研へのログイン
2. 基本コマンド
3. 並列データ処理

実習クラスタ@産総研へのログイン

- DHCP で IP アドレスが取れていますか？
- SSH を用いて sakura.hpcc.jp にログインして下さい。
 - ▶ ユーザ名：griduserX
 - ▶ パスワード：会場のマシンと同じ

Windows 環境にて SSH クライアントとして PuTTY を利用している場合は、ログインするホスト名を設定して SSH を選択し、Open ボタンをクリックして下さい。

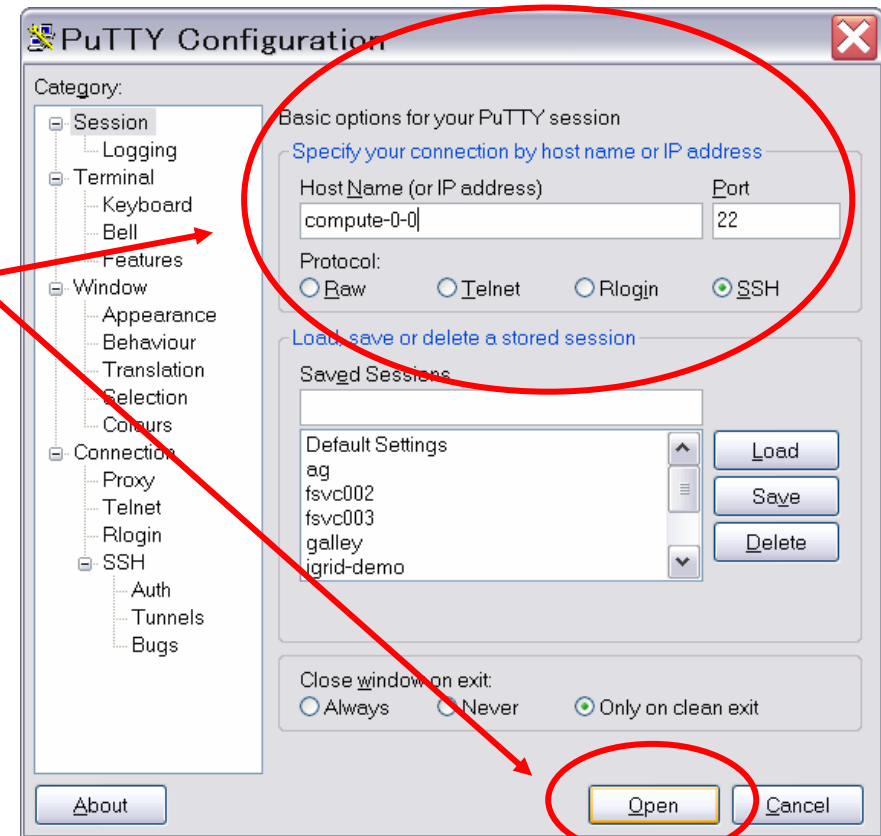
SAKURA クラスタ@つくば

sakura.hpcc.jp

- ログインノード
- メタデータサーバ (PostgreSQL, gfmd)
- メタデータキャッシュサーバ (gfarm_agent)
- ファイルシステムサーバ (gfsd)

sakuraXX.hpcc.jp

- ファイルシステムサーバ (gfsd)



GSI 認証を利用する場合 (1)

- **メタデータサーバ, ファイルシステムサーバでの設定において, GSI 関係のセットアップが必要になります.**
 - ▶ GSI 認証を行うためにホスト証明書を取得して, /etc/grid-security/
以下に配置します.
 - Ⓜ CA 証明書, signing_policy, (CRL), ホスト証明書とその秘密鍵
 - + 秘密鍵のファイルのパーミッションに気をつけること
 - Ⓜ grid-mapfile にユーザ毎のエントリを追加
- **ユーザ側はユーザ証明書を取得する必要があります. また, grid-proxy-init などのコマンド, Globus の共有ライブラリへのパスを環境変数に設定します.**

(環境変数の設定例: bash の場合)

```
export GLOBUS_LOCATION=/usr/gt4
if [ -r $GLOBUS_LOCATION/etc/globus-user-env.sh ] ; then
    . $GLOBUS_LOCATION/etc/globus-user-env.sh
fi
```

GSI 認証を利用する場合 (2)

