第 13 回
問題解決環境ワークショップ論文集
The 13th Problem Solving Environment Workshop'10

平成 22 年 9 月 16 日～17 日

主催：PSE 研究会
協賛：富士通株式会社

第13回PSEワークショップ2010開催にあたって

皆様のご支援とご協力のもと，今年度もPSE（Problem Solving Environment（問題解決環境））ワークショップを開催することができ，心より御礼申し上げます．今年は，2件の特別講演，8件の一般発表に加え，パネル討論会が企画されております．従来にも増して活発な議論，情報交換が期待できます．PSEワークショップでの研究の発表も，第1回PSEワークショップ開催当初から比べて幅が広がり，シミュレーション支援環境としてのPSEから，グリッドコンピューティング，教育支援，協調環境，さらに社会的な意義がより発揮できる社会科学分野への広がりを見せてきております．

PSEワークショップも昨年に第12回を迎え，今年は第13回，干支で言いますと第2ラウンドに入ることになります．この間に長年にわたって会をリードして来られた金沢大学教授 田子精男先生，静岡大学教授 梅谷征雄先生が相次いで退官され，メンバの顔ぶれも大分変って参りました．一方でこの会を通して学位をとられた若手メンバが学会や産業界で着実に活躍を始めております．その節目にあたり，今回は改めて田子先生，三浦先生を特別講演の講師としてお招きして日頃のお考えを伺うとともにパネル討論にもご参加頂き，本質に迫るより深い議論が出来たらと思います．

Computational Scienceには，コンピュータのパワーとアルゴリズムのパワー，さらにプログラミングのパワーのバランスが必要です．ご存知のようにコンピュータパワーの進展はとどまるところを知らず，最近ではペタフロップスマシンが実現し，エクサフロップマシンの議論に入ってきております．このような時代にあって第2，第3のパワーを担うPSEの役割を改めて問い直すこともこの会の重要な役割と考えております．今回，パネル討論ということでPSEを原点から見直す機会にしたいとは思いますが，一回の討論で結論が出るようなものではありません．次回以降も継続的にPSE発展のための議論が出来る場を是非考えて頂きたいと思っております．

最後に，HP作成を始め事務局として活躍してくださった宇都宮大学 川田研究室の海老原龍夫氏，協賛ということで会場をご提供頂いた富士通株式会社，会場，及び懇親会等をアレンジして頂きました富士通株式会社の門岡良昌氏に感謝します．

平成22年9月16日

宇佐見　仁英

玉川大学　国立情報学研究所

第13 回PSE ワークショップ組織委員会

宇佐見仁英，梅谷征雄，門岡良昌，川田重夫，小島義孝，寺元貴幸，丹羽量久，早勢欣和，茨田大輔，日置慎治，前田太陽，松本正己，　宮地英生，眞鍋保彦, Soonwook Hwang

目 次

PSE 閑話.....1
— 田子 精男
問題解決環境のシーニックバイウェイ自然科学/工学から社会科学支援へ.....5
— 前田 太陽, 村上　雅俊, 竹村　敏彦 (関西大学)
植物生産における PSE.....9
— 宇佐見 仁英 (玉川大学/国立情報学研究所)
Eventually consistent な環境での consistent read.....13
— 小橋 博道 (富士通研究所)
TIES を利用した手話教材製作の可能性.....19
— 日置　慎治 (帝塚山大学)
プログラミングコンテスト競技部門「水瓶の恵み」におけるインターネット対戦
システムの構築 21
— 寺元　貴幸(津山高専),中道　義之(沼津高専), 松野　良信(有明高専),
川田　重夫(宇都宮大学大学院)
可視化タグを使った大規模可視化システムの提案.....27
— 宮地 英生 (サイバネットシステム株式会社)
単純な Web ベースエージェントの振る舞い.....31
— 早勢 欣和 (富山高専)
異種グリッド環境におけるジョブ投入を可能とするワークフローツール
の実装と実検.....35
— 田中 義一, 合田　憲人 (国立情報学研究所)
異種グリッドにおけるアプリケーションホスティングサービス(AHS)の試作.....41
— 大西　尚樹, 水澤　有里, 金澤　宏幸, 恒川　隆洋 (富士通)
宇佐見　仁英(玉川大学/国立情報学研究所)
地上デジタル放送用シミュレータ開発における効率的な評価環境の検討.....45
— 梅田 雅敬, 古舘　英樹, 岩松　隆則 (富士通研究所)

第 13 回 PSE ワークショップ プログラム

会場：富士通研究所　川崎工場　一階　岡田記念ホール

9/16(木)　第 1 日目		
12:30-	受付開始	
13:00- 13:10	開会の辞：宇佐見　仁英（玉川大学）	
13:10- 14:40	**特別講演　　座長:門岡　良昌(富士通株式会社)**	
	三浦　謙一 (国立情報学研究所)	“Magic Planet：何でも球面の投影してみよう！”
	田子　精男 (金沢大学)	“PSE 閑話”
14:40- 15:00	休憩（20 分）	
15:00- 17:30	**パネル討論　　モデレータ:川田　重夫(宇都宮大学)** **テーマ:PSE の新たなる発展に向けて**	
	15:00-16:15 (ポジション発表)	モデレータ　川田　重夫(宇都宮大学) ”PSE の過去, 現在, 未来” パネラー 1　小島　義孝(五大開発) ”土木事業における PSE” パネラー 2　前田　太陽(関西大学) ”社会科学と PSE” パネラー 3　丹羽　量久(長崎大学) ”大学教育における PSE” パネラー 4　松本　正巳(米子高専) ”プログラム言語と PSE” パネラー 5　宇佐見　仁英(玉川大学)”植物生産における PSE”
	16:15-16:30	休憩(15 分)
	16:30-17:30	パネル討論
18:00-	懇親会（場所：富士通研究所　川崎工場本館 20 階　ラウンジ C）	

9/17(金)　第２日目

9:00-10:40	**座長：前田　太陽（関西大学）**	
	小橋　博道 (富士通研究所)	“Eventually consistent な環境での consistent read”
	日置　慎治 (帝塚山大学)	“TIES を利用した手話教材製作の可能性”
	寺元　貴幸 (津山高専)	“フロクラミンクコンテスト競技部門「水瓶の恵み」における インターネット対戦システムの構築 ”
	宮地　英生 (サイバネットシステム)	“可視化タグを使った大規模可視化システムの提案”
10:40-10:50	休憩(10 分)	
10:50-12:30	**座長：眞鍋　保彦（沼津高専）**	
	早勢　欣和 (富山高専)	“単純な Web ベースエージェントの振る舞い”
	田中　義一 (国立情報学研究所)	“異種グリッド環境におけるジョブ投入を可能とするワークフローツールの実装と実験”
	大西　尚樹 (富士通)	“異種グリッドにおけるアプリケーションホスティングサービス(AHS)の試作”
	梅田　雅敬 (富士通研究所)	“地上デジタル放送用シミュレータ開発における効率的な評価環境の検討”
12:30-12:40	閉会の辞：川田　重夫(宇都宮大学)	

PSE閑話

MEMORIES OF THE PROBLEM SOLVING ENVIRONMENT
RESEARCH & DEVELOPMENT AND ITS EXPECTATIONS

田子精男
Yoshio Tago
理博 (〒274-0022　神奈川県横浜市栄区庄戸 4-11-16, y.tago1@gmail.com)

The scientific computers, simulation, and their related technology are shown during 1940 – 2020 chronologically. The PSE research and development in Japan are reviewed mainly in connection with simulation for science and engineering. The new concept of the PSE is proposed for Japanese information service industry, which is one of the most important industries for making improvements of productivities in the primary, the secondary, and the third industries. The concept includes the education of programmers, applications of the developed program to other problems, extension to the system of the program, and open community.

***Key Words** : Information Service Industry, Education, System, Open Community,*

1. はじめに

わが国のPSEの研究は梅谷[1]による”DEQSOL”の開発からスタートした．これは物理シミュレーションで使用される偏微分方程式 (Partial Differential Equation: PDE) とその数値解法を専用の記述言語で入力して、FORTRAN言語の数値シミュレーション・プログラムを生成するシステムである．1990年代に、川田[2]はPDEをそのままコンピュータへ入力すると、自動的にその式の離散化やC言語のプログラムを生成するシステム”NCAS”を開発した．

二人の活動に刺激されて、産学の研究者が1998年2月中旬に長岡技術科学大学に集まって、今後のわが国のPSEの研究開発活動について討議した．結論は、PSEの認知度を上げる努力をすることであった．具体的には、

1) PSE Workshopの年一回の開催
2) 積極的な研究発表、
3) 書籍の出版、
4) 国家プロジェクトへの参画、

であった．

第一回のPSE Workshopを1998年の7月に金沢大学で開催し、その当時のわが国の産学のPSEシステムが紹介された．このWorkshopで、産学連携の必要性が認識された．日本計算工学会の産官学共同の研究を推進する研究協力委員会のもとに、研究分科会「計算機シミュレーション支援システムの調査・研究」を設置し、1999年から活動をはじめた．1999年に金沢大学で開催した第二回のPSE Workshopには、当時PSEの研究開発の世界的中心であったPurdue大学から、Prof. S. A. RiceとProf. E. N. Houstis を招待して国際色豊かなWorkshopを開催することができた．その後、大学では宇都宮大学、静岡大学、北海道大学、広島工業大学、企業では㈱構造計画研究所、㈱富士通大分ソフトウェアラボラトリのご協力のもとにこのWorkshopを毎年開催することができた．

ここでは、当初からのPSE Workshopの関係者の一人として、シミュレーションのPSEへ焦点を当てて、PSEの研究開発の想い出を述べる．物理シミュレーションを容易に実施する目的でスタートしたPSEは、その後モノづくりの効率化、コスト削減をめざした．第2章では、このPSEの過去を簡単に振り返る．IT(Information Technology)の進歩とともに変化してきた現在のPSEについて第3章で述べる．わが国の技術として、「モノ」つくりよりも「コト」つくりが重視されてきた．第4章ではコトつくりに関係して第三次産業の情報サービス産業のPSEコンセプトを提案するとともに、PSEへの期待を述べる．

2．PSEのはじまりは数値計算ライブラリから

2．1　シミュレーション

コンピュータが出現してから、10年ごとのおもな科学技術計算コンピュータ、シミュレーションと関連する技術をTable 1に示す．ENIACなどのデジタルコンピュータ出現直後は、シミュレーション・プログラムはコーダによって機械言語（Assembly）でプログラミングされていた1957年に科学技術言語のFORTRANが提供され、プログラミングが科学者の手が届く技術となった．

1950年代にNewton力学ベースの分子動力学法による流体の分子分布関数や、乱数を利用するMonte Carlo法を用いた流体の状態方程式と、Fermi, Pasta, Ulam の問題といわれる1次元の非線形振動子のエネルギーの再帰現象などが、数値シミュレーションのはじまりである．

PSEは1960年代のはじめに、シミュレーションに必要な数値計算ライブラリを用意することからスタートした.

PSE = *PDE*向けソフトウェア　　(1)

その年代に、Prof. Sutherlandがコンピュータの画面上にはじめて一本の線は描いたのが、数値シミュレーションはじめあらゆる分野で、現在必須技術となっているCG (Computer Graphics)のはじまりである．CGは数値シミュレーションの膨大な出力データを可視化して、結果の迅速かつ正確な理解に大きく貢献している.

Table 1. コンピュータ、シミュレーションと関連技術

年代	科学技術計算コンピュータ	シミュレーション	技術
1940 -	ABC, ENIAC	電気回路、物理	機械言語
1950 -	IBM7090, UNIVAC1108	物理（MD, MC, FPU）	FORTRAN
1960 -	CDC6600, IBM 360/370	軽水炉安全性解析、気象予測	FEM, CFD, PSE, CG
1970 -	Cray, Cyber, VP400	航空機設計	ベクター処理
1980 -	Cray, VP, NEC, Hitachi	自動車の衝突解析	PKG, 非線形FEM、計算科学
1990 -	CM, VPP, Blue Gene	環境問題	パラレル、グリッド
2000 -	Roadrunner, 地球シミュレータ	医療	高速CPU
2010 -	京速コンピュータ	可動物体、連成解析	クラウド

２．２　CAE

1980年代に入って、工業製品の製造にCAD (Computer Aided Design), CAM (Computer Aided Manufacturing), CAE (Computer Aided Engineering)が普及するとともに、パッケージ（PKG）のニーズがたかまった．FEM(Finite Element Method)、CFD(Computational Fluid Dynamics), 化学分野で数多くのPKGを使用してシミュレーションが行われた.

このようにシミュレーションの適用範囲が拡大するとともに、Cray, CDC Cyber, 富士通 VP, NEC NEX、日立 Sなどのベクター処理をするスーパーコンピュータが市場にあらわれた． PSEはこのベクター処理をより効率的におこなうための前処理をおこない高速計算に貢献をした．CAE分野で大きく貢献したのが、非線形FEMの自動車の衝突解析であった．この計算は数多くの短いベクターやスカラー計算が大部分でベクター処理向きでなく、パラレル処理を必要とした．これを契機に、パラレル処理や、ベクター／パラレル処理が実用化されることとなった.

２．３　PSE のコンセプト

PSEが注目を集めるようになったのは1980年代である．Prof. Riceを中心にPurdue大学にPSEのグループが出来た．彼らは、PDEのソルバーを中心に活発に研究開発をおこない、多くの論文を発表している.

我が国では、当時日立製作所に在籍していた梅谷征雄先生がPurdue大学へ行って、DEQSOLを開発した．これはベクター型スーパーコンピュータでの実行を意識してFORTRANを直接生成するシステムであった．川田重夫先生は、長岡技科大時代から現在までおもに流体に焦点をあてて自然言語を用いたPDEシステムの構築を精力的に行っている．一方、国のプロジェクトでPSEを開発する努力もされてきた．日本電子計算㈱が中心となって推進したIPA(情報処理振興事業協会)での1998 - 99年のプロジェクト”CAPSE”や、富士通㈱が中心で推進した「サイエンスグリッドNAREGI (NAtional REsearch Gird Initiative)」などがある．これらを含め、2007年までのわが国のPSEの活動は文献[3]で詳しく紹介されている.

しかし、我が国の製造業ではPSEの発展は鈍かった．大きな理由の一つは、我が国の製造業では、部品の標準化、新規システムがその直前まで使用していたシステムや系列会社のシステムとの互換性を重視していて、問題解決のために積極的に新しいシステムを導入する姿勢が欠けていたためであろう．また，研究開発や製造の場では、独自の技術開発というより、技術改善、技術や製品の規格化や標準化、そして技術管理が主体であったことも理由であろう.

HoustisとRice[4]は、PSEのコンセプトを下記の式(2)で表した：

PSE = アプリケーション向け自然言語

+ ソルバー + 知識 + ソフトウェアバス　　(2)

式（2）の右辺第一項は、文字、数学記号、科学記号、コンピュータのアイコンや音声など多様化しつつあるユーザインターフェイスである．第二項は、問題に適用可能なライブラリやPKGの集合体である。しかし、特定の問題を効率よく、低コストで解けるソルバーを探し出すことは難しく、このためには右辺第三項の問題やソルバーの深い知識が必要となる．第四項はコンピュータ、システム、ネットワークなどを一緒に結合する計算に必要なインフラストラクチャである．式(2)の左辺の４つの項のうち、我が国では主に第1項のアプリケーション用自然言語と第2項のソルバーの二つを中心に研究開発されてきた．門岡[5]が流体の速度の実験データを解析するPIV (Particle Image Velocity) web Laboratoryを開発したのは式（1）の第2項、第3項、第4項を統合化したPSEとして注目される.

３．PSEの現状

現在のシミュレーションは、二次産業の製造業で「モノ」つくりの工程の効率化やコストの削減をめざして、生産性の重要な鍵を握っている．シミュレーションはミクロス

コピックレベルとマクロスコピックレベルの現象を対象に行われている．これらのレベルの現象に適っていると思われる基礎方程式を選択して、時間と空間で離散化して代数方程式を作成し、数値的に解いている．

対象とする系が大規模化しており、おもにスーパーコンピュータで計算が実施されてきた．物理モデルや数値計算で、数多くのパラメータが使用されている．これらのパラメータ値をある範囲で少しづつ変化させて数多くの数値シミュレーションを実施する．これらの結果を精査して、パラメータを決定して設計に役立てている．

CPU(Central Processing Unit)の高速化とメモリーの大容量化に伴って、スーパーコンピュータを使用しなくてもPC (Personal Computer) で数値シミュレーションが実施されるようになった．この代表的なシミュレーションは流体のLattice Boltzmann Method (LBM)であろう[6]．

流体のシミュレーションは流れ速度$\vec{v}$の二乗の非線形項をもったNavier - Stokes方程式の数値解を求めている．この非線形項が数値計算を難しくし、さらにレイノルズ数が大きくなると数値的に不安定になる．一方、LBMは、流体粒子の移動方向を格子点のみに限定し、粒子衝突としては有名なBhatnagar―Gross―Krook (BGK)衝突モデルを用いている線形方程式である．

いままでの多くの研究によって、LBMは内部に複雑な形状の障害物をもった非圧縮性流体へ適用できることが明らかになった．心臓シミュレーションなど一部の場合を除くと、現在のPC上でも充分実用的な計算ができることも確認されている．

このようにPCでの数値シミュレーションの実用化の目処がつくと、シミュレーションをする層がCAE専任者から設計者へと拡大した．「モノ」つくりで、設計者が自らCAEを活用することを支援する新しいタイプのPSEである「SOL!BOX」[7]が出現している．

PCの普及とともにPSEが注目されたもう一つの分野がある．第1章の「はじめに」で述べた研究分科会では、1999年にPSEの認知度をアンケート調査した．そのなかで、将来、教育や人材育成へPSEを活用したいという多くの希望があった．教育現場でPCが普及するとともに、多くの教育ツールが開発された．さらに、学生自身がPCを保有するようになると、家で、教室で自分自身の学習の進捗状況に合わせて学習できるようになり、いつでもどこでも学習することができる教育ツールがPSEとして実現されてきた[8]．

４．PSEへの期待

わが国の産業は、1970年代の前半に農林漁業の一次産業が、製造業中心のモノつくりの二次産業へ移行し、現在はサービス業に代表される「コト」つくりの三次産業時代を迎えている．（図 1）．しかし、三次産業の従事者数の割合を欧米の主要先進6カ国と比較すると、まだ低いほうであることがわかる(Table 2).

三次産業は、製造からサービス、ハードウェアからソフトウェア、モノつくりからコトつくり、リアルからバーチャルへの移行である．これに対処するわが国の三次産業従事者は4,138万人で、そのうち情報処理技術者数を調べると約1/50の85万人である．詳細に調べてみると、

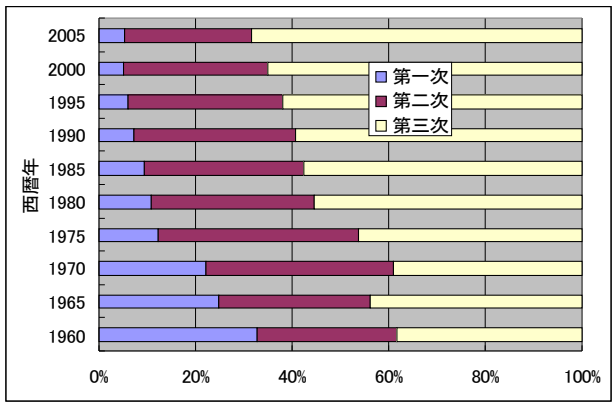


図１　わが国の産業別従業員数の推移
（平成17年（2005年）国勢調査より）

Table 2　主要先進国の産業別就業者の割合（%）

国名	第一次産業	第二次産業	第三次産業
日本	5.1	25.9	67.3
フランス	4	23.7	71.9
ドイツ	2.3	30	67.7
イタリア	4.9	31	64.2
イギリス	1.3	21.5	76.9
カナダ	2.3	21.5	75.8
アメリカ	1.6	20	78.4

（日本：平成17年国勢調査とILO Yearbook of Labour Statistics, 2004より）

情報処理・提供サービスを行うシステムエンジニア（System Engineer: SE）が圧倒的に多く77万3,600人、ソフトウェアの作成を行うプログラマはわずか7万5,900人にすぎない．情報サービス産業は、産業全体の生産性向上の鍵を握っているサービス、コトつくり、ソフトウェア、バーチャルを支える基盤産業だと、いわれているが、上の数字からとても産業全体の生産性を向上する体制が整っているとは思えない．わが国の情報サービス産業は、SEの人口が圧倒的に多く、プログラマの人口はSEの約1/10である．この他に、コーダやスクリプタもいるが、彼等がプログラミングしながら独自でアルゴリズムを開発することは難しい．業務種類別売上高を図2に示す．年間売上金額14.5兆円に対して受注ソフトウェア開発が半分近くを占めている．

情報サービス産業のPSEのコンセプトは、式(2)のようなInformation Technology (IT)レベルではなく、むしろアプリケーション・レベルの要素を考える必要があるのではないだろうか？ 上述のようなわが国の情報サービス産業の

現状を踏まえて、以下のコンセプトを提案する：

PSE = 人材育成 + 問題分析
+ システム化 + 公開・普遍化 (5)

