高性能広域計算基盤 Grid へのポータルシステムの設計と実装

鈴村 豊太郎[†] 松岡 聡[†] 中田 秀基[†,††]

本稿では，分散オブジェクト技術 Jini を基盤に構築した Computing Portal システム JiPANG(Jini-based Portals Augment Grid) の概要を示す．JiPANG は，Grid 上に任意に存在するサービスを一元的に管理するインフラストラクチャと，そこに登録されたサービスの統一的かつ透過的な使用を可能にするプログラミング環境を提供する．JiPANG を用いることによって，Ninf や NetSolve, Globus 等の多様な Grid 技術に同一のライブラリからアクセスできる他，最新のバージョンのサービスが自動的にダウンロードされる等，Grid 環境で動作するアプリケーション開発のコストを大幅に削減することができる．

Design and Implementation of a Jini-based Computing Portal System

Toyotaro Suzumura,[†] Satoshi Matsuoka[†]
and Hidemoto Nakada[†,††]

JiPANG(Jini-based Portal Augmenting Grids) is a portal system and a toolkit which provides uniform access interface layer to a variety of Grid systems, and is built on top of Jini distributed object technology. JiPANG performs uniform higher-level management of the computing services and resources being managed by individual Grid systems such as Ninf, NetSolve, Globus, etc. In order to give the user a uniform interface to the Grids JiPANG provides a set of simple Java APIs called the JiPANG Toolkits, and furthermore, allows the user to interact with Grid systems, again in a uniform way, using the JiPANG Browser application. With JiPANG, users need not install any client packages beforehand to interact with Grid systems, nor be concerned about updating to the latest version. Such uniform, transparent services available in a ubiquitous manner we believe is essential for the success of Grid as a viable computing platform for the next generation.

1. はじめに

Web はネットワーク上の情報資源の共有を可能とし，Yahoo 等の ポータルサイトによって爆発的な進化を遂げた．これは，検索サービスやディレクトリサービス等の統一的かつ直観的なインタフェースをユーザに提供し，大規模な情報の集合体から構成される Web の複雑さを隠蔽したことが成功につながったと言える．

そして近年，ネットワーク技術の目覚しい進化に伴って，Web が達成した情報資源の共有ばかりでなく，広域ネットワーク上に分散したストレージや遠隔の物理的なデバイスを含めた計算資源の共有が可能となり，それらの資源を活用して科学技術計算を行う，いわゆる Grid (高性能広域計算) 技術の研究が盛んに行われている．代表的な例として，我が国の Ninf[17], 米国の NetSolve[9], Globus[11] 等が挙げられる．

今後，これらの Grid 技術が実際に運用段階に入り，研究者からエンドユーザまで幅広い層に使用されるようになる為には，ポータルサイトが達成したように，透過的かつ簡便なインタフェースをユーザに提供することが不可欠である．近年，これらを目的とする，いわゆる Computing Portal システムの研究が活発化してきている．Grid の標準化団体である Global Grid Forum[1] の Grid Computing Environment Working Group[2] において，盛んに議論が行われており，CoG[12], HotPage/GridPort[13] 等のシステムが開発されている．しかし，これらの既存のシステムでは，特定の Grid 技術や資源に特化したインタフェースのみを提供しており，Computing Portal システムが満たすべき 一般性，統一性に欠ける．

Web は Grid のように，そのコンテンツの複雑さが恒常的に増加し，動的に変化する性質を持つが，それらに柔軟に対処するソフトウエアのインフラストラクチャを用意することで，その問題を解決している．具体的には，クライアント側に HTML/XML, JavaScript, Java 技術を使用し，サーバ側には，CGI 技術, Java Servlet, コンポーネント技術を使用している．では，Computing Portal が，Grid 環境上に存在する様々なソフトウエア基盤に対して，透過性，統一性，そし

