

グリッドコンピューティングのためのウェブブラウザ を用いたクライアント環境の構築

楢橋 幹一郎¹ 富田 清司¹ 杉崎 由典¹ 西沢 厚¹ 金澤 宏幸¹ 伊藤 泰善¹
義久 智樹² 金澤 正憲²
¹富士通(株) ²京都大学

UNICORE Client に代表されるようなこれまでのグリッドジョブコントローラーは、機能が多くの反面、構造が複雑で専門的な技術が必要であった。本研究では、機能を必要最低限に絞ることで、初心者でも Web ブラウザから簡単にジョブを投入できるシンプルなジョブ投入環境「シンプルグリッドジョブコントローラー」を提案する。本稿では、シンプルグリッドジョブコントローラーの構築環境と開発プログラムについて説明し、クライアントツールとしての有用性について考察を行う。これにより、よりシームレスなグリッド環境との接続性と、シンプルグリッドジョブコントローラーの有用性を確認する。

A Construction of Client System for Grid Computing Using Web Browser

Kanichirou NARAHASHI¹, Kiyoshi TOMITA¹, Yoshinori SUGISAKI¹, Atsushi NISHIZAWA¹,
Hiroyuki KANAZAWA¹, Yasuyoshi ITOU¹, Tomoki YOSHIHISA², Masanori KANAZAWA²
¹Fujitsu Limited, ²Kyoto University

Although conventional job controllers such as UNICORE client system have many functions, it requires expert knowledge and it is difficult for beginners to use it. In this research, we propose a simple user interface called "Simple Grid Job Controller" system. In this system, since we can submit job processes using Web browsers, it is easy for beginners to use grid computing. In this paper, we describe the design and the implementation for the simple grid job controller and discuss its availability. As a result, we confirmed that our proposed system provides nice performance for a simple grid environment.

1. はじめに

最近のスーパーコンピュータは、共有メモリ型並列コンピュータから PC クラスタまで、システム形態が多様化してきている。また実験装置や観測装置から得られるデータは膨大な量になり、データ量の増大とデータの分散が顕在化している。このため、プログラムが実行されるコンピュータと、入力データを有するコンピュータが異なる場合のジョブ処理を一般的な形態として考え、¹UNICORE[1]を用いたジョブ連携システムを検討し、開発を行ってきた。

京都大学では、2005 年に実験的に UNICORE の環境を構築した。現在、次のステップとして、他大学とのシステム連携が期待されている。

また、これまでジョブの投入や確認は PC にインストールした UNICORE Client[1]から行ってきたが、

UNICORE Client に代表されるこれまでのグリッドコンピューティング向けジョブコントローラーは、機能が多くの反面、構造が複雑で専門的な技術が必要であった。

このような背景のもと、本研究では、シンプルで使いやすく、かつ大学間のシステム連携も意識したジョブコントローラーを開発した。

本稿では、必要な機能に絞ることで初心者でも簡単にジョブを投入できる、シンプルなジョブ投入環境を提案する。本稿で構築したジョブ投入システムを「シンプルグリッドジョブコントローラー」と呼ぶ。シンプルグリッドジョブコントローラーは、内部環境を極力ユーザから隠蔽することで、必要な機能を満たしながら複雑性を排除したシンプルなインターフェースを特長としている。

シンプルグリッドジョブコントローラーは、欧州富士通研究所が開発した Arcon Client Library をベースに、Web ブラウザを用いて UNICORE を利用する

¹ UNICORE は、分散するスーパーコンピュータを遠隔地からシームレスに利用することを目的としたグリッドミドルウェアである。

方式を採用した。構築したシステムを評価することにより、よりシームレスなグリッド環境との接続性と、クライアントツールとしての有用性を確認する。

以下、第2章でUNCOREの実験環境について、第3章でシンプルグリッドジョブコントローラーのシステム構成について、第4章でユーザがジョブを投入するまでの利用フローについて説明し、第5章でシンプルグリッドジョブコントローラーの考察を行い、最後に第6章で本稿のまとめと今後の展開について述べる。