プログラマの育成は急務である．大学入試では、センター試験も含め文系では数学は選択科目である大学が多いこのことは数学や数学的思考が軽視されていることを意味する．プログラマは、自らアルゴリズムを考え出してプログラミングすることが要求され、数学的知識がなくては難しい仕事である．さらに、組込みソフトウェア開発のため、ハードウェアとソフトウェアの両方を熟知した人材を輩出する必要もある．我々は自分の問題だけが解決されればそれで研究開発は全て終了としてきた．この理由はわが国が単一民族のため、単純で、規格化されやすく、その問題以外の多様なニーズが見えないためであろう．受注ソフトウェア開発企業も納品すればそれで全て終わりである．これは、発注企業側の機密性、受注企業側の要員、資金、時間不足のためである．このような障害を乗り越えて、周囲の問題に目を向けて分析して、育成したプログラマの力でシステムを開発する．このシステムをいくつかの問題に適用して、問題解決の範囲を拡大する．また、自然現象や工学現象だけでなく、社会科学や人文科学も対象とする．図2からわかるようにソフトウェアプロダクツとしめしているパッケージ・ソフトウェア(PKG)の比率が少ない．2005年の実績では、

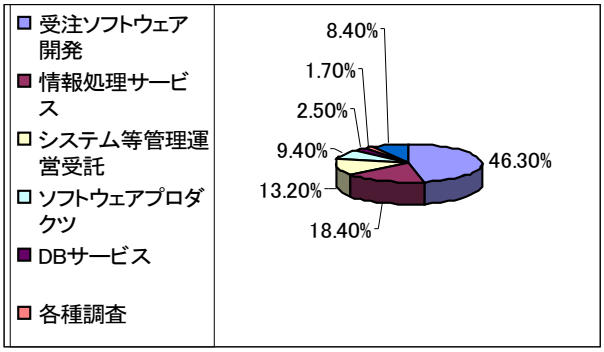


図2 わが国の情報サービス産業の業務種類別売上比率
（経産省平成17年度特定サービス産業実態調査より）

わが国のPKGの世界シェアは数%にも満たない．単一目的の受注ソフトウェアや研究開発ソフトウェアを上で述べたように普遍化して、入出力を決定してシステム化する．このシステムを公開して誰でも使用できるようにする．それらの実績を踏まえて、PKGとして世界に乗り出すことができるであろう．

三次産業の教育PSEとして、金融分野の教育がある．従来までの金融教育は場当たり主義で単発的に実施されてきた．このため、以前から、欧米のような体系的な金融教育の必要性が指摘されている．PSEで金融教育ツールを開発して、欧米並みに金融のリスクを理解できる国民を育成できれば長年の懸念が解消されるであろう．

その一方で、これからも依然としてモノつくりは基盤産業として重要である．しかし、PSEは、モノつくりの生産性向上よりも、安全性が高い、高機能の製品の研究開発をターゲットとすべきである．たとえば、可動する物体を含むシステムや、異なる物理現象や工学現象を接続する連成解析、ミクロとマクロなモデルを繋げる連成解析へ挑戦していく．

最後に、わが国の産官学の研究者とRiceやHoustisが執筆した書籍「PSE BOOK」[9]の出版によって、1998年に決定した長岡技科大でのアクション・アイテムを全て実施したことをお知らせするとともに皆様に感謝したい．

謝辞：この講演の機会を与えてくれた、川田重夫先生、宇佐見仁英先生、門岡良昌博士はじめ、PSE Workshopの幹事の皆様に心から感謝いたします．

参考文献

1) Umetani, Y, DEQSOL, A Numerical Simulation Language for Vector / Parallel Processors, Proc. IFIP TCP2 / WG 22.5, pp.147 – 1641, 1985

2) 川田重夫：数値シミュレーションプログラムを作るプログラム―プログラミングからの開放を目指す問題解決環境(PSE)の世界―シミュレーション、 Vol. 18, No. 2, pp. 35 – 39, 1999

3)「特集PSEの過去、現在、未来」、計算工学、Vol.11, No.1, pp.4-26, 2007

4) Houstis, E. N. and Rice, J, A, (private communication, 1999). これは㈱構造計画研究所によって支援された以下の研究 “On the Future of Problem- Solving Environments”のなかでも述べられている．

5) 門岡良昌, 田子精男：グリッドと PIV 仮想研究所, 計算機統計学, 第 16 巻, 1 号, pp. 59 - 76（2004）

6) Wolf-Gladrow, D. A.: Lattice-Gas Cellular Automata and Lattice Boltzmann Models – An Introduction, Lecture Notes in Mathematics **1725** (Springer, Berlin,2000)

7) 伊藤昇平、赤池 茂、古木建吾、早川 要：エンジニアリングPSE、計算工学、Vol.11, No. 1, pp. 9 – 12, 2007

8) たとえば、第12回PSE Workshop論文集、http://www.redc.nagasaki-u.ac.jp/pse2009/ に掲載されている三つの教育関連の論文

9) 川田重夫、田子精男、梅谷征雄、南 多善共編：PSE BOOK シミュレーション科学における問題解決のための環境［基礎編］、［応用編］、培風館、 (2005)

問題解決環境のシーニックバイウェイ 自然科学/工学から社会科学支援へ

Scenic byway in problem solving environments shift supporting domain from natural science / engineering to social science

前田太陽[1)]，村上雅俊[2)]，竹村敏彦[3)]
Taiyo Maeda
1)博士（理学）　関西大学 ソシオネットワーク戦略研究機構　ポストドクトラルフェロー
(〒564-8680　大阪府吹田市山手町 3 丁目 3 番 35 号, r078002@kansai-u.ac.jp)
2)博士（経済）関西大学 ソシオネットワーク戦略研究機構　助教
3)博士（経済）関西大学 ソシオネットワーク戦略研究機構　助教

This paper presents Problem Solving Environments(PSEs) for social science in our organization. We describe our studies in social science with micro data: web based survey, supporting system for both analysis and simulation. We show the feature of the approach, and discuss what part do PSEs play in social science.

***Key Words** : Problem Solving Environments, Social Science, Web based Survey, Supporting System*

1. はじめに

これまで工学や自然科学の分野において，様々な対象をターゲットとした研究者支援環境：問題解決環境（Problem Solving Environments : PSEs）の研究が行われてきた．現在では従来のプログラミング支援のみでなく，計算支援，可視化支援，教育，高性能計算など多岐にわたる．これらの研究は，対象や方法は異なるが，研究者の専門用語で解くべき問題に取り組める計算機システム(A PSE is a computer system that provides all the computational facilities needed to solve a target class of problems.[1]) という点で一貫していると認識できる．現在，このようなPSEの概念が社会科学の分野においても必要であることがわかってきた．社会科学の研究者の多くは，人々が成す現象やそのプロセスを解明することに重点をおく．このため調査や実験によるデータの収集，統計や問題に特化した評価方法による対象の状態把握，現象となる要因や関係の発見，これらを解決するための理論や手法などが彼らの研究対象である．社会科学分野において，データ処理や統計分析，モデル検証のシミュレーションなどの広い範囲で計算機が利用され，研究者の計算機利用は必要不可欠である．しかしながら彼らの研究においても，計算機を利用した多くの作業，プログラミングやソフトウウェア独自のスクリプト習得などが必要であり，本来の研究以外の作業が発生する．このようにこれまでのPSEでの課題は，社会科学の研究者にとっても共通の課題である．

そこで本稿は，これまで行ってきた社会科学分野での研究促進に関連するいくつかの事例や開発した問題解決環境のシステム，提案システムを紹介し，これまであまり議論されてなかった社会科学分野でPSEがどのような役割を担えるかを議論する．

2．社会データの入手支援

この章では，社会データに関するミクロデータ（個票）の取得に着目し，様々なシステムから得られるデータについて述べる．まず社会科学の対象となるデータの取得にはいくつかの困難な点がある．社会調査などのデータ取得の際，何らかの知見を得るために

・サンプルに関する内容（選び方，有効数など）
・データ取得の条件
・データからのモデル化と検証方法

などの調査設計を検討する．この後，サンプルのデータの真偽検証や異常値の発見など，分析にかかるまでの前処理を行い，有効サンプルを得る．調査設計からモデル化までのプロセスを単純化した図を図１に示す．図の手順では，それぞれのプロセスで社会科学に関するニーズと，作業を進める上でのニーズが存在する．そこで支援システムの研究として，データ取得からモデル化までのプロセスに着目した．この中で計算機側から研究者への技術的な支援は，ミクロデータ取得の効率化，分析を効率的に行うための取得データの電子化，データ取得後の作業の効率化である．まずデータ取得をどのように行うかを述べる．

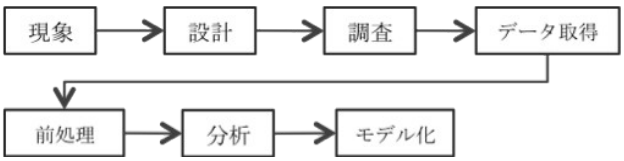


図１　調査設計からモデル化までのプロセス

２．１　既存サービスと社会データ

システムを利用した社会データ取得の一例としてTwitterでの事例を紹介する．このサービスはSNS（Social Network Service）の１つとされるが，利用者にとってリアルタイムで他者のメッセージを閲覧できることが特徴である．定期的に閲覧の必要があるブログと異なるため，利用者にとってこのようなサービスは，新たな情報取得のコミュニケーションとして利用価値がある．なりすましなどの問題があるものの，利用者は，特定の人物からのメッセージ（例えば政治家や宇宙飛行士：http://politter.com/，http://twitter.com/astro_soichi）を受け取ることができるため利用者数は多い．

一方，サービス提供者側では多くのデータの取得がシステムを通じて可能である．このサービスで得られたデータから社会的なネットワーク構造を議論する研究［2］がなされている．Twitterを技術的に見れば，高機能なチャットと理解できるが，そこから得られたデータの活用によって学術的な成果を生み出している．これより，システムを利用したサービスが社会科学の研究を促進していることが確認できる．このような背景から，データ取得の効率化としてシステムを利用したデータ取得に着手した．

２．２　Web ベースアンケート調査

近年，アンケート調査においてWebベースのシステムによるデータ取得が盛んになってきた．特に企業が提供するアンケートでは，協力者の管理情報から特定の情報を基にアンケート対象として割り付けが可能なことや，パネルデータの取得も可能であることが多い．Webアンケートに特化した協力者の属性を管理する補助システムを利用することでこのような利点を得ていると予想できる．我々はこのWebベースのアンケート調査を利用し，竹村らによる金融に関する調査［3］と村上らによる年金に関する調査［4］を行い，調査票とデータの一部をWeb上で公開［5］し，研究者に対し調査データとデータ利用環境を整えている．

一方，個人でもHTMLのフォームで簡単なアンケート調査を行うことは可能である．得られる調査結果についていくつかの問題はある反面，アンケートを自由に設計できる利点を持つ．そこで，既存のサービスmixiから独自のアンケートシステムへの回答を協力してもらう既存サービスとの連携を行った．独自のアンケートシステムでは協力者の回答と回答時間を取得した．既存のサービスと独自システムの連携によって得られたデータから心理分析が行えた［6］．この結果から，次節のWebベースで調査支援を行うシステムを考案した．

２．３　Web ベース調査支援システム

社会の振る舞いについて，個々の人が状況に応じてどのような行動を行ったかどうかを調べたい場合，社会実験が有効な手段の一つである．経済分野では，実験室に人を呼んで行うタイプの実験［7］や，近年の盛んに行われている計算機を利用した仮想実験（例えば， Zurich Toolbox for Readymade Economic Experiments(Z-Tree)［8］，Veconlab Software［9］による仮想実験）による研究が進められている．計算機上での仮想的な実験は，Webベースアンケート調査と同じく取得データがデジタル化されているため迅速に統計処理を進めることができる．我々もWebベースの実験環境提供システムを構築し，施設選択実験を行った［10］．このシステムは，実験者と協力者がブラウザを利用した仮想環境内で集まり，実験者が設計したシナリオをHTMLで配信し，協力者の振る舞いをシステム側で記録しながらを他の協力者にそれぞれのとった行動を反映させる．例えば，商品購入時の混雑を調べたいとすると，複数のレジの前で自分が選んだレジがどれだけ混雑するかを他者の情報を更新することで認識させ，混雑した場合に空いているレジに移動するかどうかを検証することができる．なお，協力者個々の行動を情報更新無く独立して取得方法が一般的なWebベースのアンケートシステムである．

開発したシステムを利用し実験を行ったところ，実験シナリオをシステムに保存するため，実験の協力者が集まれば，同様の実験シナリオの迅速に再現できたことと少し異なるシナリオの実験を容易に行えることが，システムの大きな特徴であることがわかった．実際，開発後2009年に行った実験を，関西大学総合情報学部にて，2010年8月に異なる協力者対して同じ実験を2回行っている．

３．モデル化支援としての分析支援

社会科学の研究者がデータ取得後に共通して行うことは解きたい現象のモデル化である．これはどの分野においてもプロセスの解明や現象の再現に必要不可欠であるため共通の課題となっている．社会科学の場合，先に述べたデータの再現性をとることが難しいため，推計統計によりモデルに即した変数の推定や検定を行う．そこでモデル化を行うために必要な作業である統計分析を支援するシステムを提案する．

分析を行う場合，一般的に統合統計ソフトウェアを利用することが現在の主流である．商用ではSTATAやSPSSが利用されている．これらを利用し分析する際の前提条件は，整ったデータを入力することが前提となる．このため，研究者はミクロデータのいくつかの変数の変換や

変数を用いた評価基準の結果を基に分析を進める．多くのソフトウェアは，データ変換のための機能を持ち，多くの解析や検定の機能を持つ反面，必要な作業を迅速に進めるためにはソフトウェアの習得が必要で，研究者は試行錯誤しながら分析作業を進めなければならない．本来の分析に集中するためには，ある程度の操作の知識やソフトウェア個別のスクリプトの知識を持つことが前提条件となる．そこで統計分析のための前処理を支援するために，スクリプト作成作業効率を向上させるシステムの開発を行っている．利用者はGUIから必要な変換を選び，変換する変数名を入力し，その変数に対して行いたい変換を選択すれば，システムがSTATA用のデータ処理用スクリプトを自動生成する仕様としている．これは従来のPSEでのプログラム自動生成での技術がそのまま活用できる．現在のところ，いくつかの事例の統計分析の際に必要なデータ変換方法や作業を形式化し，スクリプト生成を行うサブシステムを試作している．一般に，データ変換の方法や過去の変換方法は，利用者が個別に得る知識となる．このシステムの開発によって，この知識を研究者で共有し，解析用のスクリプトを再利用することで作業時間の向上を目指している．

４．シミュレーション支援としての計算実行と結果検証支援

モデルが決定しその検証を行う際やモデルから実用的なツールへと発展させるためには，計算機実験やシミュレーションが必要となる．この際，研究者を苦しめる点が，プログラミング，複数回の計算の実行，結果検証である．まず，モデルをコード化するためにアルゴリズムの決定や計算ライブラリの選択とともに図3に示すいくつかのプロセスを乗り越えなければならない．

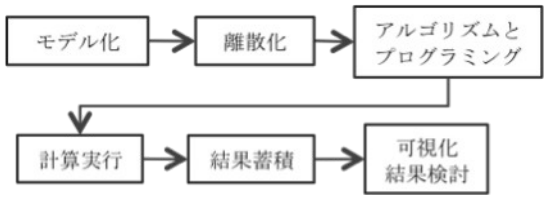


図3　モデル化から結果検討までのプロセス

プログラミングの作業効率化としはプログラム自動生成が有効であるが，特定の問題に対して何らかの式などが決定されている必要がある．意思決定や相互作用を表すモデルの形式化が確立していないため，モデルに対し適切なアルゴリズムを採用しなければならない．このため，現状でプログラミング自動生成は難しい．そこで他のプロセスに着目した．

社会科学シミュレーションの場合，確率的なシミュレーションを行うことが一般的で，現象の変化に関係する入力パラメータの数と，そのパラメータの水準数によって多数の計算実行が必要である．その後，さらに結果を複数回確認する作業が発生する．そこで，研究者が行うこれらの作業の効率化を促進させる統合システム［11］の開発を行った． このシステムは，確率的シミュレーションの分散計算，その結果の自動登録とグラフ化支援の機能を持つ．表１に示す協力者による作業時間結果からそれぞれのプロセスにおいて支援システムが有効であった．さらに１ケースあたりの計算時間が短いシミュレーションにおいては，計算実行の高速化よりも結果データへの迅速な参照が重要であることがわかった．複数のプロセスが必要な研究の作業では，それぞれのプロセスでの作業バランスが良いことが全体の作業効率に影響することが確認できる．

Table 1 Results of elapsed time per 15 cases

Measurement category	Not using system (sec.)	Using system (sec.)
Elapsed time for computing	610.3	78.5
Registering work time	266.7	1.9
Work time for screening simulation data set	1495.5	356.8
Total time	2372.5	437.2

５．まとめ

本稿では，いくつかの事例を挙げ，社会科学における支援システムとしてのPSEを紹介し，そのニーズに対する役割と特徴を述べた．これまで工学・自然科学で行われてきたPSEのアプローチは社会科学の分野においても共通点が存在することが確認できた．いくつかの事例から，最先端の技術ではなくとも必要とされる技術やシステムは多数あり，社会科科学の多方面で計算機支援の効果が予想できる．

PSEの対象となる分野の発展とともに，これらの分野をターゲットとしたPSEの相互の発展を今後の課題とする．

謝辞

本研究は，平成20年度−21年度文部科学省「人文学及び社会科学における共同研究拠点の整備の推進事業」および平成22年度「特色ある共同研究拠点の整備の推進事業」による助成を受けて行った研究成果である．

参考文献

1) S. Gallopoulos, E. Houstis and J. Rice, Computer as Thinker/Doer: Problem-Solving Environments for Computational Science, IEEE Computational Science and Engineering, Vol.1, Issue.2, pp. 11-23, 1994.

2) B. A. Huberman, D. M. Romero, F. Wu, Social Networks that Matter: Twitter Under the Microscope, SSRN-id1313405, pp.1-9, 2008

3) T. Takemura, T. Kozu, An Empirical Analysis on Individuals' Deposit-withdrawal Behavior's Using Data Collected through

a Web-based Survey, Eurasian Journal of Business and Economics, Vol.2, Issue 4, pp27-41, 2009

4) 村上雅俊，四方理人，国民年金納付者行動Web アンケート結果の概要と探索的検討Ⅱ −年金保険料納付者・未納者が持つ意識の差異の検討を中心に−，関西大学ソシオネットワーク戦略研究センターディスカッションペーパーシリーズ，Vol.102, pp.1-15, 2010

5) ソシオネットワーク戦略研究機構 http://www.kansai-u.ac.jp/global/education/riss.html

6) 曹陽，前田太陽，村田忠彦，池之本欣哉，インターネット調査の活用実験−政策研究のための社会科学と情報システムの融合−，関西大学政策グリッドコンピューティング実験センターディスカッションペーパーシリーズ，Vol.23，pp.1-13，2008

7) R. Selten, T. Chmura, T. Pitz, S. Kube and M. Schreckenberg, Commuter Route Choice Behavior. Game and Economic Behavior. Vol.58, Issue 2, pp.394-406. 2007.

8) C. A. Holt, Economic Science: An Experimental Approach for Teaching and Research, Southern Economic Journal, Vol.69, No. 4, pp.755-771, 2003.

9) U. Fischbacher, z-Tree: Zurich Toolbox for Ready-made Economic Experiments, Experimental Economics, Vol. 10, No. 2, pp.171-178, 2007.

10).….T. Maeda, T. Murata, D. Kotaka, S. Matsumoto, Y. Cao, Social Simulation Based on Human Behavioral Data Collected by Web-Based Experimental System, Journal of Convergence Information Technology, Vol.5, No.4 June, pp.141-151, 2010

11). T. Maeda, Y. Aoki, T. Murata, Construction of PSE System for Developing Reinforcement Learning Algorithms, Journal of Convergence Information Technology, Vol.5, No.4, pp.130-140, 2010

植物生産におけるPSE

A PROBLEM SOLVING ENVIROMENTS IN PLANT PRODUCTION

宇佐見 仁英
Hitohide USAMI

玉川大学 学術研究所(〒194-8610 東京都町田市玉川学園 6-1-1 usami@lab.tamagawa.ac.jp)

Recently, the plant factory comes to the front of the new Japanese industry. The plant factories have many advantages compare with ordinary green houses, especially closed type plant factory will be changing to new type pharmacy factory by using genetic modification (GM) technology. We describe concept of the intelligent plant factory and apply problem solving environments technology to plant production in closed type plant factory. Plant environment data, cultivation data, plant growth knowhow and optimizing the plant growth environment are very important issues for GM plant growth in the intelligent plant factory. Those knowledge get gather and integrate work with collaboration on the research community by using e-Science framework.