■ Gfarm の利用開始時に有効なプロキシ証明書を作成します。

- ▶ プロキシ証明書は期限が切れるたび(デフォルトでは12時間)に更新する必要があります。

(プロキシ証明書の作成)

```
$ grid-proxy-init ← パスフレーズは配布資料を参照のこと
```

```
Your identity: /O=Grid/OU=GfarmTest/OU=hpcc.jp/CN=griduserX
```

```
Enter GRID pass phrase for this identity:
```

```
Creating proxy ..... Done
```

```
Your proxy is valid until: Tue Nov 1 03:26:36 200
```

(gghost コマンドで全ホストに対して GSI 認証に成功するかを確認)

```
$ gghost -lv
```

```
0.03/0.03/0.00 g x86_64-centos5-linux 2 sakura.hpcc.jp(163.220.27.104)
```

```
:
```

```
$ gfmkdir gfarm:~ ← 初めて使う時に必ず実行すること
```

(GfarmFS に色々な形でアクセスしてみてください)



主要なコマンド

■ Gfarm コマンドが必要になるケース

- ▶ GfarmFS-FUSE を使わないとき
- ▶ Gfarm 環境を管理するとき

■ ユーザ向けのよく使うコマンド

gfmkdir	gfrep	gfkey	gfrun
gfhost	gfis	gfstat	gfps
gfwhere	gfrm	gfwhoami	

■ 管理者向けのよく使うコマンド

gdf	gfusage	gfdump
-----	---------	--------

※ 本実習では全て紹介できませんが, man で使い方を確認して下さい.

基本コマンド：ファイルの所在確認

- **gfwhere** コマンドを用いて、ファイルが保存されているノードを知ることができます。
- GfarmFS-FUSE を使って ~/material/input.dat を Gfarm FS にコピーし、保存場所を見てみましょう。
 - ② input.dat は後で使う英単語ファイルです。各ユーザのホームディレクトリ以下にあらかじめ置かれています。
 - ② デフォルトではローカルノードが優先されます。

(sakura.hpcc.jp 上で作業します)

```
$ mount.gfarmfs
$ cp ~/material/input.dat /gfs/home/griduserX/
$ cd /gfs/home/griduserX
$ ls -l
total 1865
-rw-r--r--  1 griduserX griduserX 1909549 Oct 26 16:28 input.dat
$ gfwhere input.dat ← sakura.hpcc.jp に保存されていることが分かる
gfarm:input.dat:
0: sakura.hpcc.jp
```

基本コマンド: ファイルの複製

- **gfrep** コマンドを用いて、ファイルの複製を作ることができます。
- **input.dat** の複製を4つ作成してみましょう。
 - ▶ **-N** オプションで複製の個数を指定します。
 - Ⓢ ファイルシステムノード数以上は作成できません。
 - ▶ 元ファイルや複製先のノードを指定するオプションもあります。
 - Ⓢ ``man gfrep`` で確認して下さい。
- 複製を作成した後、**gfwhere** コマンドを使って、ファイルが保存されているノードを確認してみましょう。

```
$ gfrep -N 4 input.dat
```

```
$ gfwhere input.dat
```

```
gfarm:input.dat:
```

```
0: sakura0e.hpcc.jp sakura0f.hpcc.jp sakura0a.hpcc.jp sakura.hpcc.jp0:
```

```
$ ls -l /gfs/home/griduserX/input.dat ← ls では依然としてファイルは1つに見える
```

```
total 1865
```

```
-rw-r--r-- 1 griduserX griduserX 1909549 Oct 26 16:53 input.dat
```

基本コマンド：ジョブの起動

- **gfrun** コマンドは, Gfarm のスケジュール機能が付加された SSH コマンドです.
- オプションを付けずに **gfrun** を実行すると, CPU負荷の低いノードでコマンドが実行されます.
- “**-G <ファイルパス>**” オプションを付けて **gfrun** を実行すると, ファイルが保存されているノードでコマンドが実行されます.
 - ◎ この仕組みをファイル・アフィニティ・スケジューリングと呼びます.