2. 京都大学の UNICORE 実験環境

2.1 サーバ環境

京都大学学術情報メディアセンターでは 2005 年に、2 台のサーバを使用して UNICORE の実験システム環境を構築した。実験システムの構成を図 1 に示す。また、UNCORE のコンポーネントの版数について、表 1 に示す。なお、UNCORE のコンポーネントは以下の URL から入手した。

<http://unicore.sourceforge.net/unicoreatsf.html>[2]

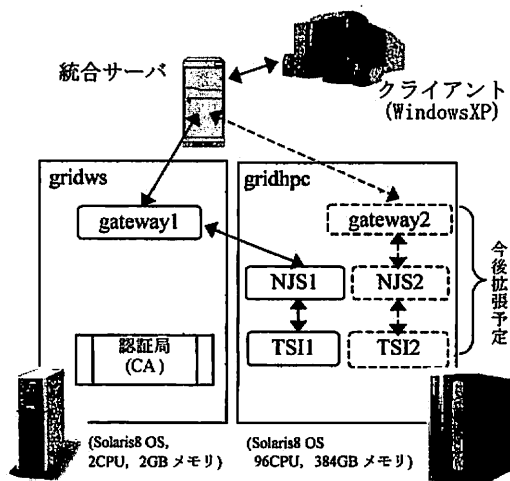


図1 UNICORE実験システム構成

表 1 対象バージョンとインストール先マシン

コンポーネント	バージョン	マシン
gateway	4.1.1 build2	gridws
NJS	4.6.1 build2	gridhpc
TSI	4.1.2 build1	gridhpc
UADB	1.0.0	gridhpc
CLIENT	5.6.2	WindowsXP端末

マシンは 2 台構成で、1 台はオペレーションノード、もう 1 台は計算ノードとして機能している。と

もにアーキテクチャは SPARC(64bit 版)、OS は Solaris8 である。計算ノードは CPU を 96 個、メモリを 384GB 搭載しているスカラ並列計算機で、最大 500GFLOPS の比較的高速な演算処理にも対応している。

オペレーションノード(gridws)には gateway のみがインストールされているが、実験用認証局(CA)の環境も構築されており、UNCORE で必要なサーバ証明書、クライアント証明書の発行は全てここで行っている。

また、計算ノード(gridhpc)には NJS, TSI がインストールされており、オペレーションノードの gateway と連携して、ジョブの監視、実行環境を提供している。

なお、一連の UNICORE 実験環境の構築手順は、北陸先端科学技術大学院大学の UNICORE インストールガイドの手順に沿って行った[3]。また、認証局(CA)については、NAREGI ミドルウェア α 版[4]の試用を行っていることもあり、NAREGI から配布されている NAREGI-CA (バージョン 1.0) で構築している。

実験環境では USite, VSite は 1 つである。VSite とは Virtual Site を省略したもので、1 つ以上の実行計算機群を 1 つに仮想化したものである。USite とは Unicore Site を省略したもので、1 つ以上の VSite から構成される UNICORE を利用する上でのグループを表す。

なお、今後の他大学との連携を視野に入れ、現在 gridhpc 上に別の gateway, NJS, TSI 環境を構築し、仮想的に 2 つ以上の USite, VSite で構成された実験環境の構築も行っている。

2.2 クライアント環境

クライアントは、UNCORE Forum から配布されているクライアントツールである UNICORE Client を PC 端末にインストールして使用した。ジョブ投入画面を図 2、ジョブ投入結果確認画面を図 3 に示す。