***Key Words** : Problem Solving Environments, Intelligent plant Factory, plant systems biology*

1. はじめに

我が国の産業構造の転換，特に農業生産に関する早急な変革が求められている．なかでも未来型農業生産として期待されているのが植物工場である．昨今の経済危機に対する「新経済成長戦略2008」においても植物工場の普及・拡大が求められている．植物工場は，栽培環境を人工的に管理した栽培システムであり，天候等の自然界の影響を受けずに高付加価値食物（無農薬の新鮮野菜等）を安定的に供給できるのが最大の利点である．しかしながら，人工環境を維持するための電力コスト等が非常に高いのが現状であり，路地物と比べてコスト面で不利となっている．中長期的な視点に立てば，葉菜類・果菜類を中心に安心・安全な食物供給の有力な手段であり，先進国における新しい都市型農業として発展していく可能性が高い．特に，光環境制御によるポリフェノール等の有効成分を増やした健康補助食品や遺伝子組み換え技術を活用したワクチン等の薬の製造など高付加価値な生産物を大量・高品質・安定的に製造することができる．

我が国における植物工場の現状は，植物工場の先進国オランダに比べ生産性が半分以下に留まっており，生産性の向上が急務となっている．また，エネルギーコストのウェイトの高い光環境に対しては，日本の得意分野である発光ダイオード（LED）を大幅に取り入れることにより，コストダウンの余地を残している．LED技術の進歩は著しく，エネルギー効率，価格面で優れたものが研究開発され，近い将来，世の中の照明が全面的にこの手の照明器具に置き換わる可能性もある．特に，LEDの得意な赤色，青色等は，植物の光感受性を考慮した人工環境に適しており，高効率な生産システムとなる可能性がある．このようなLED環境下での植物生産性の向上には，光特性を考慮した植物生産手法の確立が重要となる．

本論では，このような背景を踏まえて，植物の生育状況を非破壊で科学的に計測するとともに，植物の生育モデルをベースとした生育環境の最適化のための問題解決環境（PSE）の適用について述べる．

2. 植物工場の現状

農林水産省，経済産業省ともに，植物工場の普及促進を推進しており，植物工場を工場とみるか圃場と見るかで所管が変わる可能性があり，主導権争いは当面続くものと思われる．

2.1 植物工場とは

我が国の施設園芸では，簡単なビニールハウスタイプからガラス張りの太陽光を基本としたもの，或いは窓の無い倉庫型の完全閉鎖型の植物工場（図１）まで，幅広い施設が展開されている．

図１玉川大学の植物工場

既に商用生産を開始している施設もあるが，どの程度までの施設を植物工場と定義するかは明確にされていない．一般的には，「施設内で植物の生育環境（光，温度，湿度，二酸化炭素濃度，空気の流れ，肥料，水分等）を制御して栽培を行う施設園芸のうち，環境及び生育のモニタリングを基礎として，高度な環境制御と生育予測を行うことにより，野菜等の植物の周年・計画生産が可能な栽培施設」（農林水産省，植物工場ワーキンググループ報告書より）とされている．

このような観点から植物工場を分類すると，温室等をベースに太陽光の利用を基本とした半閉鎖環境で，雨天・曇天時の補光や夏季の高温抑制技術等により周年・計画生産を行う「太陽光併用型」と太陽光を使わずにLEDや蛍光灯のような人工光源の利用を基本とした閉鎖環境で，環境の制御により周年・計画生産を行う「完全閉鎖型」の２種類となる．「太陽光併用型」は，光源を主として太陽光とすることから，「完全閉鎖型」に比べてエネルギーコストが低く高効率である．一方，「完全閉鎖型」は環境を制御できるため，同一品種の通年栽培が可能であり，路地物では困難な収穫期と需要期が大幅に異なった場合での植物の通年生産も可能となる．

2.2 植物工場の普及と課題

三菱総合研究所が実施した委託調査によると野菜，花類（苗を含む）を生産している植物工場は全国で50箇所（完全閉鎖型：34 ，太陽光併用型：16 カ所）にのぼると言う．昨年度，農林水産省が３倍増計画を打ち出しており，早急に拡大して行くものと思われる．

しかしながら，普及には多くの課題がある．最大の課題は建設・運用コストである．植物工場の建設投資が路地物に比べて極めて大きく，特に完全閉鎖型では，空調人工照明等のランニングコストが莫大にかかるのが現状である．農林水産省では，初期投資コストを下げるために建設費の半額を補助金で支援する施策を打ち出している．また，全国の遊休工場を利用して，建設コストを低減化する試みがなされている．

2.3 植物工場の展望

課題で述べた，建設コスト，ランニングコストに見合う経済的に採算が取れる栽培品目を如何に増やして行けるかが普及のポイントとなる．現状では，レタスを中心とした葉菜類や苗等，生育期間の短い品目が中心であるが，形質転換(遺伝子組換え)作物を使った二次代謝物による薬用植物の生産など高付加価値な植物生産を進めていけば展望が開けて行けそうな状況である．

特に，植物工場の生産品は無農薬で安心・安全である点を消費者へ全面的にアピールしてイメージ向上を図って行く必要がある．しかしながら，イメージアップには「植物工場」という言葉自体，消費者受けしない言葉であり，欧米で使われている「グリーンハウス」等の響きの良い言葉に変える必要がると思っている．

3. 植物工場の知能化

3.1 植物の健康状態の把握

植物工場の知能化の最大のポイントは，植物の健康状態の把握にあると言える．人間では，体温，脈拍，脳波等の測定により，かなりの確度で健康状態を把握することが出来る．一方，植物は，元来，検診による健康管理という概念が無いため，健康状態を把握するための方法論が確立されていないのが現状である．健康観察の人間との対比を図２に示す．

図２健康観察の人間との対比

人間の勘と経験に頼った植物栽培から，工場生産のための均一で高品質な植物生産に転換しなければならない工場内の温度，湿度，光強度，液肥の養分状態等の物理的な環境は，それぞれのセンサーを活用することによりデジタルな数字として容易に把握することができる．しかしながら，植物の生育状態，健康状態を数値として的確に捉えるのは難しく，経験と勘に頼っているのが現状である．特に，完全閉鎖型植物工場では蛍光灯や赤色，青色等のLEDを使った人工的な光環境となるので，目視での健康状態の把握が難しくなってくる．

LED等の人工照明下でも適切な生育状況，植物の活性状態が把握するために非破壊で低侵襲な方法として，植物の電気的特性を測定し植物の活動状態を評価する手法がある．実験は，実際の植物工場と同じ環境を自由に設定して各種実験ができるように人工気象器（インキュベータ）を用いた環境で実施する．開発中の植物の活動状態測定システムの構成概念を図３に示す．

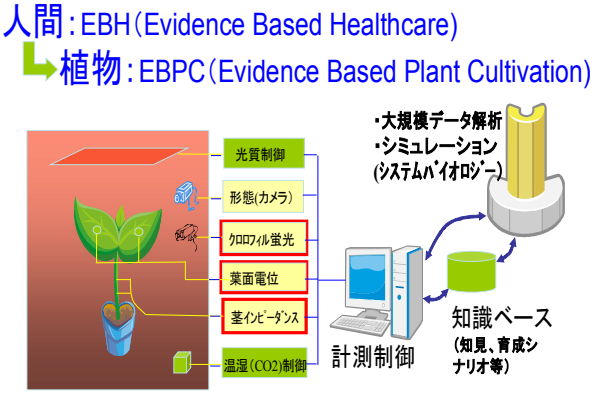


図３　植物の活動状態測定

3.2 知能化

植物工場の生産性を上げるとともにエネルギー削減による生産コストの低減が，我が国の農業が国際競争に勝ち残って行くための必須条件である．そのために，最高水準の人工知能技術を駆使した生産システムを構築する必要がある．

(1) 栽培での知識処理

温湿度等の環境データとともに，目的とする植物の生育データ（草丈，本葉数，葉色・・）を収集する．収集したデータから，データマイニング機能を使って生育のための有益な知識，特に栽培環境に関する知見を抽出し知識ベース化する．断片的な知識が網羅的に集まったら一つの生育シナリオとして体系化する．体系化されたシナリオをシナリオ DB として蓄積管理する．シナリオ作成のための知識処理プロセスを図４に示す．

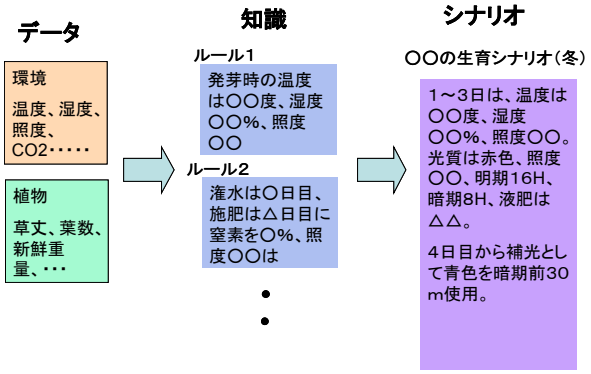


図４　シナリオ作成のための知識処理プロセス

(2) 植物工場の知能化（シナリオ活用）

生育シナリオを記述するための良い記述言語が無い．知識ベースシステムを用いて，手続き的処理としてのワークフローとプロセスでの判断としてのルールベースと併用することにより，シナリオの記述を試みる．

シナリオベースの植物工場の知能化イメージを図５に示す．図のように，生育植物と目標（レタスの短期栽培等）を入力することによって，最適なシナリオを選択し，シナリオに沿って環境がセットされるとともに，植物モデルとの照合により，生育状態の適切な把握とシナリオ修正が適時行われる知能型システムとなる．

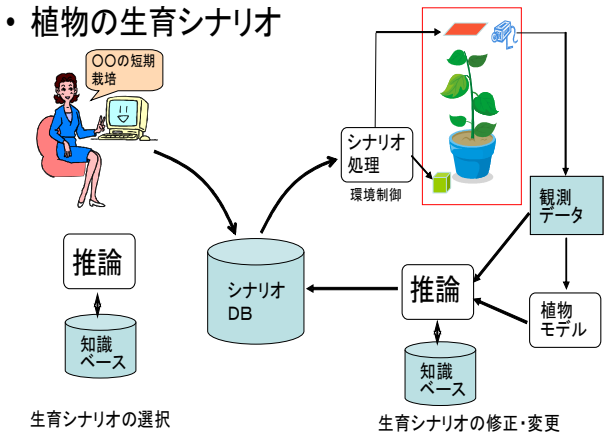


図５　植物工場の知能化

3.3 植物モデル

植物の観測データを収集しても，植物の内部状態を正確に推測することは難しい．特に，２次代謝物が最大となる生育環境条件を実験で導きだすには，網羅的な実験が必要であり組合せの爆発が起きる可能性もあり，実験計画を立案するのも困難となる．そこで，我々は，植物の内部状態を出来るだけ詳細にモデル化する手法を採用し，観測データと植物モデルとを照合することでより正確な内部状態の推定，モデルシミュレーションにより最適環境条件の絞り込みが可能と考えている．

(1) 植物の環境ストレス

植物から見た場合の環境ストレスを植物の環境ストレスとして図６に示す．

図６　植物心理モデル

(2) ストレス処理

植物の代謝物産生のメカニズムを解明するために，トランスクリプトミクス，メタボロミクスなどのゲノム機能科学からシステムバイオロジーへの展開が図られている．外部ストレスに対するシグナル伝達系を図７に示す．

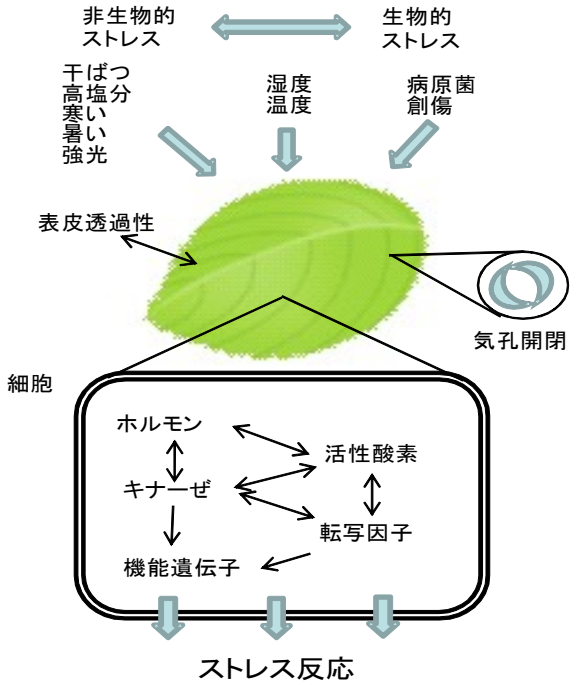


図７　外部ストレスに対するシグナル伝達
(Plant Stress Biology　p69 より)

(3) システムバイオロジー

植物は動物などに比べて優れた代謝物の多様性を有しており，食料，医薬品・健康機能成分などの形質転換作物が注目され，植物の有用代謝産物が利用されている．

しかしながら，2 次代謝産物の生成経路，産物量を最大とするための環境因子，さらには環境データなど，それぞれが新たに研究開始された分野であり未知な点が多い．特に，生物は個々の遺伝子発現だけでは生命現象を捉えることは難しく，生命をシステムとして統合的に捉えることが重要であり，そのためのツールとしてのシステムバイオロジーが注目を集めている．

我々は，植物機能の解明と生育環境の最適化のためにシステムバイオロジーの導入を検討している．システムバイオロジーツールには，E-Cell, Cell Illustrator, System Biology Workbench, Virtual Cell など幾つかある．

システムバイオロジーの全体像を図８に示す．

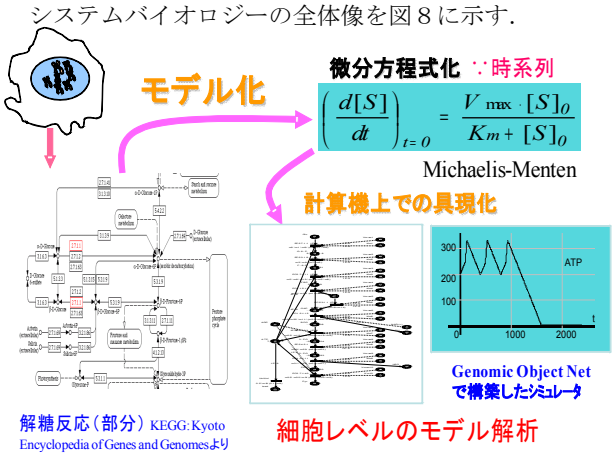


図８　システムバイオロジーの全体像

図の左下に示すネットワークはパスウェイと呼ばれ，遺伝子の制御やシグナル伝達の経路を表しており，数万の反応からなる複雑な構造をもつものもある．しかしながら，これらの個々の反応は，速度式によって数学的に表現することが出来る．これらの研究は 1900 年代初期より研究されており，図８の右上のミカエルス・メンテンの式として良く知られている．右下は，生命現象の基本である細胞内でのエネルギー生産，つまり解糖系のモデルとシミュレーション結果である．

(4) 問題解決環境としてのシステムバイオロジーツール

個々の反応を微分方程式として書き下すとしても，簡単なモデルでも数千の構成要素となり，全体の整合性を取りながらシミュレーションをすることは容易ではない．また，多くの反応式がミカエルス・メンテンの式で書けたとしても，非線形，或いは統計的・確率的な振る舞いをする反応もある．そのような場合，新たなる関数やプログラムを簡単にプラグインして全体のシミュレーションに供する必要がある．システムバイオロジーツールとしては，このように従来型の問題解決環境（PSE）の機能を有していることが重要となってきている．多くのツールが EXCEL の関数定義レベルであり，本格的なモデル式を組み込めるツールが必要である．一方では，反応式の定数の決定など，シミュレーションをするための基礎データの収集も進んでいないのが現状であり，多くの研究課題が残されている．

4. まとめ

植物の生体測定を中心に，植物工場に向けた基本的な測定環境の構築ができた．

今後は，実験データの収集，解析手法の確立を図り，植物工場の最適化設計に役立つレベルに高めるとともに「知能型植物生産システムに向けた植物計測と生育環境の制御」へと発展させていく必要がある．特に，測定データと抽出された知見(知識）により科学的根拠に基づく生育（EBPC: Evidence Based Plant Cultivation)の確立が急務と考えている．EBPC は，近年話題の EBH（Evidence Based Healthcare）の植物版であり，近未来の植物工場における重要な技術要素の一つとなる物と思われる．さらに，最先端のシステムバイオロジー技術を駆使して，植物の生育環境の最適化，或いは，植物の２次代謝物を最大とするためのストレス（刺激）を理論的に導出するなどに展開していく必要がある．

謝辞：本論文執筆にあたって，玉川大学渡邊教授，大橋准教授，大野研究員，布施研究員にご指導・ご協力を頂きました．また，e-Science に関しては，国立情報学研究所三浦教授，合田教授，大阪大学松田教授，産業技術総合研究所関口部門長，東京工業大学松岡教授，高エネルギー加速器研究機構佐々木教授，ならびに NAREGI 関係者にご指導・ご支援を頂きました．

尚，本研究の一部は，文部科学省の平成 22 年度科学技術試験研究委託事業「研究コミュニティ形成のための資源連携技術に関する研究（アプリケーション共有方式のユースケースと実証に関する研究）」の成果である．

参考文献

[1] 宇佐見仁英，知能型植物工場における問題解決環境～ e - サイエンスによる研究コミュニティの形成 ～，PSE Workshop2009，pp20-25，2009
[2] 宇佐見仁英，座古朝美，布施政好，渡邊博之：” 電気インピーダンスによる植物の状態評価”， 日本生物環境工学講演要旨，2009 年福岡大会 （2009 年 9 月）
[3] Plant Stress Biology -From Genomics to Systems Biology -, Heribert Hirt, WILEY-BLACKWELL, August 2009 .
[4] Plant Systems Biology (Methods in Molecular Biology), Dmitry A. Belostotsky, Humana Press, 2009
[5] 生体電位とコミュニケション，大藪多可志，勝部昭明，海文堂(2009 年 4 月）
[6] E-Cell; http://www.e-cell.org
[7] Cell Illustrator: http://www.cellillustratorl.com
[8] System Biology Workbench: http://www.sbw.sourceforgel.net
[9] Virtual Cell: http://www.vcell.org

Eventually consistent な環境でのconsistent read

A METHOD FOR CONSISTENT READ IN EVENTUAL CONSISTENCY

小橋博道[1)]
Hiromichi Kobashi
1) 株式会社富士通研究所(〒211-8588 川崎市中原区上小田中 4-1-1 E-Mail: h kobashi@jp.fujitsu.com)

In this paper, I describe how realize consistent read in eventual consistency. Eventual consistency is widely used to achieve scalability in distributed key-value store. I implemented consistent read in my distributed key-value store named x4u. In x4u, a commitment protocol is used to get agreement among multi nodes. By using the commitment protocol, consistent read can be executed. Consistent read is enough fast even if there are other eventual read requests.

***Key Words** : Distributed Key-Value store, Eventual Consistency, Logical Clock*

1. はじめに

Eventual Consistency とは，あるデータストアにおいて，全てのレプリカに即座に更新を適用して一貫性を保証するのではなく，最終的に一貫性が収束することを保証する一貫性モデルである．CAP 定理 [1] は eventual consistency の存在意義を示す一指針である．CAP 定理とはConsistency (C) / Availability (A) / Partition Tolerance (P) は同時に二つまでしか満たすことができない，というものである．Consistency とは，データの整合性が常に即時に保たれるこという．もし，レプリカがあるなら，レプリカも常に一貫性のある状態に保つことを意味する．Consistency を諦めるとは，eventual consistency にすることである．Availability とは，常に利用可能であることを意味する．たとえ障害が起きても，システムとしては正常に動作し，必要とあらばレプリカを再構築する．Availability を諦めるとは，障害時にシステムが止まることを許容することである．Partition tolerance とはネットワーク分断でノードが2つのグループに分割されたとしても，それぞれのグループで処理を継続できることをいう．Partition toleranceを諦めるとは，広域分散を諦めることである．

近年盛り上がりを見せている，クラウドコンピューティング(以下，クラウド) は，広域で多数のノードをインターネットを介してサービスを提供する形態のコンピューティング環境である．このようなクラウド環境では，データストアは，以下の理由で，CAP 定理のAとP を採用し，C を諦めるのが一般的である．つまり，consistency を緩めて，eventual consistency にしなければならない．まず，クラウドでは，ネットワーク分断が起こることは一般的であり，local area network のような高品質のネットワークを期待することができない．また，多数のノードを使っているため，ノード故障は日常茶飯事である．ノード故障が起こるたびに，サービスが止まっていては使い物にならない．Eventual consistencyとは，高レイテンシ且つ信頼性の低いネットワーク環境において， Availability とPartitioning Tolerance を実現するためのトレードオフとも言える．

クラウドのデータストアとしては，分散キーバリューストア(分散KVS) が広く利用されている．分散KVSは，取り扱うデータをキーバリューに単純化したデータ保管方式である．キーの値によって格納を担当するノードを決定し，データを多数のノードに分散して格納する．多数のノードで処理を分担するので性能が高い．さらに1 つのデータを別のサーバに複製して格納することでサーバが1 台故障してもデータ損失は起きない．そして，スケーラビリティのためにmultimaster アプローチのレプリケーションを採用し，レプリカの一貫性にeventual consistency を適用しているものが多い．このため，分散KVS で取得できる値は，それが全てのレプリカで一貫している保証はない．分散KVS のReadは，Eventual Read だと言える．

しかしながら，eventual consistency において，consistent read を実現することは困難である．Consistent read とは，ある時点において，全てのレプリカで一貫している値を取得することである．通常，分散環境において，consistent read を実現する場合には，Paxos [2] のような排他制御機構が必要である．その排他制御を使うと，先述のCAP 定理でいうところの，C とA を採用することと同様である．つまり，ネットワーク分断に対処できない．

本論文では，eventually consistent な分散KVS において，従来のeventual read を邪魔せずに，consistent readを実現する手法について述べる．本手法では，consistent read を実現するために，確定プロトコルを用いる．確定プロトコルとは，分散KVS を構成するノードで，一貫性を合意するためのプロトコルである．

本論文の構成を述べる．第2 章で，確定プロトコルを実装した，分散KVS x4u について述べる．第3 章で，確定プロトコルについて述べる．第4 章で，確定プロトコルを用いたconsistent read の実現方法を示す．第5章で，consistent read について，性能評価をし，第6章でまとめる．

２．X4u

分散KVS であるx4u について，この章で述べる．なお，ここで用いる用語は[3] に従う．

分散KVS として，x4u はDynamo と似ている[4] ．インタフェイスとして，Read とWrite がある．Writeはキーとバリューを指定することにより，適切なノードにバリューを保管する．適切なノードは，Chord [5] を利用することで，キーのハッシュ値から特定することが可能である x4u では，レプリカは3つ持つ．レプリカは，キーで特定したノードのConsistent hash 上で隣接する2 ノードにもupdate を送ることで実現する．Read はキーを指定することにより，バリューを持つ3つノードを特定できる．その中の1つからバリューを取得することができる．