† 東京工業大学
†† 産業技術総合研究所

て直観的なインタフェースを提供する為には，どのようなソフトウエア技術が必要であろうか．

本研究では，Sun Microsystems が提案する分散オブジェクト技術 Jini を基盤に構築した Computing Portal システム JiPANG の設計，実装を行った．JiPANG は，Grid 上に任意に存在するサービスを一元的に管理する基盤と，そこに登録されたサービスの統一かつ透過的な使用を可能にするプログラミング環境を提供する．

以下，2章では Jini 技術の概要を示し，Computing Portal のソフトウエア技術としての可能性について議論する．3章では JiPANG システムの概要，4章では，JiPANG が提供するライブラリ，5 章では JiPANG から利用可能な Grid サービスの例を解説する．6 章は関連研究，7 章はまとめである．

2. Jini 技 術

分散オブジェクト技術 Jini[4] は、分散システムに必要な耐故障性やセキュリティ等の機能を提供し、Grid 環境のように動的な環境において，有用なソフトウエア技術として近年注目を浴びている．以下にその概要と Computing Portal システムのソフトウエア技術としての可能性について議論する．

2.1 概　　要

Jini は、シンプルで柔軟性が高く、一体化したネットワークを目指して設計された Java ベースの分散システムである．Jini アーキテクチャでは、全てのデバイス及びソフトウエア、そしてユーザをサービスとして扱い、それらをグループ化することで、互いのリソースを必要に応じて利用し合える環境を実現する。Jini では，このサービスの集合を”連合体 (federation)”と呼ぶ．

Jini 技術は，Lookup サービスを発見して即座に連合体に参加するプロトコル (Discovery と Join), クライアントとサービスの交換機の役割を果たす Lookup サービス (以下，Jini LUS と呼ぶ)，基盤となるオブジェクト・モデルを提供する Java 言語，そして連合体を提供してオブジェクトを移動させる RMI 技術から構成される．

サービスの登録は，Jini LUS にリースの期間を要求して行われる．リースが期限前に更新されなければ，Jini LUS から削除される．よって，マシンやネットワークの故障によって，サービスが稼働状態でない時にはリースの更新がされず，自動的に連合体から削除される．この他に，分散システムに必要な分散イベント，分散トランザクション，Java 2 に基づいたセキュリティ機構等の機能を提供する．

2.2 Grid 環境への適用

Jini 技術は，Grid のように動的に構成が変化し，常に故障を前提としなければならない環境上にロバストなシステムを構築するツールとして，有用な性質を備えているといえる．しかし，以下に挙げる点で，Grid への直接の適用は不適当である．

拡　張　性

Jini はサービスの動的な発見にマルチキャストを用い，LAN 等の比較的範囲の狭いネットワーク環境を想定している．よって，Grid のように広域環境では，拡張性が損なわれる．Jini の枠組のみを利用して，これを解決する方法として，以下の 2 つ方法が考えられる．

- 複数の機関のマルチキャストをトンネリングする Jini サービスを用いて，別ネットワーク上の Jini サービスの動的な発見を可能にする方法 [8]．
- Jini LUS(LUS) は，それ自体が Jini サービスであるので，他の LUS に登録することが可能である．これを利用し，LUS 間で階層関係を作ることで，任意の LUS から 任意の LUS へのサービスを参照する方法．

いずれの方法も Grid の環境では機能しないことが容易に予想される．特に後者では，他の Jini LUS への接続のコストと，更にその LUS へのサービスの検索のコストがかかり，無数に存在する LUS を巡回するのは極めてコストがかかる．

サービス検索

Jini のサービス検索では，サービスのインタフェース名, プロキシのクラス名，関連のある属性情報を指定して行われる．また，直接，サービスの ID を指定することも可能である．しかし，Jini は，指定したものと完全に合致するサービスのみを返し，ユーザがあらかじめそのサービスの正確な情報を保持していなければならない．例えば，ユーザが ”NinfSolve” という名のサービスを検索する時に，”Ninf*” や ”*solve” 等の曖昧な名前を与えて検索することができない．

