社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2006－HPC－106 (8)
2006／6／12

GridRPCアプリケーションポータルの構築・運用を支援するポートレット

佐々木 耕† 渡邊 啓正† 本多 弘樹†

グリッドの普及と伴にグリッド上で動くアプリケーションは多様化し，アプリケーション開発者のグリッドポータルへの要求も多様化している。そこで我々は，アプリケーションの要求を柔軟に満たす拡張性や再利用性をもつグリッドポータル構築ツールを開発している。本システムではGridRPCシステムの一つであるNinf-Gに着目し，リモートライブラリやクライアントプログラムの管理機能をもつポートレットを提供することで，アプリケーション開発者がGridRPCアプリケーションのポータル構築から運用までを行うことを支援する。また，本システムをJetSpeed2に適用してシステムの柔軟性を確認した。

Portlets for Building and Operating GridRPC Application Portal

TSUTOMU SASAKI †, HIROMASA WATANABE † and HIROKI HONDA †

Application developers have more and more requirements for grid portals with the increased variety of grid applications. We are developing grid portal development tools with extensibility and reusability to meet application demands. We chose Ninf-G and developed portlets which have administration functions of remote libraries and client programs. They help application developers to build and operate GridRPC application portals. We also evaluated our tools had flexibility with JetSpeed2.

1. はじめに

グリッドポータルは，基盤にある複雑なグリッド技術を抽象化する高位のユーザインターフェースとして重要である。しかし，グリッドの普及と伴にグリッド上で動くアプリケーションは多様化し，アプリケーション開発者のグリッドポータルへの要求も多様化している。

そこで我々は，アプリケーションの要求を柔軟に満たす機能の拡張性や再利用性の高いグリッドポータル構築ツール[1)2)]を開発している。本ツールでは，対象とするグリッドミドルウェアをGridRPC[3)]システムの一つである Ninf-G[4)]に絞り，グリッドポータルの構築をアプリケーションの配置から支援することで，グリッドポータルを構築する手間を軽減する。

† 電気通信大学大学院情報システム学研究科
Graduate School of Information Systems, University of Electro-Communications

ツールの開発において，近年注目を浴びているGridSphere[5)]と GridPortlets[6)]をベースに設計と実装を行った。グリッドポータルとしての中心的な機能は，既に多数の利用事例[7)8)]が報告され信頼のある既存のシステムを利用できる。また，GridPortlets にGridRPC システムを考慮した主に以下に示す機能をもつポートレットを拡張して，GridRPC システムの利用を支援する。

- リモートライブラリのインストール機能
- リモートライブラリの情報提示機能
- クライアントプログラムの情報提示機能
- ユーザインターフェースの生成機能

これにより，アプリケーション開発者が GridRPC アプリケーションのグリッドポータル構築から運用までを行う手間を軽減し，高機能なグリッドポータルを構築する環境を提供できる。

本稿の構成は以下の通りである。2章で GridSphere と GridPortlets について説明する。3章と4章で開発したポートレットの設計および実装について述べる。

5章で関連研究について言及し，6章で動作試験について述べる。そして最後に7章でまとめと今後の課題について述べる。

2. GridSphereとGridPortlets

GridSphereは，GlidLab[9]によって開発が行われているグリッドアプリケーションやミドルウェアの開発に焦点を当てたオープンソースのポータルフレームワークである。ポートレットの管理，ユーザの管理，レイアウトの管理，そしてユーザに応じたアクセスコントロールなど，グリッドポータルとしての中心的な機能を提供する。ポータルの再利用可能な構成要素であるポートレットの標準化JSR-168[10]に準拠したアーキテクチャは，ソフトウェアの共有やコードの再利用を容易にし，ポータルの開発に柔軟性や拡張性をもたせる。

GridSphereを使う利点はその機能だけでなく，同伴するグリッドポータル構築ツールのGridPortletsにもある。GridPortletsは基盤にあるGlobus Toolkit[11]などのグリッド技術を抽象化し，開発者が容易にグリッドポータルを構築できるための高レベルなAPIを提供する。GridPortletsには，プロキシ認証，リソース，ジョブ，そしてリモートファイルを管理し，それらの情報を保守するポートレットやサービスが含まれる。