高可用性とスケーラビリティを実現するために，x4uは Multimaster でEventual Consistency を採用する．update を一つのobject に対して行い，それを伝播させるSingle-Master 方式では，マスタが落ちると可用性がなくなる．Multimaster は，マスタを設けず，複数のレプリカに対して独立にupdate を送出することで高可用性を実現する．また，object のupdate の際に，全てのレプリカ間で，strong consistency を保証するための制御を行わない．Eventual Consistency にすることで，スケーラビリティを確保する．Eventual Consistency は，ある時点におけるレプリカ間の不一致を許容するので，Read リクエストを送るレプリカによっては異なる値を返すことがある．

さらに，x4u は次のような分散環境での困難を解決しなければならない．まず，分散環境では，Update の転送はネットワーク遅延やルーティングなどの影響を受けるため，レプリカ間のUpdate の到着順は，全レプリカで一意である保証はない．単純に到着順でupdate するとレプリカ間のデータの矛盾が起きる場合がある．次に，あるUpdate をリクエストしたノードが一時的に応答せず，要求がタイムアウトする場合がある．このとき，依頼したUpdate が完了して落ちたのか，完了しないで落ちたのか判断するのができない．

これら困難を解決するために，分散KVS は可換性と冪等性を持つべきである．可換性があれば，update の到着順序は関係がない．全てのupdate が適切なobjectに届きさえすれば，レプリカの一貫性を保つことができる．冪等性があれば，障害時にupdate されたかどうか，確認しなくてよい．わからない場合は，もう一度update することで，レプリカの一貫性を保つことができる．Algorithm 1 に可換性と冪等性について示す．uiはあるupdate を意味し，＊ はアップデートの適用を意味する．obj は，あるobject である．可換性の例では，update u0 と，update u1 の更新順序が入れ替わっても，最終的に得ることができる結果が同じであることを示している．冪等の例では，update u0 の適用が，一度の場合と二度の場合の結果が同じことを示している．

分散KVS に，可換性と冪等性を持たせるには，バリューを，そのバリューを更新するための更新関数のペアで扱えばよい．例えば，バリューを「タイムスタンプつき命令履歴」とし，更新関数を「タイムスタンプに基づいた位置への命令の挿入かつ再実行」とするモデルである．x4u ではこのモデルを採用している．

Algorithm 1 可換性と冪等性の例

var ui : an update which has characteristics of commutative and idempotent

obj ∗ u0 ∗ u1 = obj ∗ u1 ∗ uo // 可換性

obj ∗ u0 = obj ∗ u0 ∗ u0 // 冪等性

Algorithm 2 operation の定義

var op ⇐ (f, t) : an update operation

f : update function for replica
t : timestamp of this update

Algorithm 3 update replica の定義

```
var h : array of operations
proc ur(h, op)
  if h ∋ op then
    h[op.timestamp] ⇐ op
    sort history by time stamp
end if
return h
```

Algorithm 4 x4u における可換性と冪等性の例

var op1 ⇐ (f1, t1)

var op2 ⇐ (f2, t2)

var h0 ⇐ [] //empty array

ur(ur(h0, op1), op2) = ur(ur(h0, op2), op1) {(1) commutative}
ur(h0, op1) = ur(ur(h0, op1), op1)) {(2) idempotent}

このモデルを実現するx4u は，Operation Transfer タイプの分散KVS である．x4u では，全てのupdate をupdate function とtimestamp の組とする．x4u では，この組をoperation と呼ぶ．timestamp は，レセプタでoperation に挿入する．Receptor とはユーザのリクエストを受け付けたノードである．x4u はmulti-homing採用しているため，x4uのどのノードでもリクエストを受けることが可能であるtimestamp はlogical clock [6] の一種を採用し，x4u 内部で一意に順序を決定できる．operation はキーごとに履歴を管理する．履歴のscheduling はtimestamp-base で行う．

Write はoperation を対応するキーの履歴に挿入することである．この時，履歴の最新の位置ではないところに挿入することがある．これをconflict と呼ぶ．x4uはconflictが発生すると，detect してrepair する．もし履歴の途中に新しいoperation が挿入されると，履歴を再実行し，バリューを更新する．これにより可換性を実現する．また，同じtimestamp を持つoperation が来た場合，後から届いたoperation を破棄する．これにより冪等性を実現する．Read はその時点で履歴に積まれているoperation を実行した値を読むことである．このため，全てのoperation が到着していないときには，その途中の値を読むことになる．

Algorithm 2 とAlgorithm 3 にoperation の定義とreplica のupdate 手法について示す．あるキーの履歴hは，operationのarray である．あるoperation op は，update 関数f とタイムスタンプt のタプルである．もし，あるノードがupdateを受け取れば，そのノードは，適切なキーの履歴に，そのupdate を適用する．この処理がupdate replica ur である．もし，op がh に既に届いていれば，そのop は無視する．op が新しいoperation であれば，h に挿入し，タイムスタンプでソートする．

Algorithm 4 は，x4u における可換性と冪等性の例である．Algorithm 4 (1) では，operation op1・op2 の順で到着し場合(左) と，operation op2 ・op1 の順で到着した場合(右) の例を表している．この例では，update が上記の順番で到着しても，replica としての値が同じになることを示している．これが可換性である．Algorithm4 (2) では，op1 が一度だけ適用された場合(左) とop1が二度適用された場合(右) の例を表している．この例でも，replica は同じ値になる．これが冪等である．

履歴は有限空間であるため，必要でなくなったoperationを消さなくてはならない．これをcommit と呼ぶ．x4u では，commit のために，確定プロトコルを用いる．次の章で，確定プロトコルについて述べる．

３．確定プロトコル

履歴は無限にのばすことができないため，不要になったoperation を履歴から消す必要がある．あるoperationが不要になるのは，そのoperation よりも古いoperation が発効されないことが保証されたときである．Eventually Consistent な環境で，あるoperationを消して良いか判断するためのプロトコルが確定プロトコルである．

Algorithm 5 ローカル確定タイムスタンプ算出方法
var requests // timestamps of executing write requests
var ltimestamp // local timestamp
var ctimestamp // current logical clock

if requests ≠ ∅ **then**

ltimestamp ⇐ min(requests) - Δt

else

ltimestamp ⇐ ctimestamp

end if

Algorithm 6 全体確定タイムスタンプ算出方法
var ltimestamps : array of local timestamp
var wtimestamp : whole commit timestamp

ltimestamps[mine] ⇐ ltimestamp

wtimestamp ⇐ min(ltimestamps)

３．１　プロトコル仕様

確定プロトコルでは，各ノードのローカル確定タイムスタンプのリストを交換する．このリストをタイムスタンプリスト，またはトークンと呼ぶ．

ローカル確定タイムスタンプはAlgorithm 5 のように計算できる．まず，各ノードで実行中Writeリクエストのタイムスタンプを，そのWriteリクエストが終了するまで管理する．実行中Writeリクエストがある場合は，そのリクエストのタイムスタンプからΔt 分引いたものがローカル確定タイムスタンプである．なお，Δt は論理クロックの刻み幅である．複数実行中Writeリクエストがある場合は，それらリクエストの最小タイムスタンプからΔt を引く．また，実行中Writeリクエストがない場合は，その時点の論理タイムスタンプを採用する．

また，各ノードはローカルのタイムスタンプリストを持っているとする．タイムスタンプリストを受け取ったノードは，ローカルのタイムスタンプリストを受け取ったタイムスタンプリストで更新する．次に，ローカルのタイムスタンプリスト中の自身のエントリをそのときのローカル確定タイプスタンプで更新する．全体確定タイムスタンプはローカルタイムスタンプリストの最小値を求めることで算出できる(Algorithm 6)．その後，ローカルのタイムスタンプリストを新しいタイムスタンプリストとして，次のノードに回す．

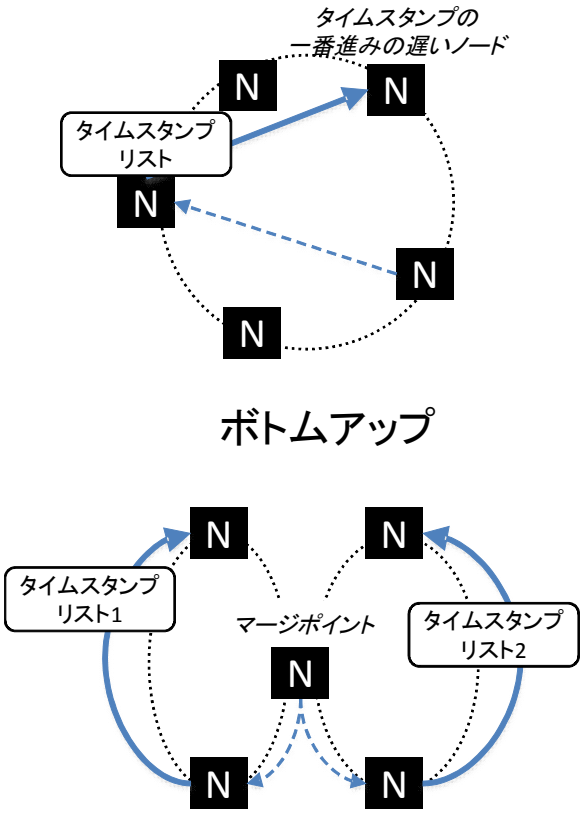


図１ 確定プロトコルのルーティング方法

３．２　ルーティング

タイムスタンプリストの回し方について説明する．(図1)

一番単純なものは，タイムスタンプリストの中でローカル確定タイムスタンプの進みが一番遅いノードに回す方法である．このルーティングをボトムアップルーティングと呼ぶ．Writeリクエストが多い場合には，効率よく各ノードの全体確定タイムスタンプを共有することができる．また，ノードをランダムに選択し，回すランダムルーティングも可能である．

他のルーティングとして，パーティショニングルーティングがある．パーティショニングルーティングでは全ノードをいくつかのグループに分割し，各グループでタイムスタンプリストを回す．分割したグループは，他のグループとノードを一つ以上共有しなくてはならない．このノードをマージポイントと呼ぶ．マージポイントでは属している全グループのローカルタイムスタンプリストを持つ．あるグループのタイムスタンプリストが回ってくると，そのグループのローカルタイムスタンプリストを更新する．全グループのローカルタイムスタンプリストの自身のエントリもローカル確定タイムスタンプで更新する．回送すべきタイムスタンプリストの自身のエントリは，他グループのローカルタイムスタンプリストの最小値で更新する．このルーティングの特徴はマージポイント以外は全てに関するタイムスタンプリストを持つ必要がなく，メモリ空間を節約できることと一回の転送量を少なくできることが利点である．なお，各グループ内のルーティングは今までにあげたいずれもルーティングでも構わない．

４．Consistent Read

Consistent Readについて，モデルと実装を本章で述べる．

４．１　モデル

バリューに一貫性があるとは，ある時点においてバリューの値が確定することである．マルチマスタを採用しているシステムでは，リクエストの到着順序は不順である．そのため，一貫性のあるバリューを取得するには，その時点よりも古いリクエストが到着することがなく，バリューが変更されないことを保証しなくてならない．

x4u では確定プロトコルを使って，consistent read を実現する．Consistent read は，自身のタイムスタンプまで確定プロトコルによって確定するまで待つ．確定すればその自身まで命令を実行し，そのバリューを返す．

このモデルを用いれば，KVS 上での複数キーに対するconsistent read，つまりスナップショットも可能である．全てのキーに対してconsistent read 命令を同じタイムスタンプで発行する．これらの命令は，先ほどと同様に確定プロトコルで確定するまで待ち，確定した時点のバリューを返す．これらバリューを全て回収することで，ある時点におけるスナップショットが可能になる．

４．２　実装

Consistent を実現する方法を述べる．

Consistent read に通常のリードと同様，レセプタで受け付けられた時点で，タイムスタンプを振る．キーに基づいてリクエストを適切なノードに転送する．通常のリードは，即時に履歴内のoperation を実行した結果をバリューとするが，これはEventual な状態のバリューである．Consistent read は，マーカを履歴に挿入する．マーカは確定しても消すことができないoperation である．マーカは確定すると，自身までの命令を実行し，バリューを計算する．計算したバリューをマーカとして再度保存する．Consistent read は自身が挿入したマーカがバリューに変換されていれば，それを確定したバリューとして返す．

スナップショットも同様に実行可能である．スナップショットでは，全ての履歴にマーカを挿入する．確定したマーカを集める全て集めることでスナップショットが実現できる．

５．評価

Consistent Read について実験・測定を行った．

５．１　実験環境

x4u はPython とTwisted を使って実装した．通信はJSONを使ってシリアライズする．本評価で利用したマシンスペックは表1 の通りである．このマシンを最大48 台用いて評価した．

表１ 実験環境マシンスペック

型名	FUJITSU PRIMERGY RX100 S5
CPU	Intel Xeon E3110 (3GHz)
メモリ	4GB
ディスク	500GB (SATA)

５．２ 測定

a) レイテンシ

Consistent Read のレイテンシを評価する．

図2 が結果である．横軸がノード数，縦軸がレイテンシである．キーの数は8192 であり，リードするキーはその中からランダムに選択する．また，レセプタもランダムに選択する．バリューのサイズは32byte である．それぞれの台数について100 回試行し，その平均レイテンシをその台数でのレイテンシとした．確定プロトコルはボトムアップルーティングで，トークン数は3 である．

この結果，12 ノードのときにconsistent read はeventual readの10 倍ほど遅いことがわかる． Consistent read と eventual read との差は，ノード数が増えるほど広がる．これは確定プロトコルのトークン数を固定しているためである．本検証では，consistent readのリクエスト以外は発行していない．このため，ボトムアップルーティングを採用しているが，実際のルーティング効率は，DHT 上の次のノードにトークン回すような，シンプルなルーティングと変わらない．そのため，確定までの時間times は，確定プロトコルが全ノードを一周するにかかる時間(timec) と，リクエストが受け取ってから確定プロトコルのトークンが最初届くまでの平均時間(timep) の和である．ノード数をn ，トークン交換にかかる時間をtimet とすると，それぞれ

$$timec = timet * n \quad (1)$$

$$timep = timec/2 \quad (2)$$

$$times = timec + timep = timet * n * 1.5 \quad (3)$$

となり，台数に比例することがわかる．

b) レイテンシの改善

times は確定プロトコルのトークン数を変えることで，改善可能である．

図3 の横軸は確定プロトコルのトークン数，縦軸にレイテンシである．12 ノードの場合は，確定プロトコルのトークン数は3 もしくは4 のときにconsistent readのレイテンシが低くく，最適であることがわかる．

しかし，このトークン数はノード数と関連しており，一意に決めることは難しい．

我々の実験では，48 ノードまでであれば，ボトムアップルーティングとランダムルーティングを1 つずつ使えば

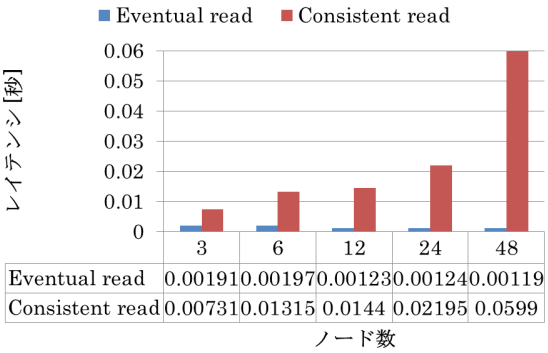


図２ Consistent read のレイテンシ

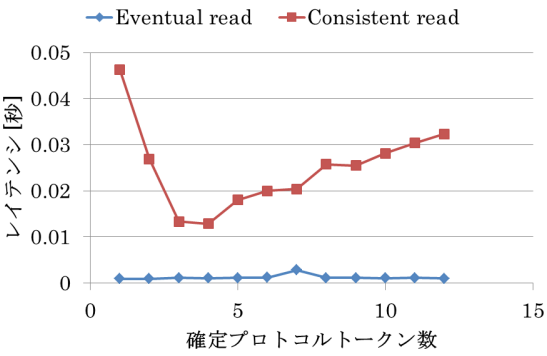


図３ 確定プロトコルのトークン数とレイテンシ

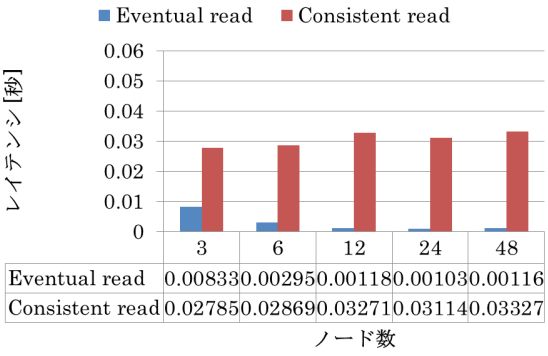


図４ ボトムアップルーティングとランダムルーティングを混在させた場合のレイテンシ

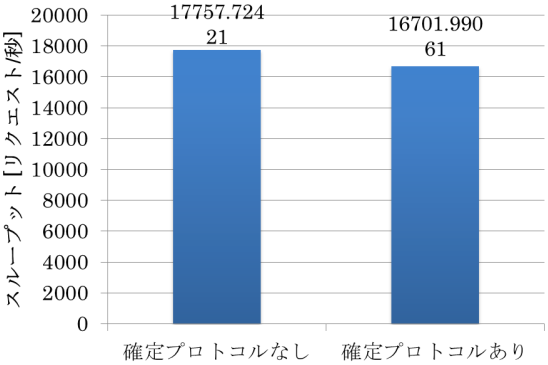


図５ 確定プロトコルがあるときとないときのスループットの違い

安定したconsistent read が実現できることを確認している．図4 がその結果である．48 ノードでも33msec でconsistent read が実現できる．これは，図2で，ボトムアップルーティングで3 トークン使ったときの59msec よりも，約2 倍高速である．

c) 確定プロトコルのスループットへの影響

確定プロトコルのeventual read のスループットに対する影響について示す．

履歴を用いない分散KVS では，確定プロトコルはなく，その分だけ処理が高速に処理できると予想できる．この比較では，確定プロトコルを動かさない場合と，確定プロトコルを動かした場合の違いを比較することにより，確定プロトコルがスループットに与える影響を検証する．

図5 が測定結果である．この測定では，ノード数12に固定し，確定プロトコルのルーティングは，ボトムアップルーティングとランダムルーティングを使用した．

確定プロトコルがないときのeventual read のスループットは，17,757[リクエスト/秒] であるのに対し，確定プロトコルが動いている時のeventual read のスループットは，16,702[リクエスト/秒] である．この結果より，確定プロトコルが動いているときは，スループットに関し，6% 程度の性能劣化が発生する．

しかし，12 台で6% 程度のスループット損失あれば，あと1 台マシンを追加すれば，確定プロトコルがない場合のスループットにスケールする．これは，x4u はスケーラブルなためである．よって，この程度の性能劣化は問題がないと言える．

６．まとめ

開発した分散KVS x4u では，履歴を確定プロトコルを用いて適切に管理することで，可換性と冪等性を持った分散データストアを実現することができる．また，確定プロトコルを用いることで，Eventually consistent な分散KVS 上において，Consistent Read を実現可能である．今後，耐故障性を含めた確定プロトコルの拡張を行っていく．

参考文献

1) S. Gilbert and N. Lynch. Brewer’s conjecture and the feasibility of consistent, available, partitiontolerant web services, ACM SIGACT News. Vol.33, No. 2, June 2002, pp. 51-59

2) L. Lamport. The Part-Time Parliament. ACM Transactions on Computer Systems, 16, (2), May 1998, 133-169

3) Y. Saito and M. Shapiro. Optimistic Replication. ACM Computing Surveys (CSUR), Vol. 37, No.1, March 2005, pp. 42-81.

4) G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A. Lakshman, A. Pilchin, S. Sivasubramanian, P. Vosshall and W. Vogels. Dynamo: Amazon’s Highly Available Key-Value Store. ACM SIGOPS Operating Systems Review. Vol. 41, No. 6, December 2007, pp. 205-220.

5) I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, Chord: A scalable peer-topeer lookup service for internet applications. Applications, Technologies, Architectures, and Protocols for Computer Communication. San Diego, California, USA, 2001, pp. 149-160.

6) L. Lamport. Time, clocks, and the ordering of events in a distributed system. Communications of the ACM, Vol. 21, No. 7, July 1978, pp. 558-565.

TIESを利用した手話教材製作の可能性

TEACHING MATERIALS OF SIGN LANGUAGE BY THE USE OF TIES

日置慎治
Shinji Hioki
工博　帝塚山大学　経営情報学部・情報教育研究センター
(〒631-8501　奈良市帝塚山 7-1-1,hioki@tezukayama-u.ac.jp)

TIES (Tezukayama Internet Educational Service) is an environment of e-learning developed by the Tezukayama University. About 80 universities not only in Japan but also in foreign coutries are now using TIES for their educations. This paper will report the possibility to develop a teaching materials of "Sign Language" by the use of TIES.