しかし，多様なサービスが存在する Grid 環境においては，現在の Web の検索サービスのように，未知の情報でも柔軟に探索する機構が必要である．

ツールサポート

Grid サービスの提供者がそのサービスを Jini サービスとする為には，ある程度の Jini の知識を必要とする．より多くの提供者がその Grid サービスを提供する為には，Jini 化の工程を簡便化する支援ツールが必要である．

3. JiPANG システム

前章では，Jini 技術が Computing Portal システムを構築するソフトウエア技術として十分な機能を提供するが，直接，広域環境に適用するにはいくつかの問題点が生じることを述べた．本章では，Jini 技術を拡張した Computing Portal システム JiPANG の概要とそのアーキテクチャ構成を述べる．

3.1 概　　要

JiPANG[5][18] システムは，Jini を基盤に構築した

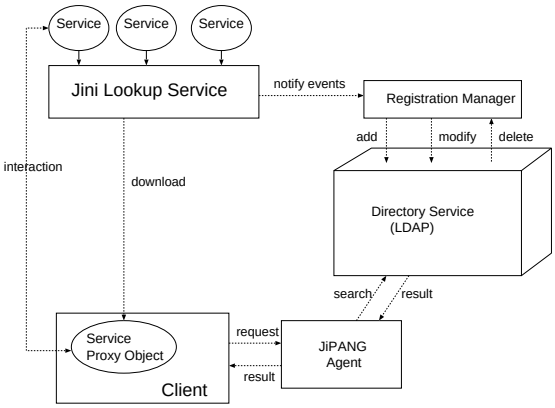

図 1　アーキテクチャ構成

Computing Portal システムである．JiPANG では，Grid 環境上のすべての計算資源, システム，ライブラリを Jini サービスとして抽象化し，Jini を広域環境に拡張したインフラストラクチャを用いて一元的に管理する．このインフラストラクチャにより，耐故障性のあるダイナミックな Grid サービスの連合体が形成される．

JiPANG のインフラストラクチャに登録されたサービスには，JiPANG が提供するプログラミング環境 JiPANG Toolkit を用いてアクセスすることができる．このライブラリを用いることによって，柔軟かつ高速に必要なサービスを検索することができる他，様々な Grid サービスを統一的かつ透過的に使用することができる．

3.2　アーキテクチャ

JiPANG システムは以下のコンポーネントから構成される (図 1).

- JiPANG (Grid) サービス
- Jini Lookup サービス
- ディレクトリサービス
- 登録マネージャ
- エージェント
- クライアント

3.2.1 JiPANG (Grid) サービス / Jini Lookup サービス

JiPANG サービスは，Grid 上のサービスを Jini サービスとして抽象化したコンポーネントである．Grid 環境を構成するサービス (計算資源, システム，ライブラリ等) をすべて Jini サービスとして捉えることで，Grid サービスの連合体が形成され，Jini 技術が提供する様々な機能を使用することができる．

Jini Lookup サービス (Jini LUS) はこれらの JiPANG サービスを実際に管理するコンポーネントである．各 Jini LUS は機関毎で稼働し，その機関内の Grid サービスを管理, 各サービスに 128 bit の ID を割り当てる．この ID は，Jini LUS のホストアドレス，サービスを生成した時間，ランダムな値の組合せにより生成される一意な値である．

JiPANG サービスの耐故障性は，サービスの登録を行うスレッドによって確保される．このスレッドは，サービスの故障を検知する他，Jini LUS へのリースの更新を行う．マシンやネットワーク，サービスの異常があった場合には，即時に，このリースの取消しを Jini LUS に告げ，Jini の連合体からそのサービスを削除する．よって，JiPANG へ登録されたサービスはすべて稼働状態である．

また，サービス提供者は，サービスプロキシを Java 言語で記述する必要がある．サービスプロキシは，クライアントに提供するメソッドを規定した Java のインタフェースを実装しており，サービスと実際に通信を行う Java オブジェクトである．プロキシとサービスの通信には，専用プロトコル又は RMI が用いられ，この過程は，サービスのプロキシ内にカプセル化される．