3. ポートレットの設計

3.1 アーキテクチャの要件

グリッドポータルを構築するシステムの開発はこれまでも多数行われてきた。しかしシステムに柔軟性がないため，グリッドポータルにジョブの結果を可視化する機能を付加したいといった要求に応じることが困難だった。アプリケーションの要求を柔軟に満たすためには，グリッドポータルを自由にカスタマイズできる機能が必要である。GridSphereのフレームワークに基づいたポートレットモデルであれば，高機能なグリッドポータルを構築する環境を提供できる。

また従来のシステムは，アプリケーションを呼び出すコードやユーザインターフェースの実装を支援するグリッドポータル構築ツールが中心だった。しかし，アプリケーション開発者がグリッドポータルを構築する手間を軽減するには，アプリケーションの配置から支援する必要がある。同じGridRPCシステムのNinf-GとNetSolve[12]であってもアプリケーションを配置する方法は異なるため，特定のグリッドミドルウェアに対応したグリッドポータル構築ツールが必要である。グリッドミドルウェアの1つであるCondor[13]のアプリケーションを利用したい場合には，CondorJobLauncherPortlet[14]がある。このようにポートレットの標準化によって，アプリケーションの用途に合わせてさまざまな機能をもつポートレットを自由に組み合わせてグリッドポータルを構築できる。

以上の理由により我々は，Ninf-Gを対象としたポートレットの開発を行った。Ninf-Gを対象としたとき，アプリケーション開発者がグリッドポータルを構築から運用までを行うのに必要な機能の設計について述べる。

3.2 リモートライブラリのインストール機能

GridRPCアプリケーションを動作させるには，利用する全てのサーバにリモートライブラリをインストールしなければならない。リモートライブラリのインストール機構にRelis-G[15]があるが，アプリケーション開発者がサーバのリソース情報を収集してサーバの管理ポリシーやインストール環境を定義するファイルを作成することは困難である。アプリケーション開発者がサーバを意識しなくても，リモートライブラリをインストールできる機能が必要である。

3.3 リモートライブラリ情報の提示機能

グリッドポータルを運用していく上でアプリケーションに何らかの障害が起きたとき，アプリケーション開発者がリモートライブラリの状態を逐一サーバに確認することは非常に手間がかかる。アプリケーションを保守するためには，アプリケーション開発者が全てのリモートライブラリの情報を視覚的に一目で把握できる機能が必要である。

3.4 クライアントプログラム情報の提示機能

リモートライブラリと同様にクライアントプログラムの情報も視覚的に確認できる機能が必要である。また，グリッドポータルを運用していく上でアプリケーションに何らかの変更を加えたとき，アプリケーション開発者がグリッドポータルを構築する作業を一からやり直すことは手間がかかる。一度利用したアプリケーションを再利用できる機能が必要である。

3.5 ユーザインターフェースの生成機能

グリッドポータルを構築するには，アプリケーションに合わせたジョブ投入サービスやユーザインターフェースを作成しなければならない。アプリケーション開発者がGridPortletsに関する知識やWebプログラミングに熟知することは困難である。アプリケーション開発者が特別な知識をもたなくても，グリッドポータルを構築できる枠組みが必要である。

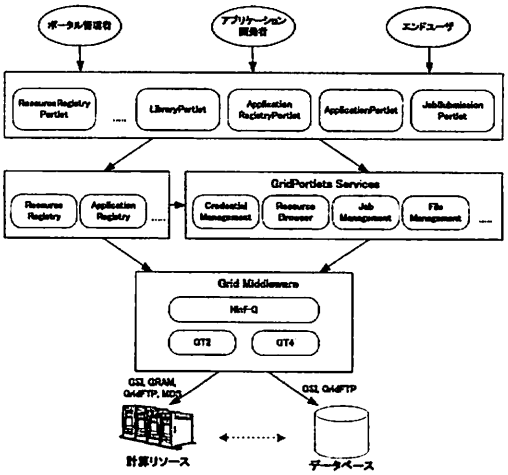

図 1.アーキテクチャ

3.6 システムの構成

本システムのアーキテクチャは GridSphere のフレームワークに基づいている。ソフトウェアコンポーネントの相互関係を図 1 に示す。本システムは以下のポートレットとその機能によって構成される。