(全体から最も負荷の低いホストで /bin/hostname が実行される)

```
$ gfrun /bin/hostname
```

(input.dat が保存されているノードで /bin/hostname が実行される)

```
$ gfrun -G input.dat /bin/hostname
```

(input.dat 以外のファイルも指定して実行してみてください)

基本コマンド：各サーバの容量確認

- **gdfd** コマンドを用いて、各ファイルシステムノードで Gfarm FS に割り当てられているパーティションの容量を確認することができます。
 - ② 実運用においては、Gfarm のファイルシステムノードがストレージ領域として利用するパーティションは、OS のシステム領域とは別に用意することをお勧めします。

```
$ gdfd
1K-blocks      Used      Avail      Capacity  Host
16253840      6945848   8469004    45%      sakura.hpcc.jp
12698092      2028464   10014176   17%      sakura00.hpcc.jp
:
```

並列データ処理のための機能

■ 科学技術アプリケーション向けのアドバンストな機能

■ 覚えておくべき概念

▶ ファイル・フラグメント

Ⓢ Gfarm では1つのファイルを断片に分けて保存可能

▶ ファイルビュー

Ⓢ アプリケーションに対するファイルの見せ方

▶ ファイル・アフィニティ・スケジューリング

Ⓢ ファイルの所在を考慮したスケジューリング機構

ファイル・フラグメント

- Gfarm では, 1つの大きなファイルを複数のファイル・フラグメントに分けて持つことができる.



- 各ファイル・フラグメントがいずれかのファイルシステムノードに保存される.



- ファイル・フラグメント毎に複製を作り, 別々に保存することができる.



ファイルビュー（1）

■ グローバル・ファイルビュー（デフォルト）

- ▶ ファイル・フラグメントに関係なく、1つのファイルとしてアクセスする時のビュー



■ インデックス・ファイルビュー

- ▶ 特定のフラグメントにだけアクセスする時のビュー

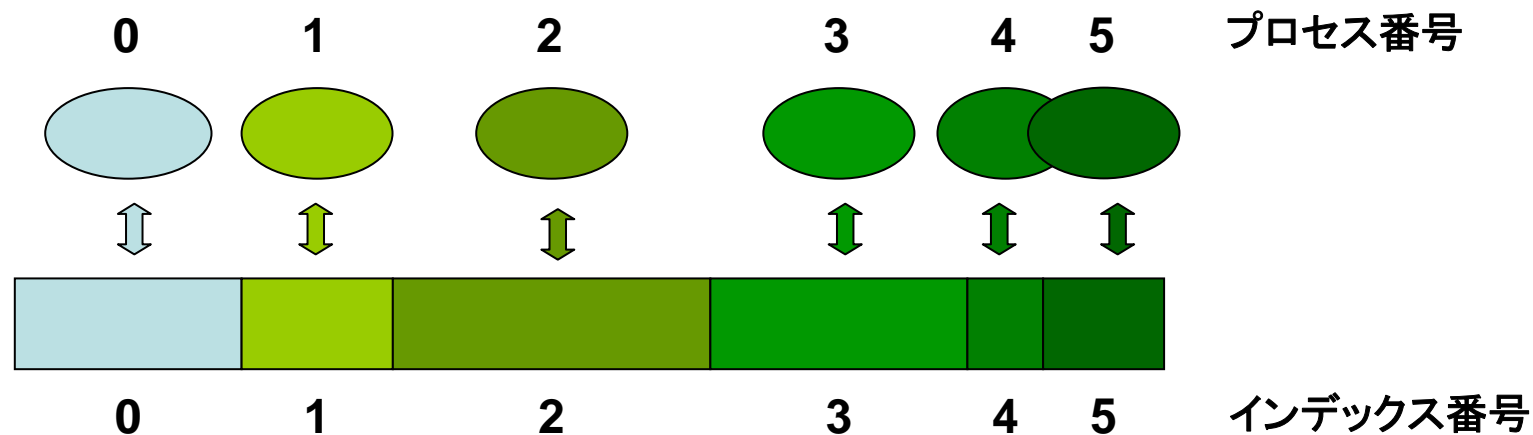


0 1 2 3 4 5 インデックス番号

ファイルビュー（2）

ローカル・ファイルビュー

- ▶ インデックス・ビューの特別なケース
- ▶ 並列に起動するプログラムのそれぞれが対応するファイル・フラグメントにアクセスする際のビュー
 - ⓐ スケーラブルな並列 I/O を得るために必須の機能