一方，JiPANG サービスは，最新のバージョンのサービスプロキシとサービスの属性情報を付加して Jini LUS に登録される．属性情報としては，サービスの提供者の情報，位置情報等が含まれ，サービスのフロントエンドを担う GUI を付加することも可能である．この GUI の属性の仕様は，jini.org が主催する ServiceUI プロジェクト [7] の仕様を用いた．

最後に，Jini LUS はエージェントからの要求があると，RMI の Dynamic Class Loading の機構を用いて，クライアントの Java 仮想マシンにサービスプロキシをダウンロードする．

3.2.2　ディレクトリサービス

ディレクトリサービスは，任意の機関に独立に存在する JiPANG サービスの連合体を集約するコンポーネントである．実際に JiPANG サービスを管理するコンポーネントは各機関の Jini LUS であるが，そのメタ情報を一元的に管理する役割を持つ．ディレクトリサービスには LDAP を用いた．

各 Jini LUS の識別子 (Distinguished Name) は，ホスト名に基づいて決定される．例えば，ホスト名 "*uva.is.titech.ac.jp*" で稼働する Jini LUS は，"*lookup=uva-lus, dc=uva, dc=is,dc=titech, dc=ac, dc=jp*" となる．また，JiPANG サービスのメタデータは，登録された Jini LUS の子ノードに格納される．メタデータは，サービス名，サービス ID，インタフェース情報等から構成され，サービスのリモート参照として使われる．ディレクトリサービス内の階層構造は, 図 2 のようである．

ディレクトリサービスを導入することにより，Grid 環境上に広域に分散した Jini の連合体を間接的に統合し，拡張性が確保される他，以下にあげる利点がある．

- 探索の柔軟性/高速化

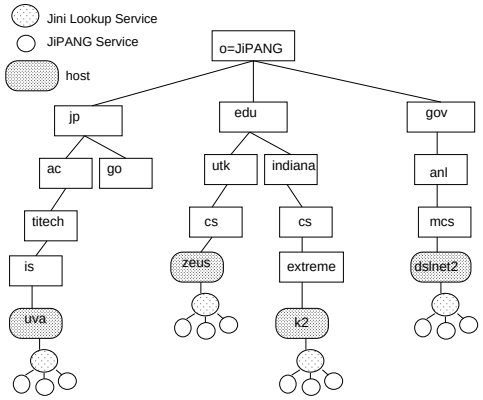

図 2　ディレクトリサービス内の木構造

LDAP の柔軟で高速な検索方法 (正規表現等) を用いることによって，多様な JiPANG を柔軟に探索することが可能．

個々のサービスはホスト名に基づいて階層構造に格納されるので，データベース中に，ある程度のネットワーク的な近さを反映することができる．これによって，クライアントは容易に探索の範囲を狭めることが可能になる．

- 他の Grid システムのデータベースとの親和性
  近年，資源やネットワーク等の情報を格納するデータベースとして LDAP が主流となりつつあり．代表的な例として，Globus の MDS[16]，Ninf が挙げられる．LDAP には 参照機能 (referral) があるので，JiPANG の LDAP からこれらの LDAP サーバへ参照することが可能になる．これによって，他の Grid サービスの資源情報を直接管理する必要がなくなり，システム自体の軽量化を図ることができる．

3.2.3　登録マネージャ

登録マネージャは，Jini LUS と同様のホストで稼働し，Jini LUS からのイベントの通知を待機する．イベントには，JiPANG サービスの追加，削除，属性の変更等が情報が含まれる．追加の通知があった場合には，イベントを起こしたサービスのメタ情報を取得し，LDAP のフォーマットに変換，適切な場所に格納する．削除の通知の場合には，LDAP 中のエントリを削除する．

3.2.4　エージェント

エージェントは，クライアントの要求に応じてディレクトリサービスに問い合わせ，適切なサービスを返す．クライアントは，LDAP と同じクエリーフォーマットである．エージェントは，ディレクトリサービスから選択されたサービスのメタ情報を抽出し，クライアント側にプロキシオブジェクトを渡す．このコンポーネントは，汎用的な目的に使用されることを想定しているが，GridRPC システム [3] に特化したエージェントに置換することも可能である．