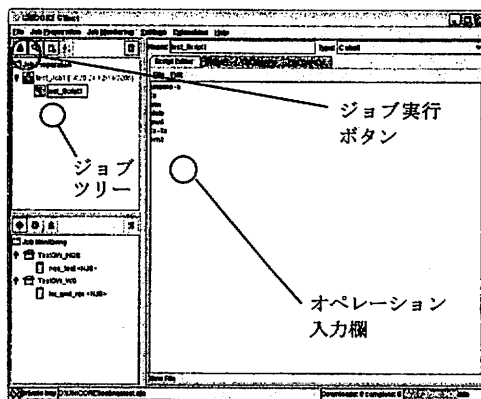


図2 UNICORE Clientのジョブ投入画面

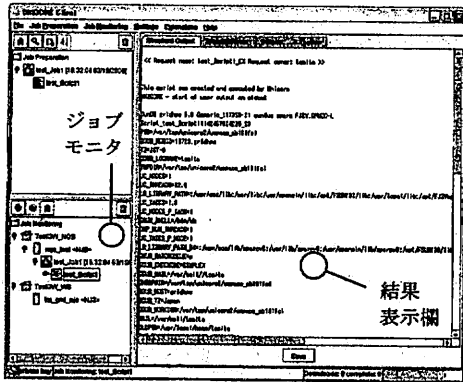


図3 UNICORE Clientのジョブ結果確認画面

UNICORE Clientは、ジョブの生成、実行、ステータス確認はもとより、ジョブの予約機能や、複数のVSiteをまたがる複雑なジョブの実行機能など、多彩な機能を備えている。その反面、以下のデメリットも同時にあわせ持っている。

- ・インストールや環境構築、また利用にあたって専門知識が必要となる。
- ・機能が豊富であるため、GUIが複雑化し、初心者にとって利用しづらい面がある。
- ・クライアントに専用プログラムをインストールする必要があるため、利用環境が限定される。

本研究では、上記問題を解決するため、グリッドに関する知識を持たない初心者でも簡単に利用でき

るジョブ投入環境を検討した。本稿では、前述の構想の下、京都大学と富士通が共同で開発し、新規のジョブ投入環境を構築した。本システムをシンプルグリッドジョブコントローラーと呼ぶ。第3章でシンプルグリッドジョブコントローラーの構成について述べる。

3. シンプルグリッドジョブコントローラーの構成

シンプルグリッドジョブコントローラーは、UNICOREをターゲットとしたグリッド環境において、Webブラウザからの簡単な操作でジョブの投入、確認、削除等の操作を行うことを可能にしたシステムである。

本システムの特長として、まず、単純化されたインターフェースが挙げられる。UNICORE Clientの場合、ユーザが定義するジョブとアクションは、Javaクラスであるアブストラクトジョブオブジェクト(AJO: Abstract Job Object)群として表現され、ターゲットとなるシステムで具体的なジョブやアクションにマッピングされる。遠隔地の複数の計算機をまたがるようなジョブを実行する場合、ユーザが作成するAJOは複雑化し、煩雑な操作が必要となる。シンプルグリッドジョブコントローラーでは、AJOの構造をパターン化してユーザに見せないよう内部でマッピングすることで、上記の問題を解決した。これにより、ユーザは、投入したいジョブのパターンを選択して、必要最小限の情報を入力するだけで、内部構造を特に意識せずに利用できるようになった。