- ResourceRegistryPortlet
  - リソース情報定義機能
- LibraryWorkflowPortlet
  - インストールワークフロー定義機能
- LibraryPortlet
  - リモートライブラリインストール機能
  - リモートライブラリ情報提示機能
- ApplicationRegistryPortlet
  - アプリケーション情報定義機能
- ApplicationPortlet
  - クライアントプログラム情報提示機能
  - ユーザインターフェース生成機能
- JobSubmissionPortlet
  - ジョブ管理機能

全てのポートレットはサービス指向アーキテクチャ (SOA) に基づいて設計されているため, GridPortlets のファイル管理サービスやジョブ管理サービスを用いて, Ninf-G システムにおけるファイル管理やジョブ管理を実現できる。各ポートレットは独立したものであるが, サービスレイヤーを通じてポートレット同士を連携させることができる。このように, SOA に基づいたポートレットアーキテクチャは, グリッドポータルの開発スピードを劇的に早め, グリッドポータルの高い拡張性を実現する。

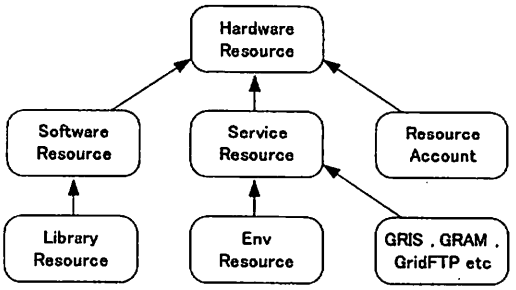

図.2 リソースモデル

```
<?xml version="1.0" encoding="UTF-8"?>
<grid-resources>
    <hardware-resource label="ISPPL MyProxy"
                       description="ISPPL Cluster"
                       hostname="lalituple0.yuba.is.uec.ac.jp">
        <myproxy-resource label="MyProxy"
                          description="Online Credential Repository"
                          portalProxyFile="/tmp/proxy.pem"
                          usePortalCredential="false"/>
        <gram-resource label="Globus Gatekeeper"
                       description="Globus Resource Management Service"/>
        <ws-gram-resource label="WS"
                          description="WS"/>
        <gridftp-resource label="Grid Ftp"
                          description="Grid Ftp Service"/>
        <library-resource libraryName="dadd"
                          libraryPath="/portal/test_ng"
                          libraryDescription="dadd"/>
        <library-resource libraryName="pi_trial"
                          libraryPath="/portal/test_ng"
                          libraryDescription="pi_trial"/>
    </hardware-resource>

    <hardware-resource label="MJ ISPPL"
                       description="ISPPL Cluster"
                       hostname="mj.yuba.is.uec.ac.jp">
        <gram-resource label="Globus Gatekeeper"
                       description="Globus Resource Management Service"/>
        <gridftp-resource label="Grid Ftp"
                          description="Grid Ftp Service"/>
        <env-resource libraryName="add"
                          srcName="dadd"
                          MG_DIR="/usr/local/mg2.3.0"
                          INSTALL_DIR="/portal/tmp"/>
        <env-resource libraryName="pi"
                          srcName="pi_trial"
                          MG_DIR="/usr/local/mg2.3.0"
                          INSTALL_DIR="/portal/tmp"/>
    </hardware-resource>
</grid-resources>
```

図 3.リソース情報ファイル

4. ポートレットの実装

4.1 ResourceRegistryPortlet

ポータル管理者が利用できる計算資源やサービス資源などのリソース情報を, Web ブラウザ上のフォームに XML ファイルとして記述できる機能を提供する

リソース情報は, 図 2 に示す HardwareResource を頂点とする階層構造でデータベースに保存される。本システムでは, リモートライブラリを管理するために LibraryResource と EnvResource を拡張して, アプリケーション開発者がリモートライブラリの情報をリソース情報ファイルに記述できるように実装した。LibraryResource はライブラリに関する情報, EnvResource はリモートライブラリをインストールする環境に関する情報である。図 3 にリソース情報ファイルの記述例を示す。タグ<library-resource>と<env-resource>で囲まれた部分が本システムで拡張した情報である。