4.　JiPANG Toolkit

JiPANG Toolkit は，JiPANG サービスに統一的，透過的にアクセスするためのプログラミング環境 であり，Service Toolkit, Client Toolkit, JiPANG Browser から構成される．

4.1　Service Toolkit

Service Toolkit は，Grid のサービス提供者が Jini 技術の知識なしに，サービスの登録を可能にするいくつかのコマンドユーティリティを提供する．その一つ *jipang_register* コマンドは，XML によって記述した設定ファイルを引数にして，Grid サービスの JiPANG への登録を自動化する．この設定ファイルには，サービス提供者や Java インタフェースの情報等が含まれる．以下は, Ninf サービスの設定ファイルの例である．

```
<JipangService>
<lookup host="uva.is.titech.ac.jp" port="4160" />
<proxy> <!- Proxy class Info ->
<class>org.jipang.grid.ninf.NinfProxy</class>
<param type="java.lang.String">uva.is.titech.ac.jp</param>
<param type="java.lang.String">3030</param>
</proxy>
<serviceInfo> <!- ServiceInfo ->
<name>Ninf</name>
....
....
<guiInfo> <!- Service GUI Information ->
<name>NinfBrowser</name>
<role>net.jini.lookup.ui.MainUI</role>
<toolkit>java.awt</toolkit>
<type>Frame</type>
<factory>org.jipang.grid.ninf.NinfUIFact</factory>
</guiInfo>
</JipangService>
```

4.2　Client Toolkit

Client Toolkit は，JiPANG 内に登録されたサービスを統一的，透過的に使用するための Java ライブラリである．このツールキットによって，異種の Grid サービスを同一のライブラリから使用できるようになる他，ある特定のアプリケーションに特化した Science Portal を容易に構築することが可能になる．ユーザは，あらかじめサービスが存在する場所を意識することも，最新のクライアントパッケージをインストールする必要もない．以下に，提供されるいくつかのメソッドを示す．

```
public JipangServiceInfo[] searchService(String service, String
                filter, String baseDN) throws JipangException ;
public Object getProxy(String dn) throws JipangException;
```

初めのメソッドは引数に，サービス名，フィルタリングの条件 (LDAP フォーマットに従ったフォーマット)， 探索を開始するノードの識別子を指定する．これらのメソッドは，LDAP サーバ中にクエリを送り，条件に適合するサービスを抽出する過程を隠蔽する．二番目のメソッドは，引数に指定された識別子と一致するサービスのプロキシオブジェクトを取得する．

クライアントのインタフェースとしては，Java 言

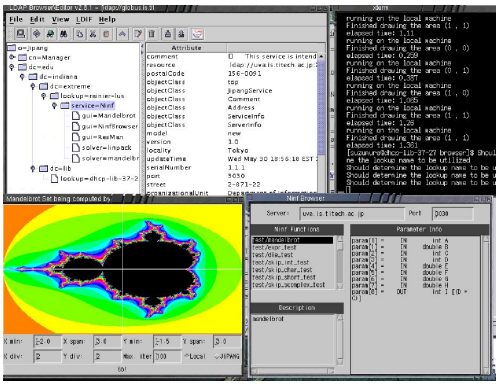

図 3　JiPANG Browser

語の他に, スクリプト言語 Python ⋆がサポートされている.

4.3　JiPANG Browser

JiPANG Browser は，JiPANG に登録されたサービスを検索，閲覧する為の GUI アプリケーションである．ユーザは，Browser を通して柔軟かつ高速にサービスの検索ができる他，GUI のフロントエンドを持つ JiPANG サービスを直接実行することができる．図 3は, JiPANG Browser から Ninf サービスの GUI フロントエンド Ninf Browser⋆⋆と，デモアプリケーションを実行している様子である．

実装は，CoG プロジェクトの LDAP Browser を基盤に開発した．

5.　JiPANG サービス

我々は，プロトタイプとして，いくつかの JiPANG サービスを構築した．本稿では，GridRPC サービス，測定サービス，Globus サービスを紹介する．

5.1　GridRPC

GridRPC とは，遠隔計算ライブラリを提供し，Grid 環境上でタスク並列のプログラミングモデルを提供するシステムである．Ninf や NetSolve 等がその代表的なシステムである．