Key Words *: E-Learning, TIES, Sign Language*

はじめに

帝塚山大学（奈良市）では，1997年ごろから，通常の対面型授業を補完する手段の一つとしてTIES[1)]と呼ぶｅラーニング環境を整備してきた．早い時期から，当システムを無料で他大学等に開放し，教材の相互利用を促進してきた結果，現在では国内のみならず国外の大学からの利用も含め約８０の機関が利用する高等教育の一大コミュニティとなっている．

図１　TIESのトップ画面

TIESには，教材をよりリッチにし，様々な方法で学習者のモチベーションを高め、また，理解度を深めるための各種のツールが整備されている．教材作製者はこれらのツールの中から，自分の目的とする教材を作るために必要な機能を取捨選択し，活用することができる．

TIESではこれまで，語学や社会学系科目，自然科学系科目など多くの講義で活発に利用されてきた．この度，生涯教育という側面から、手話への応用の可能性を探ることを目的として、当該研究をスタートさせた。本論文はその最初の報告としての位置づけである。

２．TIESの教材作成ツール

ここでは，TIESの教材作成ツールのうちいくつか紹介することにより，TIESという環境について説明したい。

２．１　教材提示

学習者に教材を提示するｅラーニングには必須の機能である。PDF，WordやPowerPointのファイルだけでなく、一般に利用可能な多くの形式のファイルを教材として学習者に提示することが可能である．教材作製者は講義時に使用する資料等をTIESに置いておき，学習者は予習や復習時にも自由に教材にアクセスすることができる．

２．２　レポート提出

各種レポート提出機能であり、締め切り等を設定することが出来る。

２．３　ドリル・クイズ

問題演習により学習成果を確認するための機能である。チャレンジ回数や合格最低点等を設定する事により、学習者のモチベーションを高めることができる。

２．４　ライブ機能

TIESの画面を表示させながら授業を行う際に、メニューにある「録画ボタン」を押すと、画面や音声、USBカメラを接続している場合には映像も全て記録される機能がある。記録されたファイルは録画後自動的に教材内に保存されるため、後でアップロードする必要はない。これを使って、授業の録画をファイルとしてTIES教材の一部に提示させることで、やむを得ない事情により欠席した学生の学習に

役立てる事も可能である。

２．５　手話教材への応用

手話の説明には、アニメーションや動画が効果的であると考えられる。上記の機能を用いることにより、例えば動画を教材提示し、それに関する問題を解かせる、という作業を通じて理解度を高めることができると思われる。しかしながら、教材提示の動画と問題演習が別々に展開されている事から、軽快な展開に無理があると感じていた。

この度、TIESに新機能が加わり、動画の任意の場所に問題演習を任意回数組み込むことが可能となった。この機能により従来不可能であった“軽快”な展開の可能性が開けてきた。

３．TIES新機能Glexaビデオを使った手話教材

Glexa[2)]とは「対面授業の拡張を目的としたマルチメディアWebラーニングプラットフォーム」であり、目的とするところがTIESと親和性が高いようである。その証拠に、TIESの新機能としてスムーズに導入され、現在活用され始めている。

ここでは、TIESに組み込まれたGlexaビデオを使った動画教材の製作を取り上げ、手話教材への応用の可能性を探っていきたい。

図２はTIESの教材編集画面である。

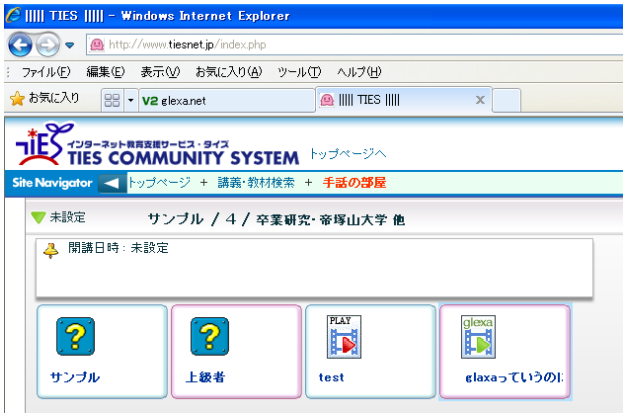


図２　TIES の教材編集画面

Glexaビデオを編集・作成するために編集対象の動画を指定すると、図３のような初期画面になる。

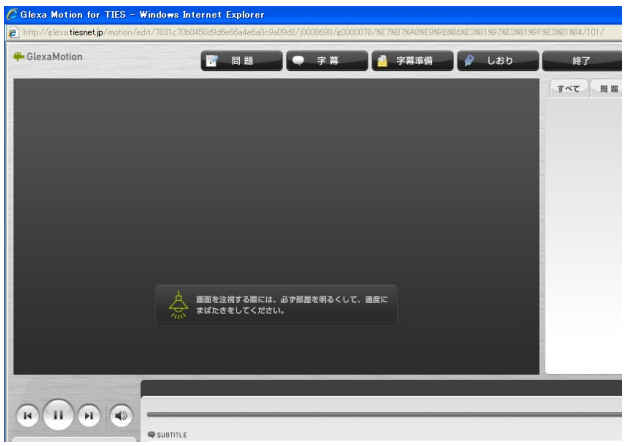


図３　TIES の Glexa ビデオ編集画面

動画を再生しつつ、問題作成の箇所で一旦動画を止め、上部メニュー左側の「問題」ボタンをクリックすると、図４ような選択問題作成画面になる。

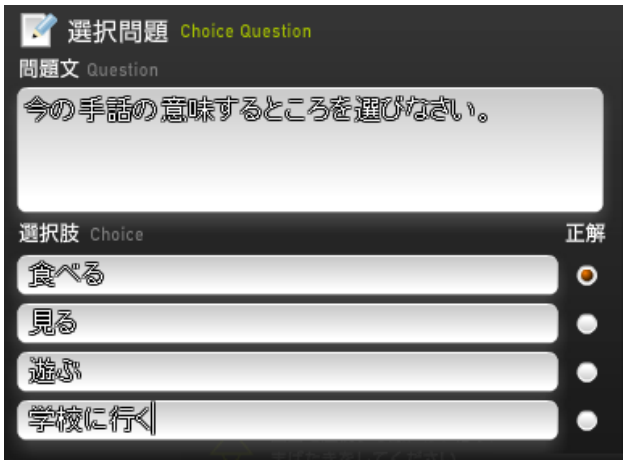


図４　選択問題作成画面

このように、動画を再生しつつ、適切な箇所で次々に問題を組み込んでいくことができる。オプションとして早送りの禁止等を設定する事も可能である。

４．まとめ

TIESの新機能を用いて、手話に関するｅラーニング教材作成の可能性を調べている．まずは第一段階として、教材として使い物になるのかどうかを様々な観点からチェックしたい。次に教材としての有効性を調べてみたいと考えている。

当教材が、手話を勉強しようとする方の適切なＰＳＥになれば幸いである。

参考文献

1) 日置慎治，中嶋航一：問題解決環境としての教育，日経サイエンス，シミュレーション科学への招待
http://www.tiesnet.jp/
2) http://glexa.net/

プログラミングコンテスト競技部門「水瓶の恵み」におけるインターネット対戦システムの構築

THE DEVELOPMENT OF THE INTERNET PLAY TYPE PROGRAMMING CONTEST SYSTEM

寺元貴幸[1)], 中道義之[2)] , 松野 良信[3)] , 川田重夫[4)]
Takayuki Teramoto, Yoshiyuki Nakamichi, Yoshinobu Matsuno
and Shigeo Kawata

1)講師 津山高専 情報工学科(〒708-8509 津山市沼624-1,Tel. 0868-24-8289, teramoto@tsuyama-ct.ac.jp)
2) 講師 沼津高専 総合情報センター(〒410-8501 静岡県沼津市大岡3600,Tel. 055-921-2700, nakamiti@numazu-ct.ac.jp)
3) 准教授 有明高専 電子情報工学科(〒836-8585 大牟田市東萩尾町150,Tel. 0944-53-8873, yoshi@ariake-nct.ac.jp)
4)工博 宇都宮大学大学院 工学研究科（〒321-8585 栃木県宇都宮市陽東 7-1-2, kwt@cc.utsunomiya-u.ac.jp）

The 21st programming contest will be held on October 16 - 17, in Kochi. We give this paper about a system construction and use of the competition section of the 21st programming contest. According to the game rule, we have compiled the competition software. To win this game, the participant must compete against the AI robot of other players to win this game. So, we write the program to let our robot finding water and distributing water on the ground to get the score. We constructed the system that the exercise of the match was possible through the Internet. Each participant can play without being influenced by other participants.

Key Words : Programming Contest, Internet, Security

1. はじめに

以前から高専生を対象とした全国規模のコンテストとしてロボットコンテスト（高専ロボコン）とプログラミングコンテスト（高専プロコン[1)]）が有名である．今年度は第21回プログラミングコンテストとして高知高専を主管校として平成22年10月16日17日の両日，高知市文化プラザかるぽーとで開催が予定されている．高専のプログラミングコンテストは自由，課題そして競技の3部門から構成されている．

近年は高専以外でも様々なプログラミングコンテストが数多く開催されるようになってきた．プログラミングコンテストは大きく分類すると，自作のプログラムのアイデアを競う創作系（高専プロコン課題部門・自由部門U-20プログラミングコンテスト[2)]など）とプログラムの性能を競う競技系に分けられる．競技系はさらに，パソコン甲子園[3)]やACM大学対抗プロコン[4)]のように時間内にどれだけ多くのプログラムを正しく作成できるかを競う問題解決型と，ロボコードのように決められルールに従って試合を行う対戦型に分けられる．高専プロコンの競技部門は毎年テーマが変更されルールだけでなく競技システムまでが全て刷新される比重に珍しいコンテストである．参加者は，毎年違ったテーマの競技に参加することになり，過去の資産だけではなかなか勝ち上がる事はできない．しかし，これはまた主催者側にとっても非常に大きな負荷となり，毎年運用システムを構築しなければならない．ここ数年競技に関するルールや運用方針の概要は全国プログラミングコンテスト委員会がとりまとめ実際の競技システムの開発や運用は主催校が主体となって開発を行ってきた．このスタイルは開催地の独自性を生かしたシステム開発が可能なこと，大会前に十分な調整時間をとることができるというメリットがある．反面開発や運用のノウハウが継承されないため，開催地に過度の負担を強いることになった．

しかし21回大会においては早々に主管校から開発困難という連絡があり，プロコン委員会で開発する事となった．そこで，ここ数回のプロコンで使用したフレームワークをベースに開発を行う事とした．また今回はできるだけ多くの参加者に質の高いプログラムで本選に臨んでもらうよう，あらかじめインターネットで練習ができるシステムを公開することとした．本稿では，スタンドアローンでの練習用プログラム，ならびにインターネットを経由した練習システムについて報告する．

２．競技概要と競技ルール

競技は図１に示すゲームフィールド上に設置された「水瓶」（青色で表現されたセル）に蓄えられた水をうまく利用し，いかに治水面積を多く取得できるかをリアルタイムで競うゲームである．各チームは自チームのロボット

を3台所有しており，このロボットを使って水を各地へ配水する．ロボットにより配給できた場所を自分のエリアとして確保でき，このエリアを拡大することで生活圏を拡大することを目的としている．

●競技用のフィールドおよび用語

セル：領地の最小範囲，形は六角形をしており辺が接続されている隣のセルに移動可能．

フィールド：セルを組み合わせて作成した競技場所もしくはその全体．

エリア：各チームが自身の領土として確保したセル．自エリア・敵エリアのように使う．

ロボット：競技参加者が操作可能な機械．1チームあたり3台保有し命令により移動・チャージ・配水などの操作が可能．

命令：ロボットに送る移動等のコマンドの総称．

移動：ロボットが隣接する6方向いずれかのセルに位置を変える動作．

占有：同一セルに1ロボットのみが存在する状態．自チームのロボットでも2台以上いる状態は占有とは言わない．

リョウマ：水の最小単位．

チャージ：水瓶でロボットが水を蓄える動作．

配水：ロボットからセルに水を移す動作．

吸い取り：敵エリアから水を吸い取る動作．

貯蓄水量：ロボットに乗せている水の量．

フレーム：競技進行の最小単位（1秒あたり60フレーム）．各命令を実行するにはあらかじめ定められたフレーム数を消費しないと次の命令が実行できない．たとえば10フレーム必要な命令は，最初のフレームで命令が実行されその後の9フレームが待ち時間となる．

セッション：フレームの集合体として3秒（180フレーム）をまとめたもの．この単位で各チームの命令をとりまとめ全チームの競技が進行する．

ゲーム：1試合のこと．セッションの集合体として1試合を構成する．

繰り越し：セッション内で命令の待ち時間が完了せず次のセッションに待ち時間が持ち越されること．

待ち時間倍率：セルの地形により移動（吸い取り）・配水などの「動作」に必要なフレーム数がこの倍率で変化する．

●競技の基本ルール

- 1つが六角形のセルを敷き詰め全体としても六角形の盤面（フィールド）で競技を行う．盤面の大きさは常に固定とし競技では変更しない．
- 1辺のセル数は21に固定する．
- 原則6チームの対戦とする，対戦チームが6チームに満たない場合は主催者側で用意したAIが対戦チームとして入る．
- 競技時間は5〜15分．
- 各チームは3台のロボットを保有しそのロボットを操作して水瓶で水をチャージし，セルに配水して自分のエリアの拡大を目指す．なお水瓶の水はいくらチャージしても減ることはない．
- セルの地形には以下の5種類がある
  - **空白**:移動・配水・チャージ不可能（競技進行には無関係なセル）
  - **平地**:移動可能、配水可能、チャージ不可能
  - **壁**:移動不可能、配水不可能、チャージ不可能
  - **水瓶**:移動可能、配水不可能、チャージ可能
  - **荒地**:移動可能、配水可能、チャージ不可能
- 定めるルールに沿って、最終的に得点エリアの多いチームが勝者となる．
- エリアは囲碁のように，自分で配水したセルを繋げ周辺を閉曲線状に囲む事によって，より広大なエリアを一度に得点エリアとなる．なお壁セル、空白セル、水瓶セルは配水できないので、閉曲線を構成する事はできない．
- 囲まれたエリアの中に別のチームが入れ子になっている場合は内側のエリアは内側のチームのエリアとしてカウントする．
- ロボットに対する操作命令は擬似的にリアルタイムで要求することができる．実際には各チームからの命令はキューに貯められ，セッション（3秒間隔）でまとめて実行され盤面状況が更新される．
- システムは3秒(180フレーム)を1つの単位（セッション）として処理する．
- 命令は各ロボットに対し送ることができる．
- 1つの操作命令には所定のフレーム数（時間）を必要とし，命令実行後に待ち時間が発生する．待ち時間が終了した後に次の動作に移る．
- 操作命令には以下のような種類がある．
  - 移動（40フレーム）
  - 配水（19フレーム）
  - チャージ（24フレーム）

●最終的な得点エリア

- 得点エリアは以下のようにカウントされる．
  - 自分チームが配水しているセル
  - 自分チームが配水しているセルの閉曲線で囲まれた、未配水のセル
- この二つの合計となる．
- 閉曲線内にある他チームの配水エリアはそのまま他チームの配水エリアとして計算さる．

●勝敗判定は以下の優先順位

1.自チームの得点エリアが多いチーム

2.フィールド内の自チーム総配水量が多いチーム

3.各チームのロボットの総蓄積水量が少ないチーム

4.この優先順位で計算され、それでも判定が決まらない場合はじゃんけん

●その他のルール等

- 各試合では全チームが同じフィールド内で試合を行い、勝敗を競う．
- 競技に持ち込んで利用できるコンピュータ類は携帯可能なものを2台以内．そのうち1台は100BASE-TXが使用できるEthernetポートとUSBポートを持つ必要がある．
- 競技用ネットワークに接続できるコンピュータは各チーム同時に1台のみ．
- 2台のコンピュータ間で情報をやりとりする事は禁止．

- テーブルには各チームに合計 150W 程度の電源コンセント 2 口を用意する．
- 競技中はチーム内での情報のやり取りは構いませんがチーム以外と情報交換することは認めない．
- 無線 LAN の利用は認めない．
- ロボットの動作以外でサーバや他チームの試合進行を妨害する行為は認めない．

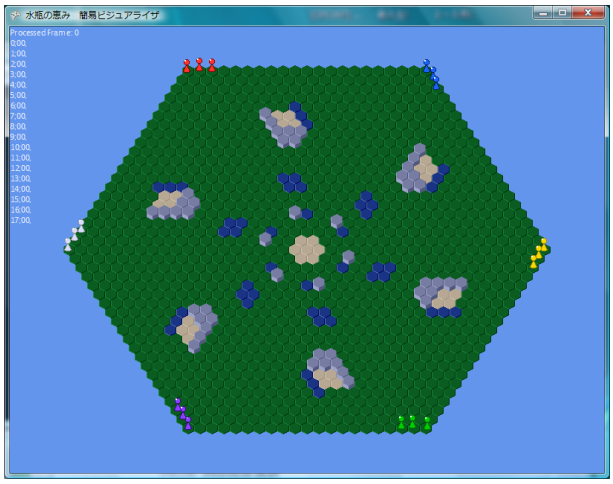


図 1　競技フィールド（初期状態の例）

3．スタンドアローン版の公開

競技のルールがあまり単純ではないので，それを理解してもらうため，ネットワークの接続を必要としないスタンドアローン動作を行うプログラムを公開した．プログラムの公開日程は以下のようになっている．

2010年2月1日　　競技概要の発表（ルール等の公開）
2010年4月29日　　スタンドアローン版公開
2010年5月14日　　質問受け付け終了
2010年5月28日　　募集終了
2010年6月26日　　予選通過チーム発表

スタンドアローン版と本選版とはシステムの動作が異なっている．スタンドアローン版の特徴は以下の通りである．

- サーバへの接続が不要で，パソコン1台で動作可能である．
- 1台で問題の読み込み，競技開始，集計をコントールする．
- 本選のシステムはサーバとの通信にxmlデータを扱うが，xmlのパーサが開発途中だったため，データはxmlではなく単純なテキストデータとする．
- 自チームはチーム 1 を制御し，相手チームの操作は付属する簡単な AI が行う．

実際のスタンドアローン版の起動状況を以下に示す。図 2 が起動直後の初期状態、図 3 が試合データを読み込んだ状態である。本来試合に関する情報はサーバから提供されるが、スタンドアローン版ではそれら全てがフォルダ io の中にあるファイルによって操作される。たとえば試合に関する情報は「gameInformation.txt」に対戦チーム・フィールド情報、そしてロボットの初期位置などが記載されている。またサーバに転送すべきデータもこの io フォルダ経由でやり取りされる．本選ではこの部分がネットワーク経由の xml ファイルになる．

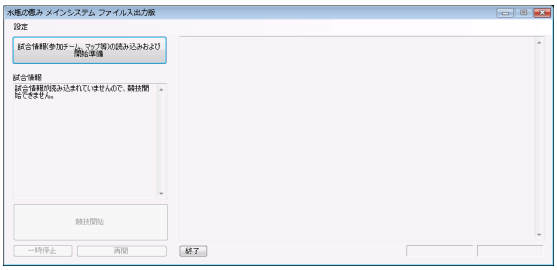


図 2　ゲーム作成画面

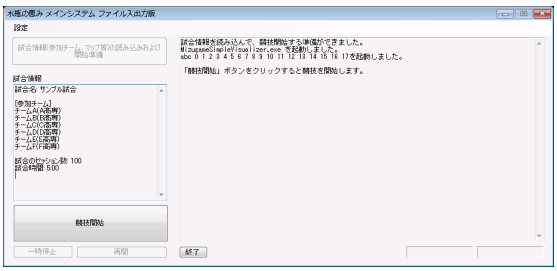


図 3　初期設定終了画面

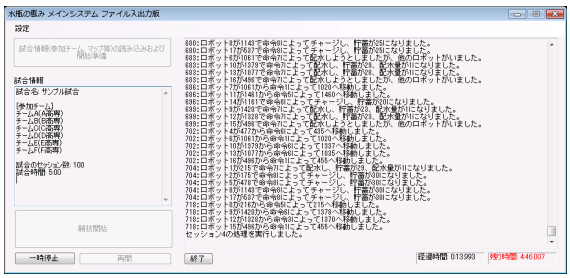


図 4　試合の進行状況

競技が開始されると，各チームからロボットへの指令が出され，その命令によってロボットが動く．しかしスタンドアローン版では，当然敵チームはおらず，自チームだけの動作しか指定できない．それでは練習としては多少不十分なため，敵チームを擬似的に操作するAIを搭載した．2コート～6コートは基本的にすべてAIによって操作される．図5にそのAIが出す命令が表示されている．当然このプログラムを自作することも可能であり，結果をioフォルダに書き込めば，各チーム独自の試合を実行できる．

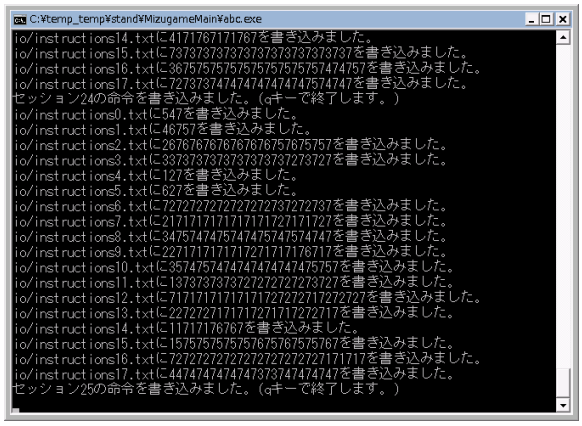


図５　AIによる自動操作の状況

競技が開始されると，各チームからロボットへの指令が出されロボットが移動する．各チームの動作を可視化するのが図6に示す簡易ビジュアライザである．

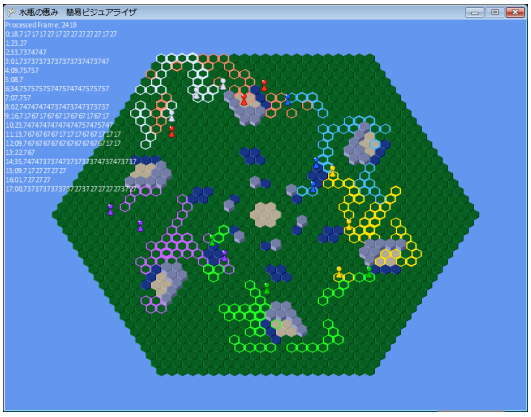


図６　競技ビジュアライザによる画面

４．インターネット練習システムの開発

スタンドアローン版でも各チームはシステム開発や動作チェックを行う事ができるが，やはりサーバとの接続がなし状態は本当のシステムとは言えない．特に本システムは1セッション3秒（180フレーム）とう動作の部分が非常にシビアであり，サーバと正常に接続できないと命令を迅速に送る事ができない．また，サーバとの接続はロングポーリングと呼ばれる技術を使って，各競技者とサーバ間の同期をとっており，この部分に関してもファイルベースのスタンドアローン版ではテストできない．