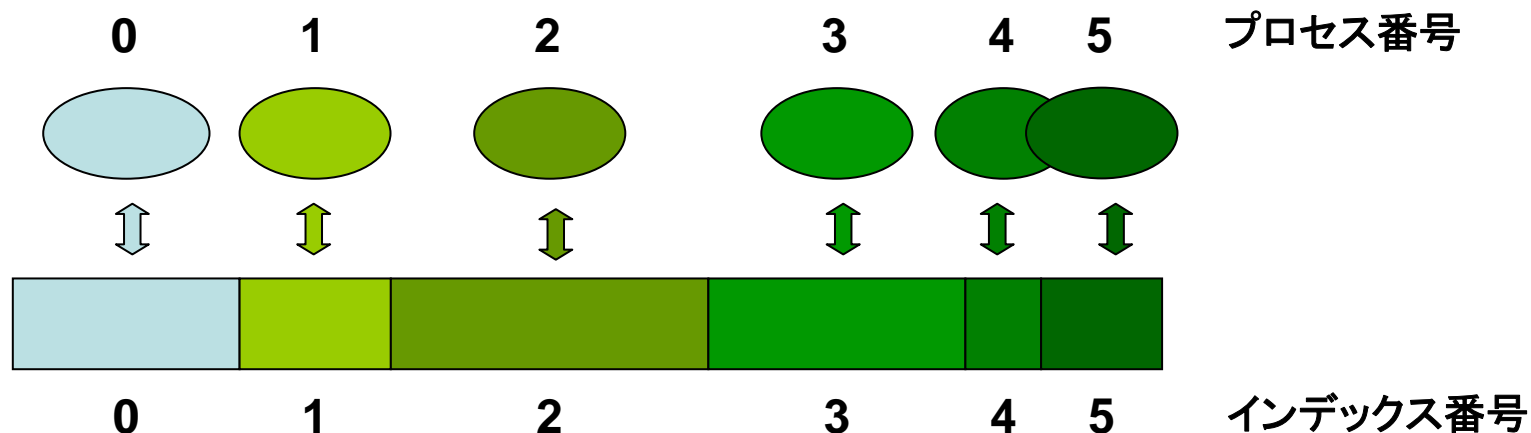
ファイル・アフィニティ・スケジューリング

■ ファイルの所在を考慮したジョブ・スケジューリング

- ▶ ファイル・フラグメント数と同じ数のプロセスを並列に実行する。1つのプロセスが1つのファイル・フラグメントに対応し、フラグメントが保存されているノードで起動される。

⊙ ファイル・フラグメントが1つであれば逐次処理になる。

- ▶ 起動されたプロセスはデフォルトではローカルビューになる。



並列データ処理をやってみよう (1)

- **gfptool** に含まれる **gfgrep** を使って並列 **grep** を行います.

- ④ **gfgrep**: Gfarm のローカルビューに対応した **grep**

- **input.dat** を4つのファイル・フラグメント(**input_fragment.dat**)に分割して **Gfarm FS** に保存します.

- ④ **gfarm:<ファイルパス>**: Gfarm FS が提供するグローバル名前空間

- ④ **gfsched**: ファイル・アフィニティ・スケジューリングによりノードリストを作成

- ④ **gfimport_text**: テキストファイルを行単位で分割して Gfarm FS に保存

```
$ gfimport_text -N 4 -o gfarm:input_fragment.dat input.dat
```

```
$ gfwhere input_fragment.dat ← 各フラグメントが4つのノードに分かれて保存されている
```

```
gfarm:input_fragment.dat:
```

```
0: sakura00.hpcc.jp
```

```
1: sakura01.hpcc.jp
```

```
2: sakura03.hpcc.jp
```

```
3: sakura04.hpcc.jp
```

```
$ diff input.dat /gfs/home/griduserX/input_fragment.dat
```

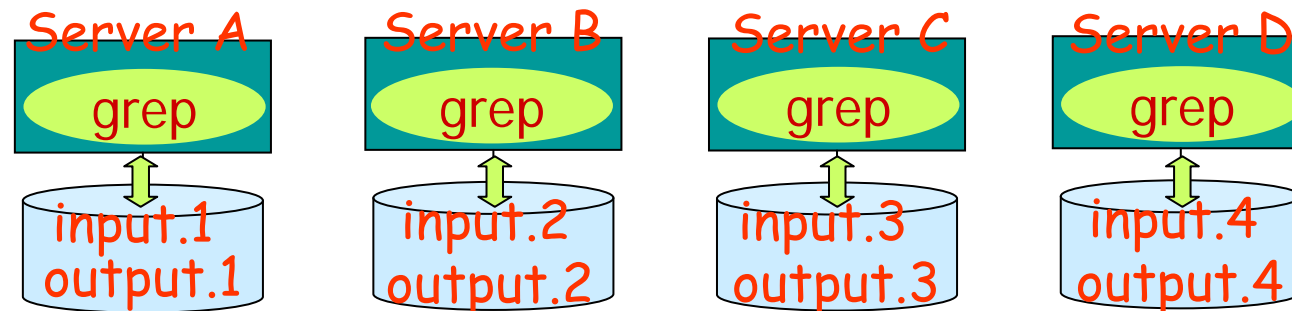
```
↑ グローバルビューでは同じファイルに見える(差分なし)
```

並列データ処理をやってみよう (2)