我々はこれらのサービスを提供するシステム間の相違を吸収し，同一のプログラムからのアクセスを可能にする Java インタフェースを下記のように定義した．現在，Ninf, NetSolve, Java RMI (RMI によって 遠隔の 計算ルーチン Java オブジェクトを実行) のサービスが実装されている．

```
public interface GridRPCInterface extends JipangInterface −
  public void setArg(Vector vec) throws JipangException;
  public JipangExecInfo jipangCall(String func)
                                  throws JipangException;
„
```

以下に，上記のインタフェースの使用例を示す．

```
public class GridRPCTest−
  public static void main(String[] args) −
  ....
  GridRPCClient client = new GridRPCClient();
  client.setArg(max);
  client.setArg(mag);
  client.setArg(xsize);
  client.setArg(ysize);
  client.setArg(left);
  client.setArg(right);
  client.setArg(top);
  client.setArg(bottom);
  client.setArg(output);
  JipangExecInfo execInfo = client.jipangCall("mandelbrot");
  System.out.println(execInfo.toString());
  ...
„
```

5.2　測定サービス (NWS)

測定サービスとは，Grid 環境上で 動的に変化するネットワーク資源や計算資源の状態の予測を行うサービスである．以下にそのインタフェースを示す．現在，NWS[15] がこのサービスの役割を果たしているが，GridRPC サービスと同じく，今後，このインタフェースを実装した JiPANG サービスがあれば，それらを統一的に使用することができる．

```
public interface ForecastInterface extends JipangInterface −
  public String getNameServer(String nameWithPort)
                        throws JipangException;
  public String[] getForecasts(String sourceMachine,
      String destinationMachine,String experimentName,
      int atMost) throws JipangException;
  public String[] getMeasurements(String sourceMachine,
     String destinationMachine,
     String experimentName,int count, double sinceWhen)
              throws JipangException;
...
„
```

5.3　Globus サービス

Java CoG Kit は，Globus の各種サービスに対する Java インタフェースである．我々は，この CoG パッケージを用いて，Globus の JiPANG サービスを実装した．Globus サービスが JiPANG から利用可能になることにより，Globus ユーザは，Globus ソフトウエアを持つ必要がなくなり，軽量のクライアントプログラムを構築することができる．現在，GRAM, MDS, GASS 等，低レベルの Globus サービスを提供している．これらの API は，Globus が提供するインタフェースとほぼ等価なインタフェースであり，CoG を用いて構築された既存の Grid アプリケーションを容易に，JiPANG へ移行することができる．以下に，GRAM サービスの インタフェースの一部を示す．

```
public interface GlobusGramInterface extends JipangInterface −
  public void gramPing(String resourceManagerContact);
  public void gramRequest(String resourceManagerContact,
                  String rsl);
  public  void gramCancel(JipangGramJob job);
  public  void gramJobStatus(JipangGramJob job);
  public  int gramJobSignal(JipangGramJob job,
                  int signal, String arg);
  ....
„
```

⋆ Java によって実装された Python, Jython[6] を用いる
⋆⋆ Ninf サーバに登録された計算ルーチンを閲覧するアプリケーション

6. 関 連 研 究

WebFlow[10] は，Web ベースの Computing Portal システムである．一連の HTTP サーバが Web サーバとして機能する他，CGI 技術を用いることによって，バックエンドの計算サーバへのプロキシの役目を果たす．WebFlow では，ユーザが既存の Grid サービスに対する CGI インタフェースを書かなければならないのが難点である．

GridPort[13] は，Grid 上の計算資源を Web から透過的に使用し，Science Portal の構築を支援するツールキットである．Globus や，CGI，Perl 等の標準の Web 技術を用いて構築されている．しかし，そのアーキテクチャ構成では，JiPANG のように透過性，統一性のあるインタフェースを提供していない．