そこで，インターネット上に実際にサーバを置き，各参加チームがサーバに接続することで練習試合が可能なシステムを構築することとした．システム公開の日程は以下の通りである．

2010年8月31日　　ネットワークサーバの公開
2010年9月3日　　各チームのID発行
2010年9月9日　　ソースコード公開
（Java,C#,C++(Win32,Linux)）

以前インターネットとSSL-VPNを使った競技システムを作成した[5,6]．これはシステム管理者の管理の下，参加チームだけ（8チーム）がサーバに接続し，管理者の合図で8チームが同時に試合を行った．これを実現するためには，試合の開始時刻や接続パスワードの配布など，日程調整が非常に困難であった．

今回の練習システムではこのようなスケジュールを組む事は不可能なので，各校は独自に試合を作成し，独自に試合が進行できる必要がある．そこでゲームの作成から試合進行を各校単独で行う事ができ，しかも他のチームの影響を全く受けないように設計する必要がある．さらに参加60チームが同時に試合を行う可能性もあるのでサーバのパフォーマンスとレスポンスが非常に重要となる．

各チームが独自に試合を作成できるように，各チームに以下の情報を配布した．

- **コンテストID**：3桁の整数で各校に配布
- **コンテストパスワード**：MD5ハッシュ値
- **チームID**:各チームごとに配布（練習用6チーム含む）
- **チームトークン**：MD5ハッシュ値

これらにより，各チームは他のチームから影響を受けることなく，試合の作成から進行を独自に行う事が可能となった．またチームトークンは本番でも使用し，各チームによる不正や誤った送信を防ぐ機能も果たす．

サーバの詳細についてはコンテスト前であるため詳細は割愛する．

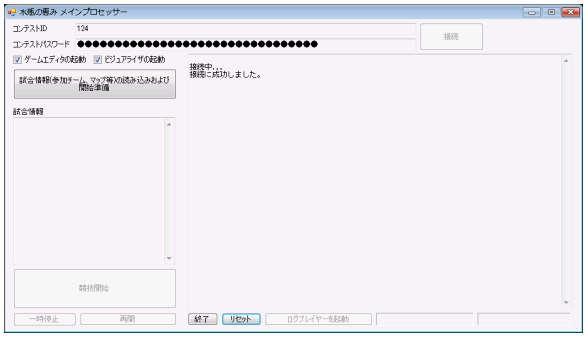


図７　コンテストIDとパスワードによるサーバ接続

コンテストIDとコンテストパスワードを入力するとサーバと接続することが可能になる．なお，本システムはプロキシ経由の接続も可能であり，多くの接続環境にも対応している．ただし，各チームが独自作成するプログラムには標準ではプロキシ超えの仕組みが含まれないと思われるので，xmlの解析とプロキシ接続方法を説明した（実際に使える）ソースコードを公開することで，この部分の敷居を下げた．

サーバと接続するとゲームを作成するゲームエディタが起動する．これは，ゲームのセッション数，参加チーム，そしてフィールドの作成を行う事ができる．フィールドは自動的に6チーム対照になるようにしか作成できないようになっている．チーム情報は最新情報をサーバか

らダウンロードし，ローカルに保存する．これらの情報はローカルにテキストファイルで保存する事ができるので，容易に同じ試合を開催できる．

図７　ゲームエディタ

ゲームの作成が終了すると「試合を作成（サーバに送信）」ボタンを押す事で，必要な情報がサーバに転送され，サーバ側で試合を開始する準備が行われる．

「試合開始」のボタンを押すことで，10秒のカウントダウンの後に試合が開始される．以下の場合はチーム1を自チーム，残り5チームをAIとして設定した．各チームは自チームを動かすプログラムを作成し，サーバに転送刷ることで本選と全く同じ条件で試合を行う事ができる．図8に試合開始直前の画面を示す．

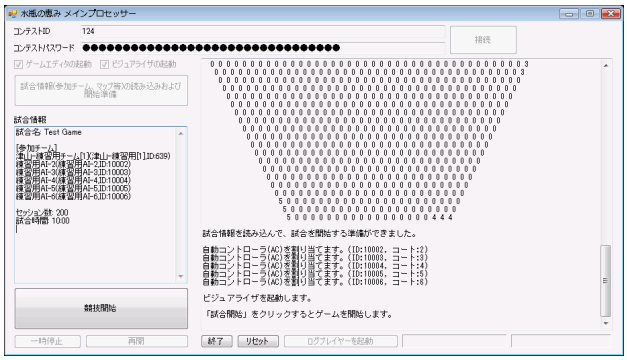


図８　ゲームの初期設定状態

実際に競技が開始されると，図9のように2～6チームのロボットに対してAIから操作命令が送信され動作する．図10にビジュアライザによる試合の進行状況が表示される．このビジュアライザに表示されているのと全く同じ情報が，各チームには配布される．配布は命令コード転送の戻り値として，各チームに配布される．このタイミングが全チーム共通のため，3秒ごとの進行を各チーム公平に行う事ができる．

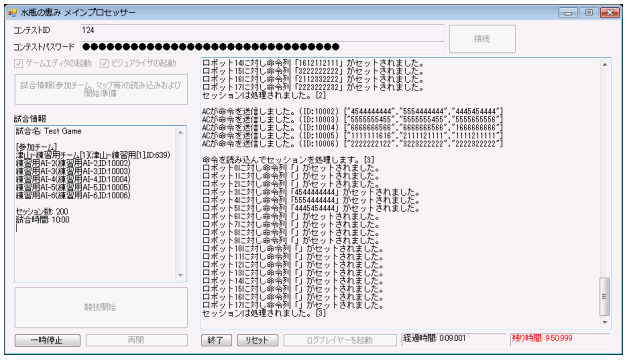


図９　ゲームの進行状態

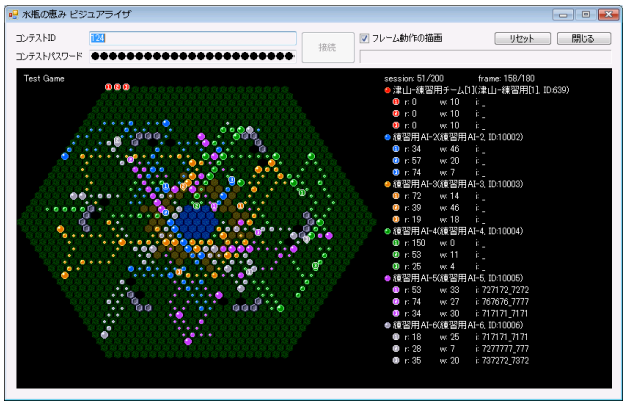


図１０　ビジュアライザによる競技進行状況

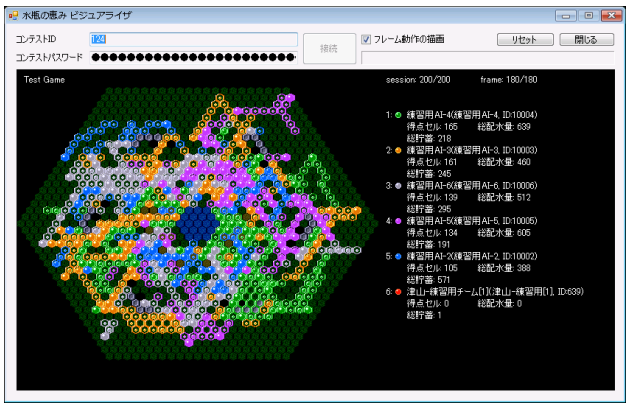


図１１　ゲームの終了状態

本来，各ロボットの操作は各チームが開発したプログラムによって制御を行うものであるが，動作をチェックするという目的から，手動でロボットを操作するコントローラを準備した．図12にコントローラによりチーム1のロボット1～3を操作しているようすを示す．

競技が進行し，全て終了すると各チームが取得したエリアを計算し，図11のように順位が表示される．

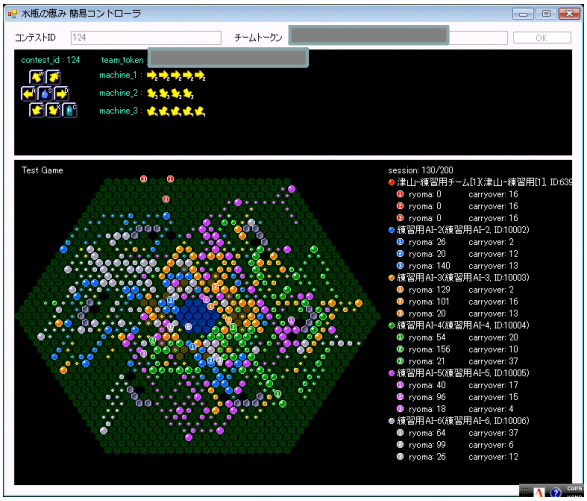


図１２　コントローラによる手動制御

５．まとめ

第21回プロコンでは，非常に早い段階からプログラムの開発を行い，参加者が参加するために必要な情報を可能な限り早く公開する方針をとった．これにより，競技者の積極的な参加を促すと同時に，システムの不具合についても同時にチェックする意図がある．

システムの開発に当たって，セキュリティ対策にも十分配慮した．しかし，以前作成したSSL-VPNのような非常に厳密な体制は不可能であり，新たにシステムを開発する必要があった．

これから10月にプロコンの本選があり，これらのシステムが本選でも正常に動作することを目指して，更に改良を重ねている．またエンターテインメント性を高めるため，3次元のビジュアライザも開発中であり，高知では活躍できると考えている．

今回の経験を糧に今後の高専プロコンの発展に寄与できれば幸いである．

参考文献

1)プログラミングコンテスト公式ホームページ http://www.procon.gr.jp/
2)U-20 プログラミングコンテスト，http://www.jipdec.or.jp/archives/project/procon/index.html
3)パソコン甲子園公式ホームページ，http://web-ext.u-aizu.ac.jp/pc-concours/
4)ACM大学対抗プロコン（日本支部），http://www.acm-japan.org/Welcome-j.html
5)寺元貴幸,宮下卓也,最上勲,岡田正,井上恭輔,松野良信,高専教育32, pp. 921-926(2009)
6)飯田忠夫，田中永美，長岡健一，山田洋士，金寺登:第13回プログラミングコンテスト競技部門運用支援システムの構築について，高専教育，Vol.27 pp. 721-726(2004)

可視化タグを使った大規模可視化システムの提案

A PRP`PSAL OF LARGE SCALE VISUALIZATION SYSTEM BY USING VISUALIZATION TAGS

宮地英生[1)]
Hideo Miyachi
1)工博　サイバネットシステム株式会社(〒101-0022　東京都千代田区神田練塀町 3)

This document describes a proposal of "Visualization Tag". In the field of medical imaging, the standard image format, DICOM, was defined in 1993 and it has contributed to use the images effectively. In the field of scientific visualization, because the standard format has defined, the visualized images were never reused. Then, we propose a "Visualization Tag". Using it, the images can be arranged according to the parameter, and it is useful for the comparison and the analysis of the visualized images.
***Key Words**: Tiled-displays, visualization, PNG*

1. はじめに

世界最高速のスーパーコンピュータはペタフロップスを超えてきた。計算速度の向上につれ、計算量は時間・空間方向に増え対話的な可視化が困難になるという問題が指摘されている。その中でも解像度の不足については意識する人も少なく、あまり解決方法が議論されていない。その中で、近年タイルディスプレイが普及してきた。ハードウエアはサムソン、シャープがベゼル幅の小さなディスプレイとPCがセットされたシステムを開発、デジタルサイネージで活用されている。

タイルディスプレイの科学技術分野での利用方法は、先に述べた高解像度の表示、および、たくさんの結果の比較が考えられる。このとき、可視化画像に標準のメタ情報がないので、レイアウトは手動か、システム毎に考えなければならないという問題がある。

そこで我々は、可視化タグを定義して、その情報を使って効率的にレイアウトし、可視化結果を検討するシステムを提案する。本稿では、そのプロトタイプの開発と、いくつかの事例を紹介する。

2. タイルディスプレイ

タイルディスプレイへの画像表示を支援するソフトウエアはDMXなど多数あるが、われわれは米国EVLで開発されたSAGE[1]をベースとしたシステムxK[2]システムを開発している（図１）。

xKシステムに含まれるxKUIは、SAGEのユーザインターフェイスを拡張したもので、複数のユーザが各自のPCを持ち寄り、タイルディスプレイを前に対話的にコンテンツを操作しながらディスカッションするときの支援を目的としている。 x KUIでは、参加者が参加者全員の画像を自由にレイアウトできる。しかし、対話的にレイアウトするだけでなく、それらを日時順や作者別に整列させたい場合もある。

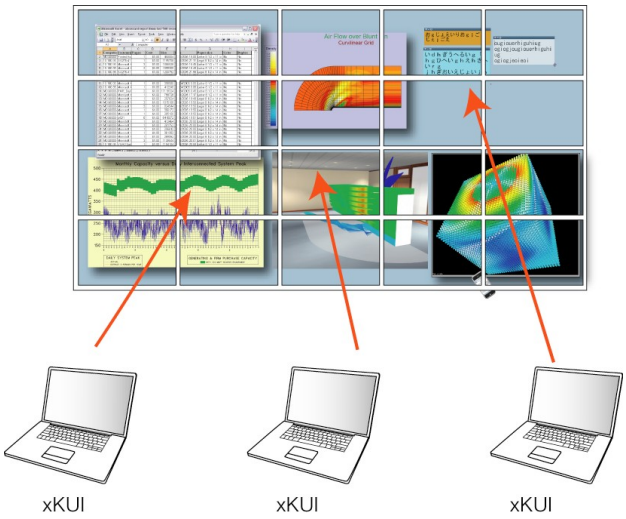


図１　ｘKユーザインターフェイスの活用法

数値計算と可視化に特化すれば、同じ計算条件の結果を集める、速度の速い順に並べる、時刻暦で並べることが考えられる。そのためには、画像ファイルにファイル名や作成日時以外のメタ情報が必要となる。

図２は可視化ソフトウエアAVS/Expressを用いてタイルディスプレイに、時系列の一覧表示、3面図表示、そして1枚の高解像度表示を行った例である。これは１つのアプリケーションから全ての表示をコントロールしているので、マルチ表示をしながら、一斉に回転・拡大などの操作やパラメータの変更ができる。そのような利点がある反面"ライブ"で可視化をするには、すべての計算結果にアクセス可能な状態で、それをフルに可視化するだけの能力がマシンに必要となる。さらに、多様なレイアウト方法に対して全て対応できるように可視化アプリケーション自体をカス

タマイズしなければならないという大きな問題がある。

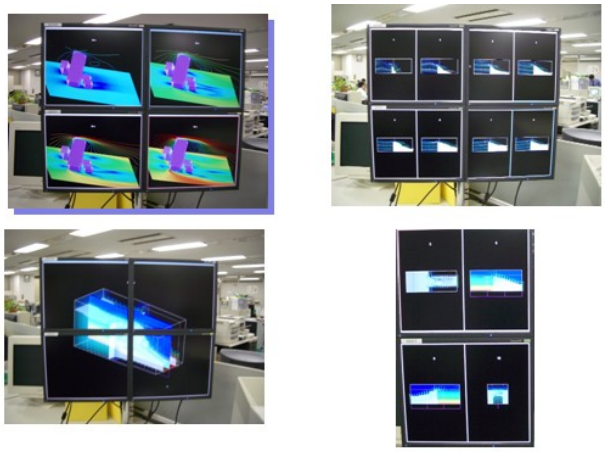


図２　タイルディスプレイでの数値計算結果の表示例

それに対して、画像を並べる方法では、対話的に新しい画像を生み出すことはできないが、いくら大規模な計算結果であっても小さな計算機で画像表示が可能で、データベースと連携することで過去の可視化結果との比較ができる利点がある。また、画像とタグ情報という抽象化された概念に統一することができるので、実験結果と計算結果の比較、図１のようにグラフやExcelの情報など、多様な情報を統一的に扱うことが可能となる。

３．画像とタグ情報

画像が付加的な情報を持つのは一般的である。

例えば、デジタルカメラの画像は1995年10月に(社)日本電子工業振興協会（JEIDA）の企画としてExif (Exchangeable Image File Format)1.0が制定され、それをベースにデジタルカメラの統一ファイル運用規定(DCF(Design rule for Camera File system)が1998年12月に制定された。そこではTag構造により、撮影日時、メーカー名、露出時間、フラッシュなど、撮影に関する付帯情報のフォーマットが規定され、撮影時に記録される[3]。バージョン2ではGPS情報により撮影位置も記録されるので、土木工事の施工書類にも利用される。

「医療におけるデジタル画像の通信」規格として1993年に公表されたDICOM (Digital Imaging and Communications in Medicine)は患者情報、撮影日時、撮影機器など、医療に必要な多くの情報を含む[4]。

このように画像を実務に応用するためには、ピクセル数や色数、圧縮方式など、表示のために必要な情報だけでなく、その画像に関る付帯情報が重要である。論文の執筆時に、可視化結果の画像を挿入しようとして、本当に該当の画像かどうかを迷うことがあるだろう。そのとき、画像の正当性に確信が持てない場合、多くの人は、もう一度、計算結果から可視化するに違いない。これは研究業務にとって無駄になるし、一度、取り違えてしまった画像が、著者の思い違いで利用され、参照され続ける可能性もある。したがって、画像に付帯情報を付けることはタイルディスプレイの活用だけでなく、数値計算の可視化全体に利益をもたらすと考えている。

４．プロトタイプシステム

プロトタイプの開発ではPNG(Portable Network Graphics)形式[5]の画像ファイルを対象とした。PNGはタグ形式で、付帯情報の追加や削除に関して、該当部分だけを編集すればよく取り扱いが簡単であること、付帯情報を画像情報より後ろに記述することができるので、前から解釈していくソフトウエアには影響を与えず、また、大きな付帯情報をつけても表示速度が遅くならないことから、プロトタイプの対象として選択した。

今回、下記の２つのツール（図３）を開発し、xKシステムにタグ情報を読み取り、それにしたがって配置する機能を追加した。

PNGviztag

PNGの中からtEXTタグを取り出して標準出力に表示する。

PNGViewer

引数で与えたファイルの内容をtEXTタグとして追加または削除する。

これらの２つのツールを使うことでPNGの画像ファイルに付帯情報を追加し、それを確認することができる。

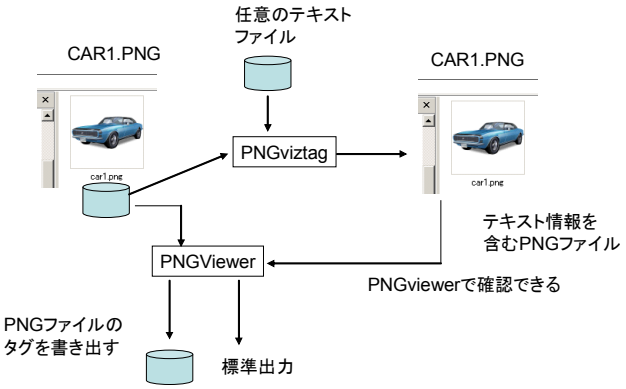


図３　PNGにタグを追加、確認するツール

PNGはWindowsのファイルエクスプローラで解釈され、表示を「縮小版」にしたとき、アイコンとして表示される。適正にtEXTタグが追加された場合、そのような表示には何の影響も与えない。

可視化タグは <Viz> </Viz>で囲むこととし、その中に記述するタグの詳細は決めていないが、例えば、図４のようなタグが考えられる。

図４のような記述からC:/2010/Cavity/re100.fldというファイルを読み取り、等値面のP=1.0の表示を、ある角度から見た画像がレンダリングされたことがわかる。しかし、これだけでは、画像の元となったデータファイルの名前しか解らないので、可視化パラメータはわかっても、その画像

の正当性は保証できない。ソルバプログラムの出力にも計算に関る情報を含め、それを可視化タグに継承するようにできれば、画像の正当性が保証可能となるだろう。

```
<Viz>
 < source> C:/2010/Cavity/re100.fld</source>
  <method> iso-surface  </method>
  <parameter> p=1.0 </parameter>
  <View>
        <look>
              <at> 0,0,0 </at>
              <from> -12,0,0 </from>
              <up>   0,1,0 </up>
        </look>
  </View>
</Viz＞
```

図４　Visualizationタグの例（想定）

５．レイアウト事例

2次元の円柱周りの流体解析の結果を使って、いくつかのレイアウトを試した。

（1）　時刻方向の変化の一覧表示（図5）

円柱周りの流れは周期解となるが、計算前にその周期は解らない。そこで通常、適当な間隔でファイル出力を行う。Windowsの「縮小版」表示でもサムネイル表示は可能だが、それはファイル名、日時などの順序でしか整列できない。Windowsでは、ファイルの属性としてタグを追加する方法も試したが、画像ファイルの本体に記録されていないのでFTPやメールへの添付で属性がなくなってしまうことがある。可視化タグを使うことで情報は消えることなく、付帯したままで、ｘK側の実装によって、時刻100-300のような区間表示や3個毎などのピックアップが可能となる。