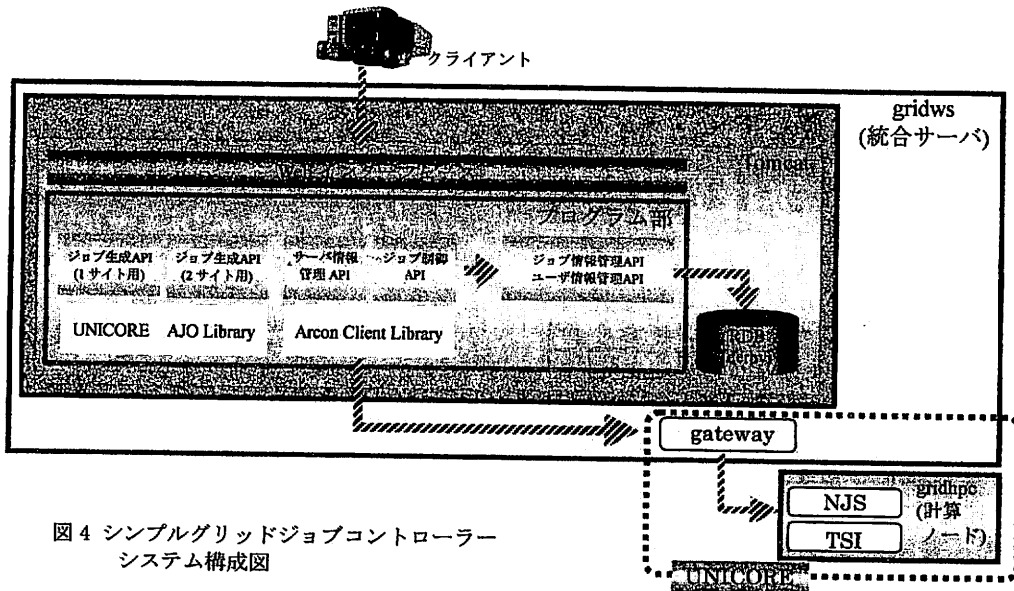


図4 シンプルグリッドジョブコントローラーシステム構成図

図4に、シンプルグリッドジョブコントローラーの構成を示す。本機能を、プログラム部とシステム部に分けて説明する。

ジョブの制御を行うプログラム部は、全て Java で実装されている。サーバ情報、ジョブ制御管理ライブラリには、Arcon Client Library を使用しているが、それ以外のジョブ制御部分およびジョブ管理関連の API 等については、京都大学と富士通が共同で開発した。表2にシンプルグリッドジョブコントローラーのジョブ制御プログラムの API の一覧と機能を示す。

表2 ジョブ制御プログラムの java API 一覧

機能	API
ジョブ生成 (1サイト用)	createJob(JobInfoForSingleBean jobinfo, int type)
ジョブ生成 (2サイト用)	createJob(JobInfoForMultipleBean jobinfo, int type)
ジョブ登録	Result registerJob()
ジョブ投入	Result submitJob(java.lang.String jobId)
ジョブ状態取得	Result getJobStatus(java.lang.String jobId)
ジョブ一時停止	Result holdJob(java.lang.String jobId)
ジョブ開放	Result releaseJob(java.lang.String jobId)
ジョブ中止	Result cancelJob(java.lang.String jobId)
ジョブ削除	Result deleteJob(java.lang.String jobId)

システム部については、サーブレットコンテナ (Tomcat) を使用しており、ユーザが Web ブラウザで Java プログラムを実行できるようにした。

なお、本システムではジョブをユーザ単位に管理しており、利用者を識別する必要がある。そのため、ログイン時に認証を行う仕組みを採用している。具体的には、まず、ユーザが事前に認証局 (CA) でユーザ証明書を発行してもらい、管理者がシステムのデータベース (Apache Derby) に証明書とパスワードを登録しておく、ユーザがログイン画面でユーザ ID とパスワードの入力を行うと、システムはデータベースに問い合わせを行い、証明書のユーザ ID とパスワードの照合を行ってログイン可否の判断が行われる。表3に、シンプルグリッドジョブコントローラーの構成コンポーネントとバージョンについて示す。

表3 シンプルグリッドジョブコントローラーの構成コンポーネントとバージョン

OS	Solaris	8
JAVA	Java SDK	1.4.2_10
APサーバ	Tomcat	5.0.28
Ant	Apache ant	1.6.5
DB	Apache Derby	10.1.2.1
APFW	Spring Framework	1.2.6

次に、シンプルグリッドジョブコントローラーの画面遷移について説明する。

利用の前に、システムへのユーザの証明書とパスワードの登録、計算ノードの UADB へのユーザ登録を行っておく必要があるが、一連の作業については、4章で述べることとし、ここでは登録作業が既に完了しているものとして説明する。