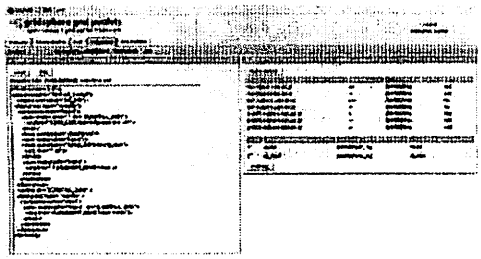

図 4.LibraryWorkflowPortlet と LibraryPortlet

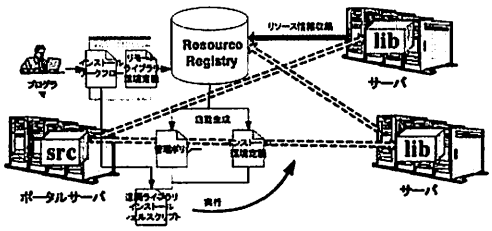

図 5. LibraryPortlet の動作

4.2 LibraryWorkflowPortlet

アプリケーション開発者がサーバにおけるリモートライブラリのインストールワークフローを，Web ブラウザ上のフォームに記述できる機能を提供する。インストールワークフローファイルは Relis-G で用いられる。

4.3 LibraryPortlet

提示されたライブラリの一覧から 1 つを選択すれば，サーバへ自動的にリモートライブラリをインストールできる機能を提供する。図 5 に示すようにデータベースのリソース情報から，データバインディングツールの Castor[16]を用いてサーバの管理ポリシーとインストール環境を定義するファイルが自動生成される。そして，インストールワークフローファイルと合わせて Relis-G のシェルスクリプトが実行され，サーバにリモートライブラリがインストールされる。

また，リモートライブラリがインストールされた各サーバにおけるリモートライブラリの情報を提示する機能を提供する。データベースのリソース情報と GridPortlets のファイルサービスを用いて提示する情報は以下の通りである。

- サーバ名
- リモートライブラリ名
- リモートライブラリのパス
- リモートライブラリの有無

```
<?xml version="1.0" encoding="UTF-8" ?>
<application>
        <appinfo>
                <exe>pi_test</exe>
                <subtitle>monte carlo pi computation</subtitle>
                <client>latitude10.yuba.is.uec.ac.jp</client>
                <path>/portal/$site</path>
                <configuration>client.conf</configuration>
                <libraries>
                        <library name="pi"/>
                </libraries>
                <description>test program</description>
        </appinfo>
        <parameters>
                <parameter type="text">
                        <title>Input Parameter</title>
                        <arg>arg0</arg>
                </parameter>
</application>
```

図 6.アプリケーション情報ファイル

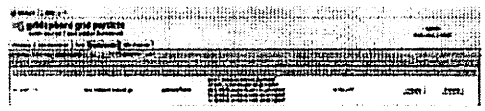

図 7.ApplicationPortlet

4.4 ApplicationRegistryPortlet

アプリケーション開発者がクライアントプログラムやパラメータなどのアプリケーション情報を，Web ブラウザ上のフォームに XML ファイルとして記述できる機能を提供する。

アプリケーション情報の記述例を図 6 に示す。アプリケーション情報は O/R マッピングツールの Hibernate[17]を用いてデータベースに保存される。このとき，GridPortlets のファイルサービスを用いて，リモートライブラリのインストールによって生成された Ninf-G の LDIF ファイルが各サーバから指定したクライアントプログラムのフォルダに自動的にダウンロードされる。さらに，データベースのアプリケーション情報とリソース情報からクライアントコンフィグレーションファイルが自動生成される。

4.5 ApplicationPortlet

これまでに登録したアプリケーションの情報を提示する機能を提供する。一度登録したアプリケーションは何度でも再利用できる。データベースのアプリケーション情報を用いて提示する情報は以下の通りである。