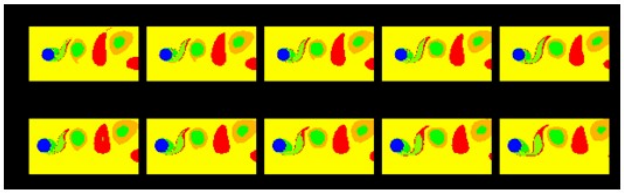


図５　時刻方向変化の一覧

（2）ケース比較の一覧表示（図６）

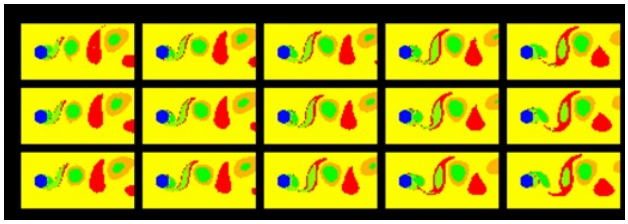


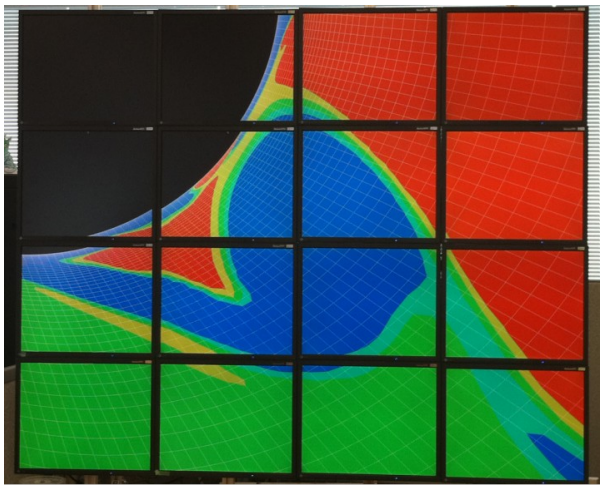
図６　条件（レイノルズ数）の異なる流れの比較

次にレイノルズ数を変更した３つの結果を比較するためにレイアウトした。

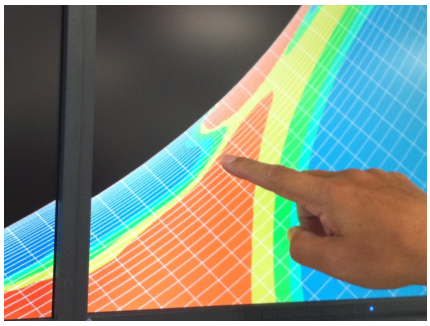
ここでは上からRe=3000，5000，10000、横には同じタイムステップでの結果を並べている。ここでは周期解が出るまでRe=10000で計算をし、それを初期条件として、Re=3000,5000,10000の計算を行った。図４は、Reが変更された直後の結果である。1つずつシリーズを見ていると、ほとんど差を感じることができないが、このように並べてみると少しずつ後流の渦の様子が異なっていることが解る。

（3）高解像度画像の表示（図７）

比較ではなく、1枚の高解像度表示のためにレイアウトした事例を図５に示す。これは、先の2次元の円柱周りの計算結果の渦度を表示した様子である。格子数は200x100と、それほど大規模な計算ではないが、円柱表面に格子が集中しているので、図５や図６のように後流を見ているときには最小メッシュを確認することはできない。一方、最小メッシュを確認するためには図７(b)の拡大図のように表示する必要があるが、ここまで拡大してしまうと全体との関係が全く解らなくなってしまう。



(a)全体図



(b) 拡大図

図７　高解像度表示

このように、20,000メッシュ程度の計算結果においても、粗密のある格子においては、全体と詳細を同時に把握することが困難である。現在、大規模計算といわれる数十億のメッシュでは、3次元の断面を取り出した場合でも1面に100万点以上入ることになるので、通常のディスプレイでは解像度は不足する。

また、このようにズームアップの状態で10x10のようなタイル状の画像をバッチ処理で生成しておき、ｘKのユー

ザインターフェイスから、X=1-5,Y=-2-+2 のように範囲を選択して表示することもできる。少し時間がかかるが、限られたリソースのマシン上で、超大規模な計算結果に対して対話的な可視化を行うことができる。

５．おわりに

画像に可視化タグとして付加情報を挿入し、それを使ってタイルディスプレイにレイアウトするプロトタイプシステムを開発した。このように可視化結果の画像に計算条件や可視化パラメータを挿入することで、過去の結果との比較、時系列データの検討、高解像度の表示が行えることを示した。現在は、独自のツールで付加情報をつけているが、可視化ソフトウエアが自動的に情報を付加することが必要である。さらに、本当に作業を効率的にするには、ソルバーも含めた標準化が必要になる。

謝辞：本研究は独立行政法人情報通信研究機構の委託研究の一部として行われた。

参考文献

[1] L. Renambot et al.: SAGE: the Scalable Adaptive Graphics Environment, Workshop on Advanced Collaborative Environments (WACE 2004), 2004.

[2] 宮地英生、久木元伸如：タイル型ディスプレイへの画像表示インターフェイスの開発, 第11回PSEワークショップ（静岡）、2008

[3] （社）日本電子工業振興協会：JEIDA 規格　ディジタルスチルカメラ用画像ファイルフォーマット規格(Exif) Version2.1 日本語版 JEIDA-49-1998, 平成 10 年 12 月改正版、1998.12

[4] Herman Oosterwijk ： DICOM 入門、Herman Oosterwijk、（（社）日本画像医療システム工業会監訳）、 篠原出版新社、2008

[5] http://www.libpng.org/pub/png/ ： Portable Network Graphics ホームページ

単純なWebベースエージェントの振る舞い

BEHAVIOR OF SOME SIMPLE WEB BASED AGENTS

早勢欣和
Yoshikazu Hayase
富山高等専門学校 電子情報工学科 (〒933-0293 富山県射水市海老江練合 1-2, hayase@nc-toyama.ac.jp)

This paper describes about PSE (Problem Solving Environment) that is constructed using cooperation of some Web based agents which process on one or more servers for users to concentrate on a problem solving. A behavior of one simple Web based agent is predictable about, because a simple Web based agent carries out only simple processing respectively. However, behaviors by cooperation of two simple Web based agents become less simple. Furthermore, behaviors by cooperation of some simple Web based agents are complicated. This paper considers behavior of some agents seen when some simple problems were solved.

Key Words *: web-based PSE, distributed processing, agent*

はじめに

アプリケーション間の連携処理を行うことで問題を解決するといった環境は，ネットワークに接続されている複数のWebサーバを用いることで比較的安易に構築することができる．しかし，Webサーバの多くは，単一の処理のみではなく並行して様々な処理が行われているのでCPUやNICの負荷状況は常に変化している．このため，Webサーバの処理応答時間にばらつきが生じてしまう．また，インターネットの通信品質は必ずしも安定していないのが現状であり，たとえLAN環境であっても，時間帯によって接続ノード数が変化するなど，トラフィックが変動する要因は多い[1, 2]．このようにWebサーバによる連携システムは通信の不確定性を内包するので，高精度や高速度が要求される数値計算などで利用するには技術的な対策を行うことが必要だと考えられる．

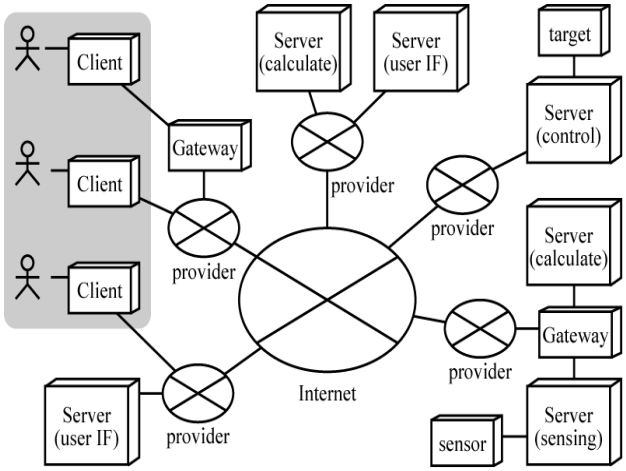


図 1　インターネットにおける連携処理環境例

ところが，この環境に対して技術的な改善をすることなくそのままに，単純な処理のみを行うWebベースエージェントを配置してみたところ，システム環境の不確定性から明白であるとも考えられるが，各エージェントの振舞いが複雑になるといった現象を見出すことができたこれまで，この複雑な振舞いを示すことをシステムの特徴とする分散Webサーバ連携によるPSEを提案し，複数のエージェントを静的に配置することによる問題解決の事例として4彩色問題への適用について報告してきた[3, 4]．今回は，複数のエージェントによる連携において，データを保存したままエージェント間を移動する事例における振舞いについても検討を行う．

単純なWebベースエージェント連携PSE環境

ネットワーク構成

分散Webサーバを用いた複数の単純なエージェントによる連携処理を試行するために，図2のように学内LAN環境に接続されたサーバによるシステムを構築した．

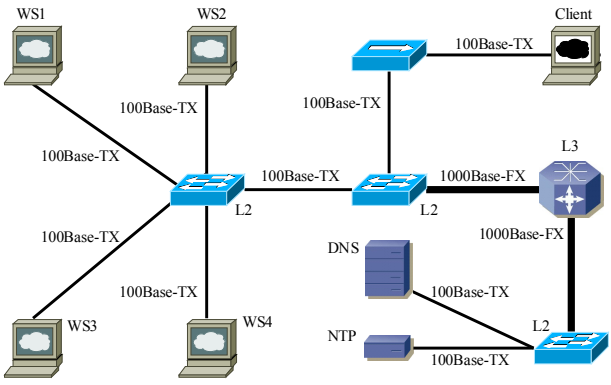


図 2　分散 Web サーバ連携システム構成概念図

表 1　分散 Web サーバ連携システムスペック

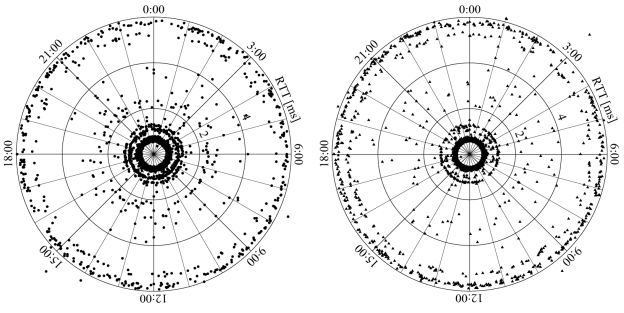
	OS	Web	PHP	CPU	Memory	NIC
WS 1	Ubuntu 10.04.1 LTS	Apache 2.2.14	5.3.2-1	AMD Athron™ 64 3500+ 2.2GHz	8GB	1000Base-TX
WS 2	Ubuntu 10.04.1 LTS	Apache 2.2.14	5.3.2-1	AMD Athron™ 64 3500+ 2.2GHz	8GB	1000Base-TX
WS 3	Ubuntu 10.04.1 LTS	Apache 2.2.14	5.3.2-1	AMD Athron™ 64 3500+ 2.2GHz	8GB	1000Base-TX
WS 4	Ubuntu 10.04.1 LTS	Apache 2.2.14	5.3.2-1	AMD Athron™ 64 3500+ 2.2GHz	2GB	1000Base-TX
DNS	CentOS 5.1.0	-	-	Intel® Xeon® 3065 2.33GHz	2GB	1000Base-TX

NTP	SEIKO TS-2540 Time Server	-	-	100Base-TX
L3	Cisco Catalyst 6500	-	-	1000Base-FX
L2	Cisco Catalyst 2960, 2950	-	-	1000Base-FX 100Base-TX

各サーバ等のスペックを表1に示す．WS1, W2, W3, W4は同一スペックのものを同時に導入したサーバである．ただし，その後，OSやメモリ構成を変更している．学内LANの基幹はギガビットで構成され，学内LANには多数のコンピュータ・ネットワーク機器が接続されている．

サーバ間応答距離の不確定性

ここでは連携処理で用いるWebサーバ間の応答距離をTCP httpポートへpingをかけた際の応答時間(RTT: Round Trip Time)の値で表す．同一スペックでネットワーク接続も同等であるWS1とWS2の間の双方からのサーバ間応答距離を5日間測定した際の結果を図3に示す．1日を1周24時間で表し，各時刻でのRTTの値をプロットした．中心が0.0[ms]であり，外周に向かうほどサーバ間距離が遠い状態となる．この測定期間は，各サーバではApache等のいくつかのプロセスは起動したままとしたが，ping以外のアクセスは意図的には行っていない．理想的なコンピュータ・ネットワーク環境であればスペックに応じ一筋の円が双方に同じ結果として現れると考えられる．実際には分散Webサーバ間の応答距離は一定ではなくかなりのバラツキが生じており，ハードウェアによっても個体差が生じることになる．ときには極端に応答が悪くなることもあり，ネットワーク環境によってはタイムアウトとなってしまうこともある[3]．



from WS1 to W2　　from WS2 to W1

図 3　サーバ間応答距離 6[ms]以下(WS1 – WS2)

不確定性を示す分散Webサーバ連携PSE

分散 Web サーバへの単純なエージェント配置

単純な戦略に基づいて値を決定し-1または1のいずれかの値を保持するといったエージェントについて考える．同一の戦略を用いる2つのエージェントを用意し，他のエージェントと同じ値となる(non zero sum)，あるいは異なる値となる(zero sum)ことを目的とした場合の振舞いについて検討する．任意の時刻における各エージェントの値を $A_0(t), A_1(t)$ で表すとすると，zero sumでの戦略は $A_0(t') = -A_1(t), A_1(t') = -A_0(t)$ となるが，各エージェントの初期値が $A_0(t_0)=A_1(t_0)$ と等しいとき，計算をいくら繰り返しても目的を達成することができない．しかし，このエージェントを2台のWebサーバにそれぞれ1つずつ配置した場合は，サーバ間応答距離のわずかな変化が処理の順番に影響することで目的を達成することができる．

応用としての４彩色問題解決への適用

4彩色問題は，マップの各領域を隣接する箇所で色が重ならないよう塗り分ける問題である．この解法として，例えば分枝限定解法やニューラルネットワークを用いたものが提案されている[4, 5]が，単純なWebベースエージェント連携による問題解決環境を用いることでも解を得ることができる．

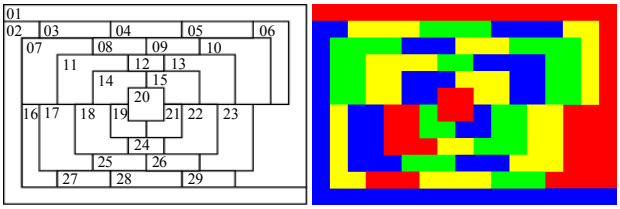


図 4　４彩色問題のマップとその解の例

各分散Webサーバ(WS1,WS2,WS3,WS4)に図5のように振舞うエージェントプログラム(PHPで記述)を配置する．各エージェントはそれぞれ独立して機能し，非同期に連携して処理を行う．隣接領域で全色が使用されている場合は色の選択候補が無いことになるが，例えば色表をもとに現在の色の次のものとするなどの戦略を組み込む．

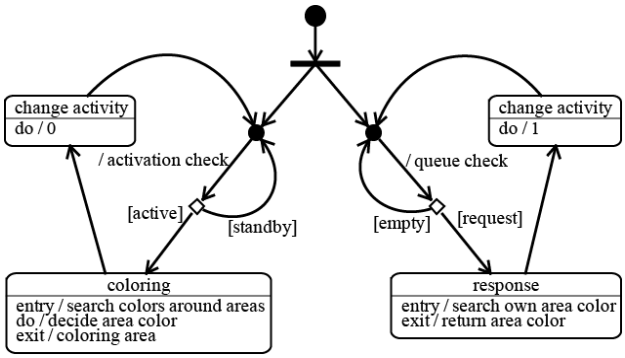


図 5　４彩色問題のために用いたエージェント仕様

Webベースエージェントの振舞い
データ非保存問題における動作事例

複数の単純なWebベースエージェントの連携処理における振舞いを検討するために，1,024個のエージェントを用いた動作試験を行った．領域を32×32の小領域に分割し，それぞれにエージェントを配置する．各エージェントは隣接する小領域の色情報を集め，最も多くの領域を占めている色に自身の値を変更するといった単純な処理を行う．なおドメインにおけるデータの保存は行わない．なお今回は常に活性化した状態とした．

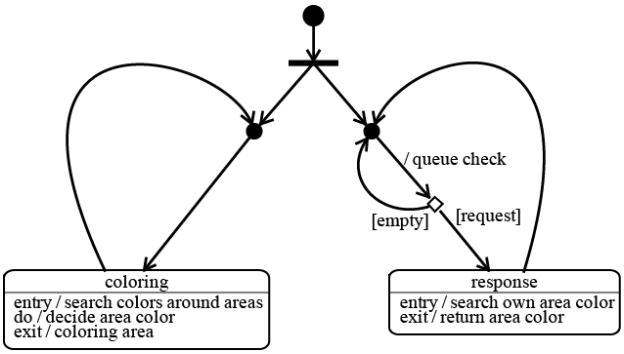


図 6　振舞い検証試験のためのエージェント仕様

各小領域の初期値は乱数を用いて無作為に設定した(図7左上).

$$A_i(0).color = rand()\%4$$

この初期状態から1,024個の各エージェントを活動させ収束した状態の結果を確認した(図7).

$$A_i(step+1).color = \max(colorA\,round(A_i(step)))$$

エージェントが自分の周囲の色情報を収集し終えるまでの時間は一定ではなく，また同時刻の情報ではない．また隣接している小領域の色が情報を収集した直後に別の色に変化してしまっている可能性は大きい．

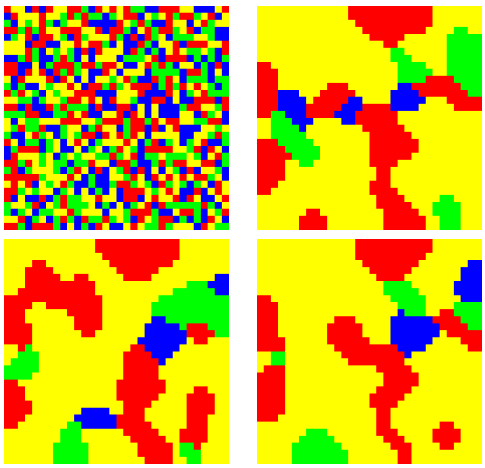


図 7　振舞い検証試験結果の一例

次に．Webベースエージェントを配置し，非同期の連携処理を行った場合の振舞いについて，4彩色問題に適用した際の事例を確認する．この問題でもエージェントを領域ごとに配置したが，初期値は全て同じ色からはじめ隣接する領域にあるものとの連携処理を行わせた．このときのエージェントデータ構造を図8に示す．色情報に加え隣接領域のエージェント名，色決定処理で選択できるものがない場合に用いる戦略などの情報が含まれる．

```
<?xml version="1.0" encoding="UTF-8"?>
<four-color>
 <agent>
  <name>01</name>
  <server>hadjet.toyama-cmt.ac.jp</server>
  <color>0</color>
  <link>02</link>
  <link>03</link>
  <link>04</link>
  <link>05</link>
  <link>06</link>
  <link>10</link>
  <link>23</link>
  <link>29</link>
  <strategy>($color + 1) % 4</strategy>
  <activity>1</activity>
 </agent>
</four-color>
```

図 8　エージェントデータ構造

各エージェントの<activation />の値が1のとき活性化していることを表す．このとき先ず担当領域の色を決定するための処理を行い，次に<activation />の値を0として停止する．なおエージェントの活性化は他のエージェントからの色情報収集のアクションで行われる．図4の例題で用いた29個のエージェントにおける<activation />状態の遷移を時刻順に並べてプロットしたものを図9に示す．分散Webシステムそのものに潜在する不確定性の影響もあるとは考えられるが，単純処理を繰返すだけの各エージェントだが，非同期で互いに連携することで活性化のパターンが画一化していないことが確認できる．

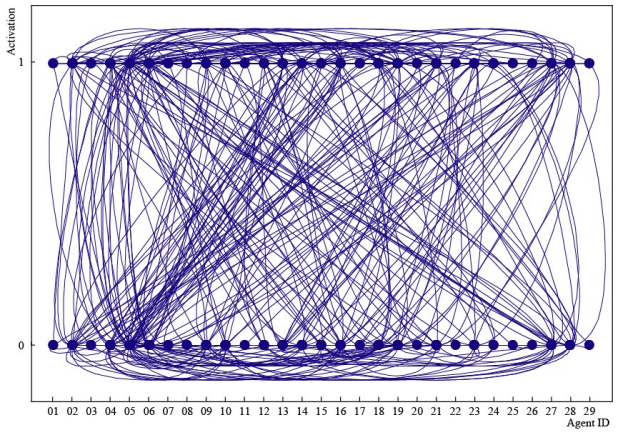


図 9　エージェントの activation 状態遷移

データ保存問題における動作事例

問題解決の対象とするドメインにおけるデータの値を保存しつつ連携を行う事例として，エージェントが単純なルールによりデータをエージェント間で受け渡してい

く際の振舞いについて検討した．なお，データをエージェントの主要部とすることで，エージェントが移動していくとみなすことができる．

- 領域を8×8の小領域に分割する
- 各小領域にエージェントを配置する
- ドメイン内のデータは1～8の8個とする
- データを保有するエージェントは活性化する
- エージェントは複数のデータを保有できない
- 縦横斜めの直線上のどこかに他のデータがあれば隣接する他のエージェントにデータを渡す

初期状態として対角線上の各エージェントにデータを割り当てて連携処理を行った際の，データの移動についてスタートから100回までの遷移図を図10に示す．

エージェントの戦略は以下のようにした．

- データが存在しない方向のエージェントに渡す
- データを渡せない場合は待機する
- 周囲のサーチおよびデータを渡すエージェントの選択は時計回りで行う

この例では，同じ戦略でデータの移動を行わせたことから，全体としてはデータが集団を形成していく結果となっている．8個のデータを保有する各エージェントが非同期で活動することによる影響から，データの遷移はそれぞれで異なるパターンを描いている．