まず、ユーザは Web ブラウザを起動してシステムへアクセスする。ログイン画面が表示されるので、

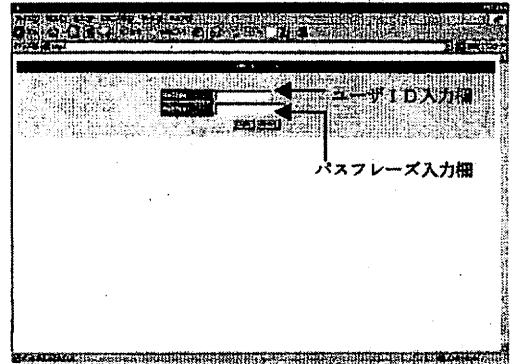


図5 ログイン画面

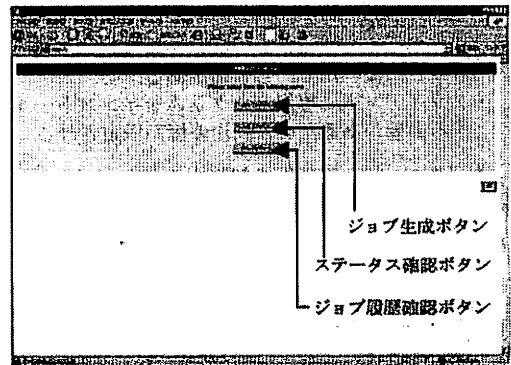


図6 メニュー画面

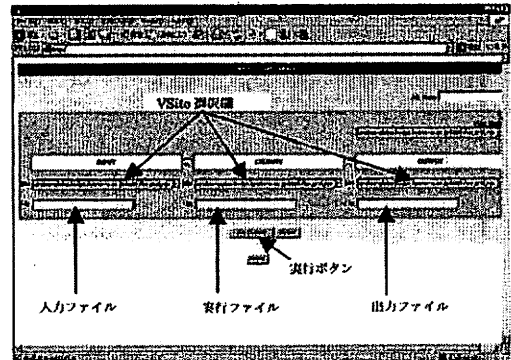


図7 ジョブ生成画面

あらかじめ登録しておいた証明書ユーザ ID とパスフレーズを入力して“login” ボタンを押す。ログイン画面を図 5 に示す。

認証が行われるとメニュー画面に遷移し、ジョブ生成ボタン、ステータス確認ボタン、ジョブ履歴確認ボタンが表示される。各々のボタンを押下することでそれぞれジョブの生成および投入、投入したジョブの状態確認、実行済みジョブの投入履歴の確認を行うことができる。ジョブの生成画面を図 7 に、ステータスの確認画面を図 8、図 9 に、履歴の表示画面を図 10、図 11 に示す。

ユーザは、ジョブ生成画面で、実行ファイルと出力ファイルのパス名を、リストからそれぞれ実行ファイルが存在する VSite とファイルを出力する VSite を選択し、“Submit” ボタンを押すことでジョブを投入できる。また、必要に応じて入力ファイルを指定すれば、実行ファイルの入力として指定することも可能である。なお、VSite の一覧リストには、利用可能な Vsite 名の一覧が表示される。このリストは、あらかじめ登録しておいた VSite を巡回してジョブの実行が可能な VSite を抽出して自動的に作成される。なお、出力結果を他の VSite で入力として実行するような複数の VSite をまたがる複雑なジョブについては、ジョブ生成画面の前にジョブ投入モデルパターンとして選択させることで、対応可能である。

ステータス確認画面では、ジョブ ID、ジョブ名、VSite 名、ステータス、投入日時が表示され、

Job ID	Job Name	VSite	Status	Submit Date
1092293973	test01	global.kudpc.kyoto-u.ac.jp/4001/ku_grid_njs	ABNORMAL_END	2006-03-15 19:15:38.852

図 8 ステータス確認画面

Job ID	Job Name	VSite	Status	Submit Date
1092293973	test01	global.kudpc.kyoto-u.ac.jp/4001/ku_grid_njs	ABNORMAL_END	2006-03-15 19:15:38.852

図 9 ステータス確認画面
(項目部分拡大図)

Job ID	Job Name	VSite	Status	Submit Date
1092293973	test01	global.kudpc.kyoto-u.ac.jp/4001/ku_grid_njs	ABNORMAL_END	2006-03-15 19:15:38.852

図 10 履歴表示画面

Job ID	Job Name	VSite	Status	Submit Date
1092293973	test01	global.kudpc.kyoto-u.ac.jp/4001/ku_grid_njs	ABNORMAL_END	2006-03-15 19:15:38.852

図 11 履歴表示画面
(項目部分拡大図)