CoG[12] プロジェクトは，Commidity Grid Toolkits (CoG Kits) の設計と開発を行っており，Globus が提供する機能に対して Java, CORBA, DCOM 等の コモディティ技術のインタフェースを定義し，その実装を提供している．CoG は，5 章で述べたように，JiPANG の Globus サービスとして使われ，JiPANG とは相補関係にある．

7. まとめと今後

本稿では，Jini 技術を基盤に構築した Computing Portal システム JiPANG の概要について述べた．JiPANG は，様々な Grid 技術の統一的，透過的な使用を可能とし，Grid 環境上のアプリケーション開発を促進する．

JiPANG のインフラストラクチャは様々な応用が考えられる．現在，米国インディアナ大学の CCA (Common Component Architecture) の実装である XCAT システム [14] のコンポーネント管理に使用することが検討されている．また，JiPANG の基盤上に遊休計算機を有効利用したシステムの構築や，Computing Economy 等の導入が考えられる．

今後は，JiPANG をより実用的なシステムとする為に，Global Grid Form で定められた Grid 上のセキュリティモデルの統合や，サービスの充実化を図る必要がある．

参 考 文 献

1) Global Grid Forum. http://www.gridforum.org/.
2) Grid Computing Environments Working Group. http://www.computingportals.org/.
3) GridRPC Tutorial. http://ninf.etl.go.jp/papers/gridrpc_tutorial/.
4) Jini. http://www.jini.org/.
5) JiPANG. http://ninf.is.titech.ac.jp/jipang/.
6) Jython. http://www.jython.org/.
7) The ServiceUI Project. http://www.artima.com/jini/serviceui/.
8) Piyush Mehrotra Ahmed Al-Theneyan and Mohammad Zubair. Enhancing Jini for use across non-multicastable networks. In *ICASE Report No.2000-34*, 2000.
9) Henri Casanova and Jack Dongarra. NetSolve: A Network Server for Solving Computational Science Problems. In *Proceedings of Super Computing '96*, 1996. http://www.cs.utk.edu/netsolve/.
10) WojtekFurmanski ErolAkarsn, GeoffreyC.Fox and Tomasz Haupt. Webflow - high-level programming environment and visual authoring toolkit for high performance distributed computing. In *Proceedings of Supercomputing '98*. http://www.npac.syr.edu/users/haupt/WebFlow/.
11) Ian Foster and Carl Kesselman. The Globus Project: A Status Report. In *Proc. IPPS/SPDP '98 Heterogeneous Computing Workshop*, pages 4–18, 1998.
12) Jarek Gawor Gregor von Laszewski, Ian Foster. Cog kits: A bridge between commodity distributed computing and high-performance grids,a java commodity grid kit. In *ACM 2000 Java Grande Conference*, June 2000.
13) Steve Mock Mary Thomas and Jay Boisseau. Development of web toolkits for computational science portals: The npaci hotpage. In *Proceedings of HPDC 9*, pages 308–309, August 2000.
14) Dennis Gannon Randall Bramley. A Component Based Services ARchitecture for Building Distributed Applications. In *Proceedings of HPDC 2000*, August 2000.
15) Neil T. Spring Rich Wolski and Jim Hayes. The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing.
16) C. Kesselman G. von Laszewski W. Smith S. Fitzgerald, I. Foster and S.Tuecke. A Directory Service for Configuring High-Performance Distributed Computations. In *Proc. 6th IEEE Symp. on High-Performance Distributed Computing*, pages 365–375, 1997.
17) Satoshi Sekiguchi, Mitsuhisa Sato, Hidemoto Nakada, and Umpei Nagashima. −Ninf−: Network base information library for globally high performance computing. In *Proceedings of Parallel Object-Oriented Methods and Applications (POOMA)*, Feb. 1996. http://ninf.etl.go.jp/.
18) Toyotaro Suzumura, Satoshi Matsuoka, and Hidemoto Nakada. A Jini-based Computing Portal System. In *Proceedings of SC 2001*, November 2001. 掲載予定.