- アプリケーション情報ファイル
- サーバ名
- クライアントプログラム名
- クライアントプログラムのパス
- LDIF ファイル
- クライアントコンフィグレーションファイル

また，提示されたアプリケーション情報の一覧から 1 つを選択すれば，自動的にアプリケーションに合わせたユーザインターフェースを生成できる機能を提供する。データベースのアプリケーション情報を JSP(Java Server Pages)に渡してユーザインターフェースが生成される。

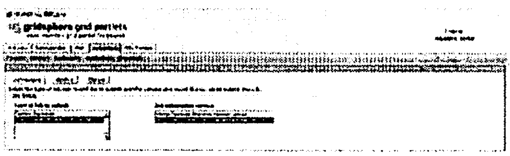

図 8.JobSubmissionPortlet

```
<?xml version="1.0" encoding="UTF-8"?>

<job>

<jobCredentialEndpoint xsi:type="wsa:EndpointReferenceType"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
            xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/03/addressing">

 <wsa:Address xsi:type="wsa:AttributedURI">
        https://latitude10.yuba.is.uec.ac.jp:8443/wsrf/services/DelegationService</wsa:Address>
 <wsa:ReferenceProperties xsi:type="wsa:ReferencePropertiesType">
  <wsa:DelegationKey xmlns:wsa="http://www.globus.org/08/2004/delegationService">
        61ba97a0-dc0a-11da-ba77-c28d848d0442</wsa:DelegationKey>
 </wsa:ReferenceProperties>
 <wsa:ReferenceParameters xsi:type="wsa:ReferenceParametersType"/>

</jobCredentialEndpoint>

    <executable>/portal/2site/pi_client_multi</executable>
    <directory>/tmp</directory>
    <argument>1000</argument>
    <argument>ml.yuba.is.uec.ac.jp</argument>
    <argument>grid00.yuba.is.uec.ac.jp</argument>
    <stdin>/dev/null</stdin>
    <stdout>stdout</stdout>
    <stderr>stderr</stderr>

</job>
```

図 9.ジョブディスクリプションファイル

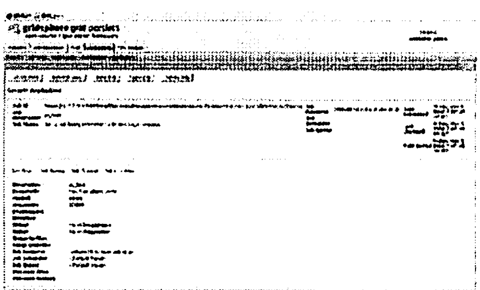

図 10.ジョブの結果

4.6 JobSubmissionPortlet

エンドユーザがジョブの投入や結果を取得できる機能を提供する。標準では一般的なジョブ投入のユーザインターフェースしか提供されていないため，図 8 に示すように ApplicationPortlet で生成したユーザインターフェースを用いてジョブを実行できる新たなジョブタイプを実装した。

エンドユーザがユーザインターフェース上で入力するパラメータは，ジョブ実行サービスに渡されてクライアントプログラムが実行される。しかし Ninf-G は，クライアントプログラムを呼び出すときに Globus Toolkit の GRAM を用いるため，同じ GRAM を使う通常のジョブ実行サービスでは Globus Toolkit の仕様によりクライアントプログラムを実行できない。そこで，現在開発中の gt4portlets のジョブ実行サービスを用いて，図 9 に示すようなジョブディスクリプションファイルを作成してクライアントプログラムを実行できる新たなジョブサービスを実装した。ジョブ投入後の画面を図 10 に示す。