“Update” ボタンを押すことで最新のジョブステータスに更新される。

履歴表示画面では、ログインユーザが過去に実行して終了したジョブの履歴の一覧が表示され、確認が可能である。

図 7 のジョブ投入画面からも明らかなように、シンプルグリッドジョブコントローラでは、ユーザは実行したいファイルと出力先さえ指定すれば、システムの利用が可能であり、グリッドミドルウェア (UNICORE) に関する特別な知識がなくても望む結果を得ることができる。

4. シンプルグリッドジョブコントローラのジョブ投入フロー

ユーザがシンプルグリッドコントローラを利用するにあたり、ユーザおよび管理者に必要な手順を以下に述べる。

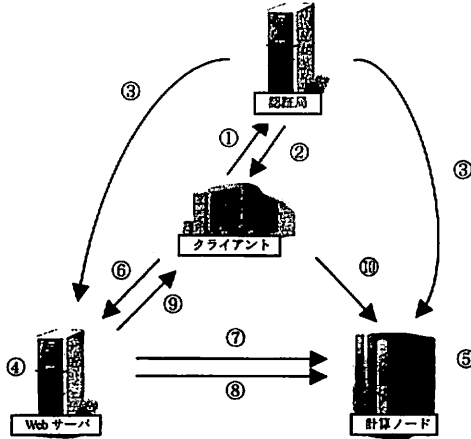
シンプルグリッドコントローラを利用するためには、事前に UADB への利用者登録と、別途システムへの利用者登録が必要となる。本システムの利用フローを図 12 に示す。

シンプルグリッドジョブコントローラをユーザが利用するためには、以下の 3 つの情報が必要となる。

- ・ ユーザ ID
- ・ ユーザ証明書
- ・ ユーザ証明書のパスフレーズ

ユーザはまずユーザ証明書を取得する必要がある。

実験用の認証局である NAREGI-CA には、Web エンロール機能という、Web からクライアント証明書を発行できる仕組みを実装しているため、クライアントはまずここで証明書の発行を行う(①, ②)。Web エンロール機能を使用してユーザ証明書の申請を行った Web ブラウザの画面を図 13 に示す。



番号	処理内容	From→To / location
①	ユーザ証明書要求	クライアント → 認証局
②	ユーザ証明書発行	認証局 → クライアント
③	ユーザ証明書の転送	認証局 → Webサーバ, 計算ノード
④	ユーザ証明書の配置	Webサーバ
⑤	RDBへのユーザ登録	Webサーバ
⑥	UADBへのユーザ登録	計算機
⑦	ログイン、ジョブ操作依頼	クライアント → Webサーバ
⑧	ジョブ操作	Webサーバ → 計算ノード
⑨	結果情報の取得	Webサーバ → 計算ノード
⑩	結果情報の表示	Webサーバ → クライアント
⑪	結果の取得	クライアント → 計算ノード

図 12 シングルグリッドジョブコントローラーのジョブ投入フロー

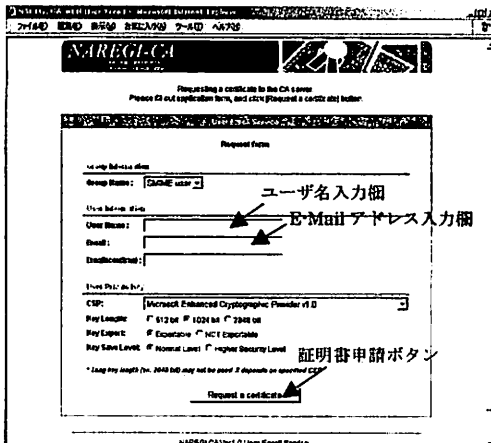


図 13 Web エンロールユーザ証明書申請画面

管理者は、証明書をシングルグリッドジョブコントローラーのマシンへ転送し(③)、システムのデータベース(apache derby)へユーザを登録する(④)。データベースには、ユーザ証明書ファイルとパスワードを格納したファイルのパス情報も同時に登録する。