- input_fragment が保存されるノードで gfgrep コマンドを起動し、適当な英文字列 (例. food) を検索します。
 - ▶ ファイル・アフィニティ: input_fragment.dat
 - ▶ gfgrep の出力ファイル: gfarm:result.dat
 - ▶ gfgrep の入力ファイル: gfarm:input_fragment.dat

```
$ gfrun -G input_fragment.dat gfgrep -o gfarm:result.dat food gfarm:input_fragment.dat
$ grep food input.dat > output.dat ← ↓ 通常の検索結果と結果が同じことが確認できる
$ diff output.dat /gfs/home/griduserX/result.dat
```

- gfgrep の各プロセスの入出力が Gfarm ファイルシステムノードへのローカル I/O になり、**処理量が多い時に高速化可能**。



応用事例

科学技術分野

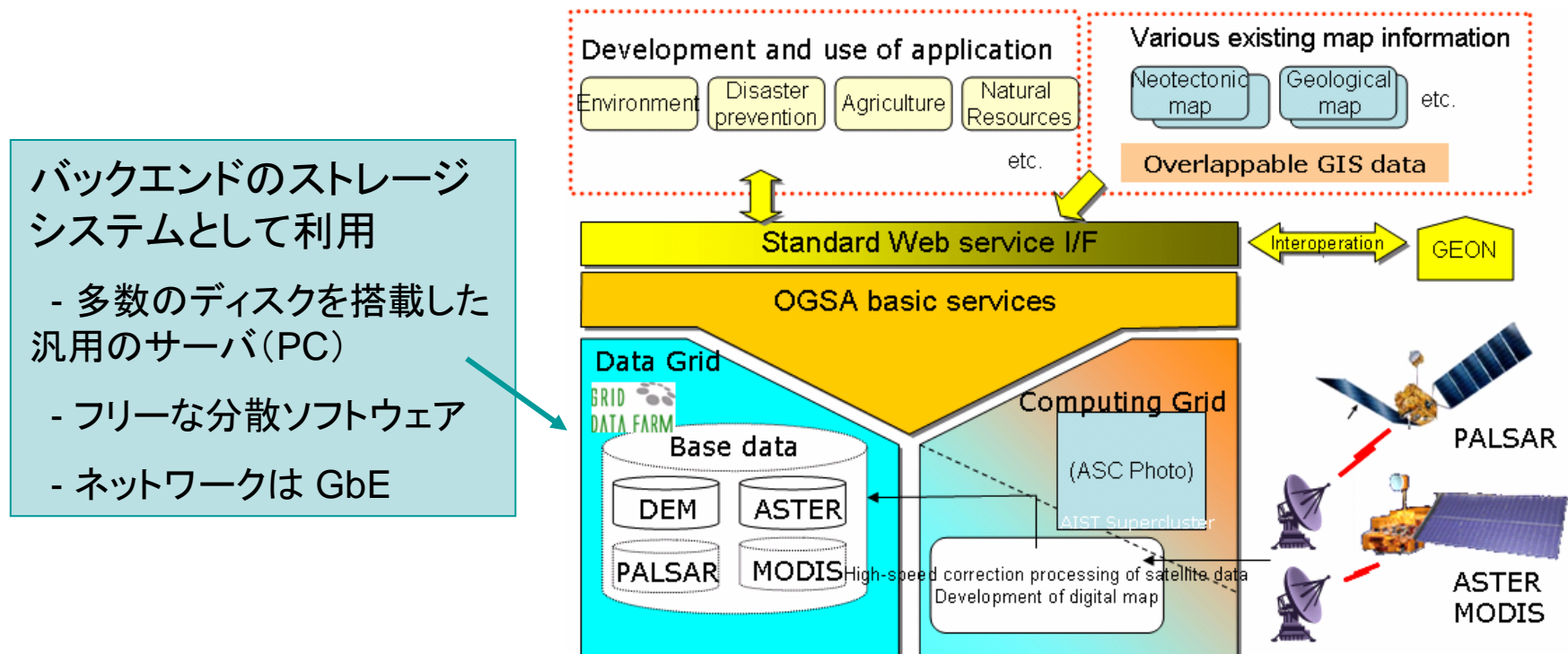
ビジネス分野

Scientific application: GEO Grid

■ GEO Grid (Global Earth Observation Grid)