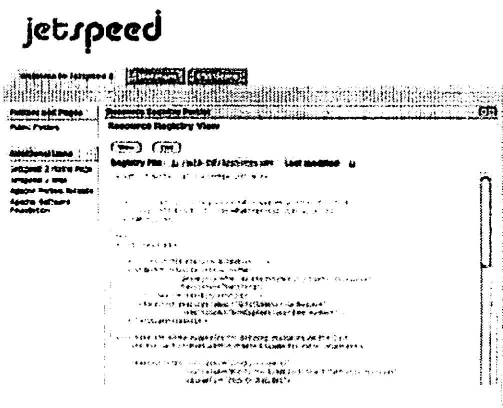

図 11.JetSpeed2

5. 動作試験

モンテカルロ法により円周率を求める GridRPC アプリケーションを本システムに適用して，グリッドポータルを構築できることを確認した。

また，GridSphere 以外のポータルフレームワークで本システムが動作するかを検証するために，JSR-168 に準拠する JetSpeed2[18] を選択した。JetSpeed2 は Apache[19]によって開発が進められている，企業情報ポータル(EIP)のオープンソースの実装である。図 11 に示すように本システムを JetSpeed2 上で動作させて，機能の再利用性や拡張性が高い柔軟性のあるシステムであることを確認した。

6. 関連研究

産業総合技術研究所で開発が進められているグリッドポータル構築ツールに GridASP[20]がある。これは，ASP（アプリケーションサービスプロバイダ）をサービスプロバイダ，アプリケーションプロバイダ，リソースプロバイダの 3 つに分けて，それぞれを異なる事業体が受け持つことで今後のビジネス展開が期待されている。このように GridASP は，GridSphere よりも不均質な機関をまたがって利用されることからセキュリティに関する機能が強化されている。しかし現状では，ポートレットの標準化である JSR-168 に未対応の JetSpeed1 をベースにして開発が行われているため，本システムを GridASP 上で動かすことができない。

7. おわりに

本稿では，アプリケーション開発者が GridRPC アプリケーションのポータル構築から運用までを行う手間を軽減するポートレットの設計および実装について述べた。GridRPC アプリケーションを本システムに適用してグリッドポータルを構築できることを確認した。

また，本システムをJetSpeed2に適用してシステムの柔軟性を確認した。

今後の課題としては，より実用性のあるGridRPCアプリケーションを本システムに適用して，グリッドポータルの構築だけでなく運用についても本システムの有効性を検証することが挙げられる。

参考文献

1) GridPSEBuilder. http://www.gtrc.aist.go.jp/GridPSEBuilder/
2) Suzumura, T., Nakada, H., Matsuoka, S. and Casanova, H.: GridSpeed: A web-based Grid Portal Generation Server, HPCAsia2004, (2004).
3) Semimour, K., Nakada, H., Matsuoka, S., Dongarra, C., Lee, C. and Casanova, H.: Overview of GridRPC: A Remote Procedure Call API for Grid Computing, Proceedings of Grid Computing – Grid2002, pp.274-278 (2002).
4) Ninf-G http://ninf.apgrid.org/
5) GridSphere. http://www.gridsphere.org/
6) Russell, M., Novotny, J., and Wehrens, O.: The Grid Portlets Web Application: A grid Portal Framework, http://www.gridsphere.org/gridsphere/html/publications/GridPortlets.pdf
7) Chongjie, Z., Chirag. D., Gabrielle, A., Ian, K. and Jon, M.: An Application portal for Collaborative Coastal Modeling, (2005).
8) Lin, M., W.Warker, D.: A Portlet Service Model for GECEM, AHM2004, http://www.allhands.org.uk/proceedings/papers/233.pdf#search='GECEM%20portlet', (2004).
9) GlidLab. http://www.gridlab.org/
10) JSR168. http://www.jsr168.org/
11) Globus Toolkit. http://www.globus.org/
12) NetSolve. http://icl.cs.utk.edu/netsolve/
13) Condor. http://www.cs.wisc.edu/condor/
14) OGCE. http://www.collab-ogce.org/nmi/index.jsp/
15) Watanabe, H., Honda, H., Yuba, T., Tanaka, Y., and Sato, M.: Relis-G: Remote Library Install System for Computational Grids, HPCAsia, (2004).
16) Castor. http://www.castor.org/
17) Hibernate. http://www.hibernate.org/
18) Jetspeed. http://jakarta.apache.org/
19) Apache. http://www.apache.org/
20) GridASP. http://www.gridasp.org/