次に、計算ノードを利用するための処理を行う。UNICORE 側でユーザ登録を行うため、計算ノードのデータベース(UADB)に利用者を登録し、ユーザと UNICORE のユーザとのマッピングを行う(⑤)。

上記、①から⑤の手順により、シングルグリッドジョブコントローラーの利用ができるようになる。

利用にあたっては、第 3 章で説明したように、ユーザは Web ブラウザからシステムにアクセスし、ユーザ ID とパスワードを入力して認証を受ける(⑥)。システム側では、ユーザはユーザ ID、各ジョブはジョブ ID でそれぞれ一意に識別され、管理される。ユーザは、Web ブラウザからジョブの投入、確認、履歴の参照、ジョブの削除等を行うことができる。システムは UNICORE のインターフェースを介してその都度、計算ノードへ処理要求を行い、処理結果を Web ブラウザに表示する(⑦, ⑧, ⑨)。

なお、計算結果ファイルについては、最後に計算が行われた計算ノード上に出力される。このため、ユーザは、ftp 等のファイル転送ツールを利用するか、ログインして直接、計算結果を参照する(⑩)。

5. 考察

5.1 クライアントツールとしての機能

シングルグリッドジョブコントローラーの最大の長は、簡略化されたインターフェースである。本システムでは、Web ブラウザでログインしてから、最短の場合で画面遷移が 2 回、5~7 箇所に入力を行うことでジョブを投入することができ。このため、UNICORE の専門知識がないユーザでも、マニュアルを読まずに利用することが可能である。また Web ベースのツールであることから、UNICORE のサイト環境に変更が生じた場合であっても、管理者が対応することで、ユーザは同一のインターフェースでシステムを利用できる。このように、本システムは、ユーザにとって、UNICORE を利用する上での技術的な敷居を下げる効果がある。今後他大学と連携し、システムの利用範囲をさらに拡大した場合、複数のサイトで実行するような複雑なパターンが想定されるが、そのような場面こそ、本システムの意義が発揮されると確信している。

5.2 システムの可搬性と拡張性

シングルグリッドジョブコントローラーは、ポータビリティを重視し、開発プログラムやシステムを構成するサーブレットコンテナやデータベースも全

て Java による実装で構成されている。そのため、ファイル式をコピーするだけで、容易に異なるアーキテクチャのマシンへシステム環境を移動することが可能である。またフレームワークを用いることでシステムコンポーネント間が疎結合となるため、サブレットコンテナやデータベースとの連携を容易に分離することができる。これにより、設定を変更するだけで Web システムとしてだけでなく、UNICORE クライアントツールキットとしてもそのまま利用することが可能である。さらに、Arcon Client Library にはないジョブ生成機能と拡張インタフェースを持っているため、プログラムを組み込むことにより、より複雑なジョブ生成機能を持ったシステムへ拡張することができる。

6. まとめと今後の展開

われわれの研究グループでは、これまで UNICORE の実験環境を構築してジョブ投入環境について検討してきた。本稿では、クライアント操作の煩雑性を解消して、より簡単にジョブを投入できる環境を実現するために、新規に「シンプルグリッドジョブコントローラー」を開発した。

これにより、gateway の存在や AJO の構造などの UNICORE の内部構造をほとんど意識することなく、ジョブの操作を行うことができ、シンプルグリッドジョブコントローラーの有用性を確認することができた。

今後の他大学の UNICORE システムと連携することにより、シンプルグリッドジョブコントローラーを利用した複数サイトへのジョブの実行により、シンプルグリッドジョブコントローラーのさらなる有効性を検討したい。

最後に、実験用認証局環境の構築と助言をいただいた京都大学金澤研究室の越本浩央様、ならびにシンプルグリッドジョブコントローラーの開発に尽力いただいた富士通株式会社科学ソリューション事業本部の関係者の方々に深く感謝する。

参考文献

- [1] UNICORE Forum Site, <http://www.unicore.org>
- [2] SourceForge.net Site, <http://sourceforge.net>
- [3] UNICORE install Guide, http://www.jaist.ac.jp/~h-ooiwa/research/unicore_install.pdf
- [4] National Research Grid Initiative Site, <http://www.naregi.org>