http://www.gtrc.aist.go.jp/project_geo

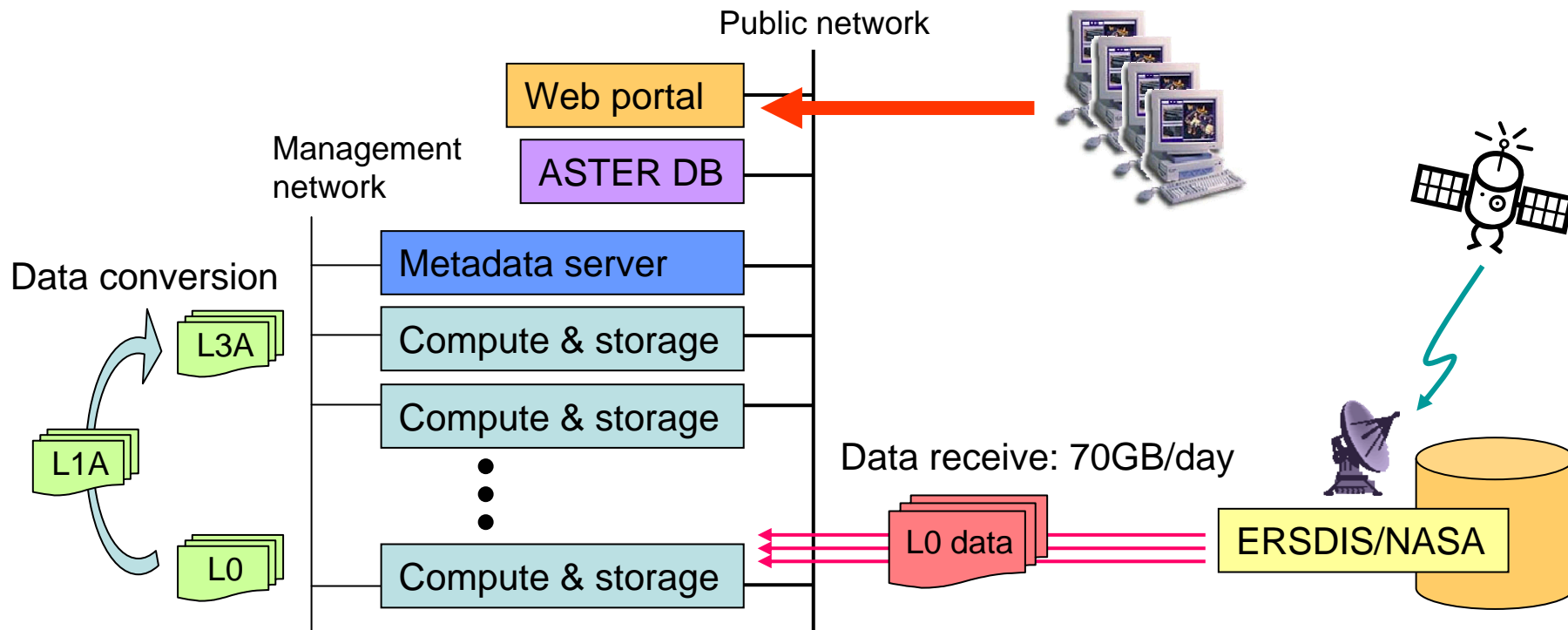
🌐 Gfarm is being utilized for storage infrastructure of GEO Grid.



Storage system for ASTER

■ 153 TB (15 millions' files) data has been stored.

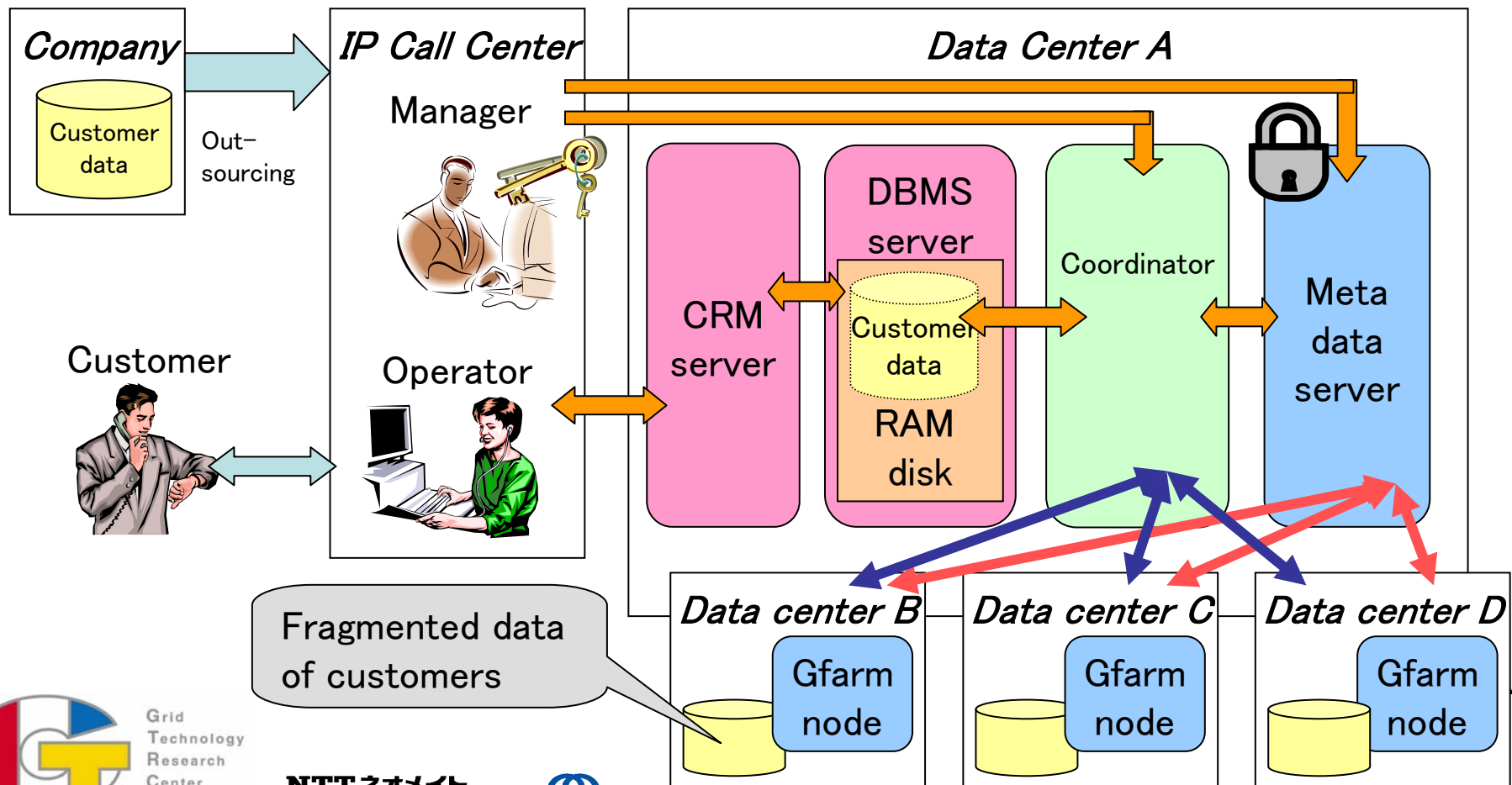
- ▶ **Gfarm metadata server** (PostgreSQL 8.2.1) + 4 cache servers
 - ⊗ Dual-core Opteron 2.6 GHz x 2, 16GB memory, CentOS 4.4
- ▶ **24 compute & storage servers**
 - ⊗ Xeon 3.8 GHz x 2, 4GB memory, 7TB disk (16-drive, RAID-6), CentOS 4.3



Business application: Secure database (1)



- Building a secure database for IP Call Center
- Research collaboration with the NTT Neomeit cooperation

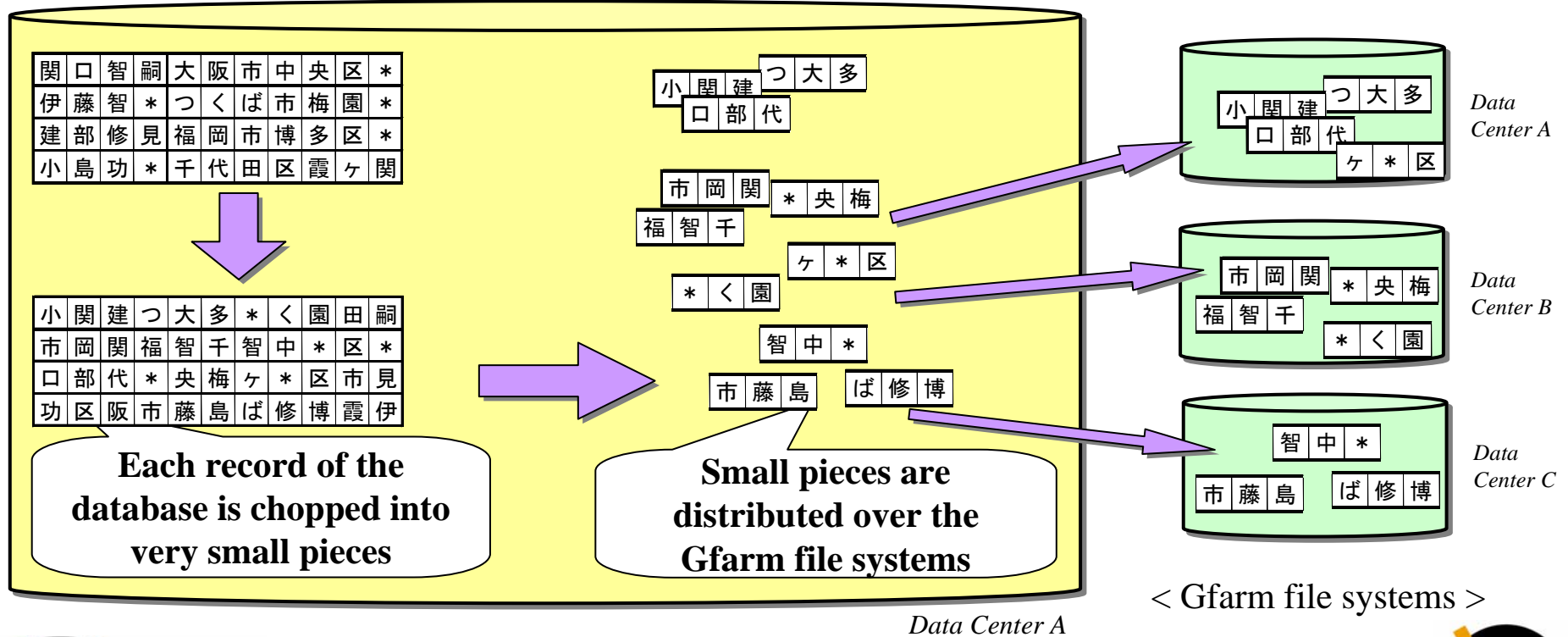




Business application: Secure database (2)

■ Implement scramble file view for security

- ▶ Data of the file is shuffled and chopped in storing into Gfarm file system.
- ▶ A decode key is stored on metadata server which is well protected.



Q & A

何でも聞いてみて下さい

おわりに

- Gfarm を用いて、簡単にネットワーク共有ファイルシステムを構築することができる

- 利用ケースは様々

- ▶ 大規模データ処理からクラスタファイルシステムまで
- ▶ GSI 認証を用いることでグリッド環境でも利用可能
- ▶ 産総研では GEO Grid で Gfarm を利用中

◎ http://www.gtrc.aist.go.jp/project_geo

- Gfarm のバージョン

- ▶ v1.4.1(安定板): メタデータアクセスの性能, v1 の基本設計に依存した問題がある
- ▶ v2(開発版): v1 の問題が解決されているが, 基本機能の実装を完了した状況であり, v1 の全機能が利用できるわけではない

Gfarm 関連情報

- Gfarm 本家のページ
 - ▶ <http://datafarm.apgrid.org/>
- SourceForge の Gfarm プロジェクトのページ
 - ▶ <http://sourceforge.net/projects/gfarm>
 - ▶ Subversion を利用して開発版(v2)をダウンロード可能
- ドキュメント
 - ▶ Web ページ <http://datafarm.apgrid.org/software/#doc>
 - ▶ Gfarm 配布版(ソースパッケージ, あるいは doc パッケージ)
 - ◎ INSTALL, INSTALL.RPM
 - ◎ GFARM_FAQ
 - ◎ KNOWN_PROBLEMS
 - ◎ 参照マニュアル, man ページ
- 質問, ご要望, バグ報告
 - ▶ 日本語メーリングリスト gfarm-discuss-ja@apgrid.org
 - ▶ 日本語バグトラッキングシステム
<http://datafarm.apgrid.org/bugzilla-ja/>