なお，この例では左上にデータの輪ができるとデータの移動先がなくなり手詰まりとなる．

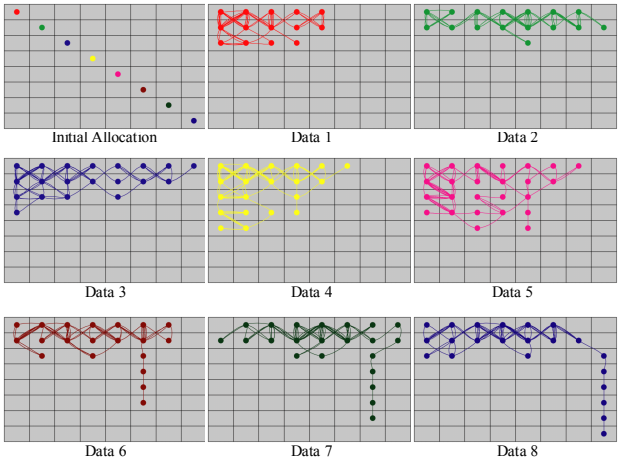


図 10　データごとの遷移の様子

おわりに

分散Webサーバ環境に，複数の単純なWebベースエージェントを配置して連携処理を行うように構築するだけといった問題解決環境を提案した．このシステムの特徴は各エージェント間の連携を非同期で行うこと，およびネットワーク環境に起因する不確定性を有していることである．

ひとつひとつのエージェントの振舞いは単純なもので決定論的であるとみなすことができる，しかし，こうした単純なエージェントの複数が互いに連携しあうことでその振舞いは複雑なものとなるといった結果が得られ，これにより，複雑な挙動を示すシステム処理系として構築することができていることを明らかにできた．

今後，ゲームやカオスなどとの関連性の検討を予定している．また，問題解決環境として利用できるドメインを明確にしていくことが課題である．

参考文献

1) 亀井陽一郎, 平沼賢次, 田伏正佳, 高橋伸弥, 河野通夫: インターネットを介した遠隔制御のための基礎実験(ネットワークサービス), 情報処理学会論文誌, Vol.45, No.3, pp.838-841, 2004

2) 汐月哲夫: インターネットの遅延特性と双方向遠隔通信制御, システム制御情報学会誌, No.45, Vol.12, pp.695-702, 2001

3) 早勢欣和: 不確定な連携挙動を示す分散サーバ環境の4彩色問題への応用, 第11回問題解決環境ワークショップ論文集, pp.69-72, 2008

4) 早勢欣和: 4彩色問題のための単純なWebベースモジュール連携によるPSE(WEB-BASED SIMPLE MODULES COOPERATIVE PSE FOR FOUR-COLORING MAP PROBLEM), , 計算工学会講演会論文集(Proceedings of the Conference on Computational Engineering and Science), 第15巻 第2号, pp. 1023-1026, 2010

5) 早勢欣和: 分散サーバ連携における不確定な挙動を考慮したユーザ支援に関する検討, 計算工学会講演会論文集(Proceedings of the Conference on Computational Engineering and Science), 第13巻 第2号, pp.979-982, 2008

6) 下田善隆, 田岡智志, 高藤大介, 渡邉敏正: グラフ彩色問題に対するPCクラスタ並列分枝限定解法の性能評価, 電子情報通信学会技術研究報告. COMP, コンピュテーション, Vol.104, No.642(20050121), pp.67-76, 2005

7) 山田祐司, 康敏: マキシマムニューロンニューラルネットによる4彩色問題アルゴリズム, 電子情報通信学会技術研究報告. NLP, 非線形問題, Vol.97, No.531(19980206), pp.59-66, 1998

異種グリッドへのジョブ投入を可能とするワークフローツールの実装と実験

IMPLEMENTATION AND EXPERIMENTS OF WORKFLOW TOOL SUBMITTING JOBS TO MULTPLE GRID INFRASTRUCTURES

田中義一，合田憲人
Yoshikazu Tanaka and Kento Aida
国立情報学研究所　リサーチグリッド研究開発センター

This report describes design and implementation of Workflow Tool that provides users with a single seamless interface to manage jobs on many infrastructures operated using different middleware. The experimental results of a computational fluid dynamics application consisting of mesh generation job, numerical fluid simulation job and visualization job show that this system enables users to run these jobs on computing resources operated with different middleware easily.

***Key Words** : Workflow, Grid, Middleware, JSDL, HPCBP,OpenFOAM*

1. はじめに

e-Science の実現のためには，分散された計算機やアプリケーションなどの資源を連携させる技術が必要である本報告では，ユーザの属する研究室が所有する PC クラスタなどの計算資源(LLS: Laboratory Level System)と情報基盤センタ等で運用されるスーパコンピュータなどの計算資源（NIS: National Infrastructure System）間の連携に関して述べる．グリッドはこれらの資源を連携するための重要な技術であるが，グリッド運用には高度な技術が必要とされるため，グリッドにより管理される資源は情報基盤センタなどで使われる高性能計算機に限られており，研究室に設置されるような計算機には及んでいない．従って，ユーザが研究室の計算機とグリッド上の計算機を連携させたい場合には，煩雑な処理が必要となっている．

著者らは，上記の問題を解決するため，LLS 資源にもインストールや管理が簡単なグリッドミドルウェアを導入し，異なったミドルウェアで管理される LLS/NIS 資源に対してシームレスにジョブ投入や連携ができるワークフローシステム(RENKEI-WFT)を検討した．

ところで，ジョブの連携のためのワークフローシステムとして， Unicore, GridSAM, Condor, GridAnt/Karajan, Taverna, Triana, NAREGI[1)-8)]などが知られている．しかし，これらのシステムは，Taverna, Triana を除き，１つのミドルウェアに対応するもので，複数の異なるミドルウェアに跨ってジョブを実行することができないという問題があった．また，ジョブを指定する記述方法が，それぞれ異なる言語を採用しているため，種々のミドルウェアをシームレスに利用することはできなかった．

異なるミドルウェアのもとで動く計算機を，ユーザが特に意識することなしに使えるためには

＊同一のジョブ記述インタフェース

＊同一のジョブ投入監視インタフェース

を備えていることが必要である．さまざまなミドルウェアに対するインタフェースの標準化に関しては，OGF[9)]で現在議論され，HPC Basic Profile(HPCBP)[10)]で定義されている．前者に関しては，そこで参照されている JSDL(Job Submission Specification Language)[11)]を，LLS/NIS 内の様々な計算資源に対するジョブ記述言語として採用した．WFT は，それぞれのミドルウェア用の記述言語のドキュメントに変換して，ジョブを投入する．

後者に関しては， BES(Basic Execution Service)[12)]をベースとした標準化が進められているが，実装されたミドルウェアはまだ存在していない．そこで，各ミドルウェア対応のジョブ投入や監視機能をもつプラグインモジュールを実装した．ユーザにおいては，WFT を GUI 経由で使うことで，ジョブの投入監視機能に関しては，ミドルウェアの差を意識する必要はなくなる．

本報告では，はじめに設計と実装について述べる．ついで，典型的なユースケースとして，流体力学シミュレーションにおいてメッシュ形成などの前処理を LLS 計算機，シミュレーションを NIS 計算機，そして可視化などの後処理を LLS 計算機で実行する実験について報告する．

２．RENKEI-WFTの設計と実装

２．１グリッドミドルウェアの選択

LLS向けミドルウェアとして，研究室の計算資源に関する情報取得サービスは不要で，単にジョブの実行や監視ができ，軽くてインストールが簡単な単機能のミドルウェアが適切なため， UKe-Science が開発した GridSAM[2)]， および globus[4)] の WS-GRAM を選んだ．GridSAMは複数のPBS/SGE/globus等のジョブマネージャを選択できるが，2.3で述べるように，システム全体でGSI認証を行うため，globus(gt2)を選択した．

一方，NIS向けのミドルウェアとしては，NAREGIを選択した．一般ユーザは，基盤センタなどが所有する巨大な計算資源に関する情報を正確に掴むことはできない．そのため，計算資源の情報を提供する情報サービスや，

その情報をもとに，最適な資源にジョブを投入するスーパスケジューラ，複数サイトに跨るmpiであるGridMPI[15]などをサポートするNAREGIが適切と判断した．

２．２　システム構成

図1は，RENKEI-WFTを用いたシステム構成例を示している．研究室にある資源は，LLS向けポータルサーバ(LLS-Portal)と，研究室で使用可能な計算サーバである．計算サーバ上にはGridSAM，またはWS GRAMがインストールされている．LLS-Portalはウェブサーバとしてapache，アプリケーションサーバとしてtomcatがインストールされ，ワークフローツール（WFT）とワークフローエンジン(WF engine)が実装されている．AHS[17]は，プログラムのデプロイ等を行うために開発中のサービスである．NIS側のミドルウェアの資源は，文献8に述べられている．また，gfarmserver[13]は，LLS/NIS計算機間のデータ連携のために使う広域ファイルシステムである．

LLS 資源と NIS 資源の両方を利用するユーザは，LLS-Portal にブラウザでアクセスし,アプリケーションの登録からジョブの実行まで行う．その際，LLS-Portal サーバにある WFT，WF engine は，LLS 計算機に対するジョブの実行や監視及び結果の取得を直接行う．一方，NIS 側のファイアウォールポリシは一般に厳しく運用されているので，ジョブの実行の要求などは，直接のアクセスを行うのではなく，NAREGI-Portal の servlet に中継機能を実装して解決した．即ち，LLS から NIS に対するジョブ実行要求などは https プロトコルのみで行う．

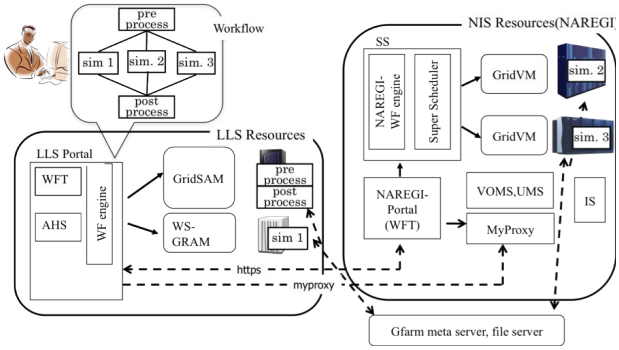


図 1 RENKEI-WFT システム構成

２．３　認証

認証はNAREGIでのGSI(Grid Security Infrastructure)ベースの方式を用いた．ユーザは，システムを利用する場合には，予めUMSサーバに保管されているユーザ証明書を用いて，MyProxyサーバにVOMSによる属性証明書付きのproxy証明書の作成と保管を行う．この操作はLLS-Portalから簡単行うことができ，その際，LLS-PortalはNAREGI-Portal経由で処理を実行する．ユーザが実際にWFT等を使う場合には，myproxyのidとpasshraseでLLS-Portalにsignon-onを行う．LLS-Portal及びNAREGI-Portalの中では，myproxy-idとpassphraseからproxy証明書を得る事ができる．LLS内の計算資源に対するジョブは，LLS-PortalがNAREGI内のMyProxyサーバからproxy証明書を得て，またNAREGI内の計算資源に対するジョブは，LLS-Portalから依頼されたNAREGI-PortalがMyProxyサーバからproxy証明書を得てジョブAPIに添付してジョブが実行される．

２．４　ワークフローツール

ユーザは，WFTのGUIを用いることにより，LLS上とNIS上のプログラム及びデータをアイコンとして表現し，アイコン間の依存関係を線で結ぶことでワークフローを表現する．そして，GUI上でワークフローの実行や実行状態の監視を行うことができる．ユーザがGUIで記述したワークフローは内部的にはNAREGI-WFML(NAREGI WorkFlow Markup Language)[18]として表現される．このWFMLは，コマンドラインによるジョブ投入でワークフローを指定する際にも使用するアプリケーションレベルの言語であり，ミドルウェアの基盤部分のアーキテクチャの変化に対応しやすい共通中間語として開発した言語である．WFTのアーキテクチャ依存部は，ジョブの投入時に，基盤ミドルウェアに対応したAPIに変換を行う．

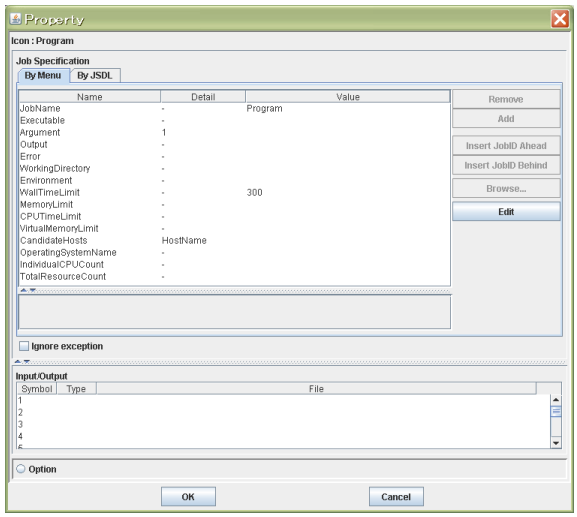


図2 JSDL エディタ

NAREGI-WFML の構文は，ワークフローにおける最小の仕事の単位であるactivityを定義する部分と，activity間の依存関係について規定する部分に分かれている．ジョブに関する定義は，activity要素の中にJSDLで定義される．JSDLはプログラム実行のためのジョブ要件（Executableのパス，引数や環境変数など）をXMLベースで記述する言語である．しかし，ユーザが直接記述するのはやや煩雑なので，図2のようなJSDL エディタを開発した．By Menu タグでは一般によく使用される要素が表形式で記述できる.もし，これ以外の要素を指定したい時や， JSDLが拡張された場合には， By JSDLタグを開くとBy Menuですでに入力した要素がJSDLに変換されているので，新規の要素を直接入力することにより対応できるようにした．

２．５　ワークフローエンジン

図3に，実装するWF Engineとジョブサービスサーブレット（jobserviceServlet）の構造を示す． 1 つのワークフロージョブに対して 1 つのインスタンスを作り， Java1.5 から導入されたExecutorServiceを用いたスレッドプールに

投入して並列処理を実現する．各インスタンスは，全体のワークフローを表現するWFMLから，各計算サービスに投入する部分を新たにWFMLとして作り直す．WS GRAMやGridSAMは，単一のジョブの実行サービスなので，単一のジョブのみを含むWFMLに変換する．一方，NAREGIのスーパスケジューラ(NAREGI-SS)は，複数のジョブを含むワークフローに対するサービスなので，一般にワークフローを表現するWFMLである．ジョブ実行インタフェースの共通化のためプラグイン化されたjobserviceServlet部が，切り出されたWFMLを，各サービスでのジョブ記述言語に変換し，ジョブ投入APIを用いて各ジョブ実行サービスに投入する．GridSAMを指定されたジョブでは，WFMLからJSDLを抜き出してGridSAMサービスに投入し，WS-GRAMの場合は，RSL(Resource Specification Language)に変換してWS-GRAMサービスにジョブを投入する．NAREGIプラグインでは，入力のWFMLを，そのままNAREGI-SSサービスに投入する．

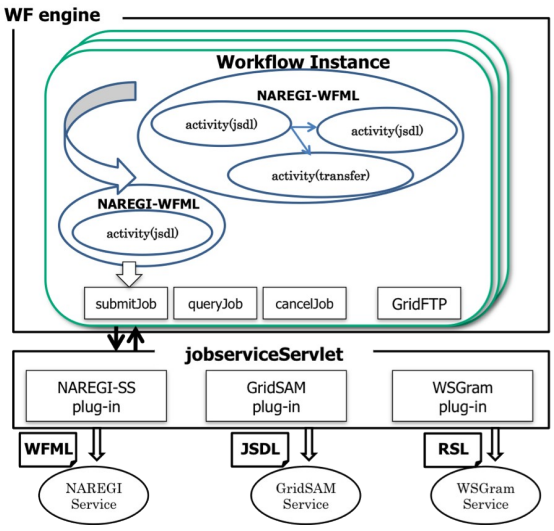


図3 ワークフローエンジン

WF Engineでは，ワークフロー制御においてジョブの状態を知る必要があるが，GridSAM/NAREGI-SSにはジョブ状態の変化に対するnotificationの機能がないため，ポーリングによりジョブの状態を検出した．各スレッドのrun()メソッドでは，ワークフローインスタンスの中で，実行可能なジョブに対してsubmitを行い，一定時間sleepした後，状態の監視を行うループとなっている．

jobserviceServlet部のプラグイン部に関しては，クラスライブラリに競合があるため，別々のwebアプリケーションとしてサーブレットに実装を行い，名前空間を分けることにより競合を回避した．

ジョブの状態(Pending, StageIn, Active..等)はミドルウェアによって異なるので，各プラグインモジュールは，NAREGI-SSの状態の定義にマッピングして返却した．また，ユーザに提示するlog等の情報は，NAREGIの形式と同一にしている．

３．実験

RENKEI-WFTの有効性を示すために，まず，RENKEI-WFTのジョブ起動性能と安定性の評価を行った．次に，流体力学用のユーティリティやアプリケーション集からなるOpenFOAMを利用して，前処理，流体計算と可視化を含む後処理の一連のワークフローに対して評価を実施した．

３．１　ジョブ起動性能

RENKEI-WFTでは，LLS資源のミドルウェアに負荷の軽いミドルウェアとしてWSGRAM/GridSAMを選択した．この目的が達成できているかどうかを評価するため，/bin/hostnameコマンドをRENKEI-WFTを使用したジョブ起動性能を測定した．図4に，ローカルバッチスケジューラとしてpbs(torque)を用いた場合の比較を示す．qsub, globusrun-wsは，pbs及びWS GRAMでのジョブサブミットコマンドを用いた場合， WFT(service name)は，RENKEI-WFTを通じて各種グリッドサービスを用いた場合の経過時間を示す．処理時間の内訳は各計測からの推定である例えば，WFT(WS-GRAM)とglobusrun-wsの経過時間の差がWFTによる処理時間である推定した．この結果から，WFTによるオーバヘッドの増加は各グリッドミドルウェアの処理時間に比べ無視ができ，LLS環境にNAREGIのようなスーパコンセンタ向きグリッドミドルウェアではなく，軽いWS GRAM等のミドルウェアを採用する効果が十分にあると結論できる．なお，NAREGIにおいては，50秒を超えるジョブ起動時間を必要としている．これは，NAREGI-SSは，情報サービスのデータをもとにしたマッチメイキングに時間を要していること，及びジョブの流量制御のために平均30秒に1つしかジョブを受け入れない制御を行っていることが原因である．しかし，NAREGIに対するジョブは，通常長大時間のジョブなので特に問題はない．

３．２　安定性

次に，RENKEI-WFTに多量のジョブを投入した時のシステムの安定性に関して評価を行った．

WFT(WS GRAM)を用いた場合，/bin/sleep 300 ジョブを4CPUのシステムへの投入において，特に問題もなくすべてのジョブは完了した(約11時間 300/4*500=37500sec)．なお，設定したThreadPoolの大きさは50である．WFT(GridSAM)の場合，ジョブ数が400程度までは完了したが，それ以上になると，オープンしたファイル数が限度を越えてLLS portalノードが動作しなくなった．調査の結果，GridSAM(globus ver2.1.4)では，１ジョブ投入すると1つのクローズされないsocketファイルが残る不具合があり，LLS Portalのtomcatサーバが立ち上がってから，過去の投入ジョブが一定限度を超えると，ファイルリソース枯渇のため動作しなくなることがわかった．なお，WS GRAMを使う場合にはこのような問題はなく，LLS-Portalノードにおける実行中の使用ファイル数は{

ThreadPool数＋定数　}であることを確認した．ThreadPool数だけ使用ファイル数が多くなるのは，ジョブの状態変化を受け付けるnotification機能を使用しているため，同時にジョブ監視が必要な数だけsocketファイルが必要となることによる．なお，上記の指摘に対し，GridSAM開発者からはver2.1.10で対策がなされる可能性があるという連絡を受けている．

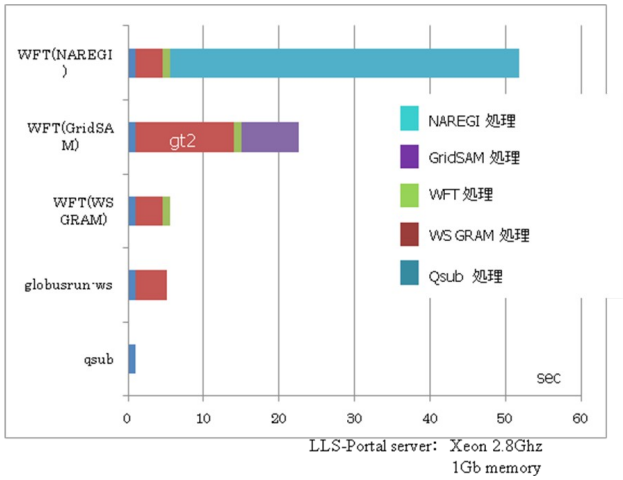


図4 ジョブ起動性能

図5は，LLS-PortalノードのCPU負荷状況を，mpstatコマンド（5秒間隔）を用いて測定した結果である．測定は，ThreadPool数を30，ジョブ監視インタバルを15secとした環境で，時刻0からジョブを連続60個投入した環境で行われた．横軸はmpstat間隔5秒を単位としたものであるが，450単位あたりで残りのジョブ数は30となり，それまでThreadPoolが満杯であったのが，しだいに，ThreadPool中のジョブがなくなりCPU負荷がさがっていくことがよくわかる．

これらの実験では，１つのワークフローが１つのジョブで構成されているものを行った．今後，１つのワークフローの中に並列に実行できるジョブが多数ある場合の評価が必要である．現在の実装では，同時にワークフロー制御されるワークフローインスタンス数の制御をスレッドプールで行っているが，１つのワークフローの中に複数の並列実行可能なジョブに関しては制限していないためである．

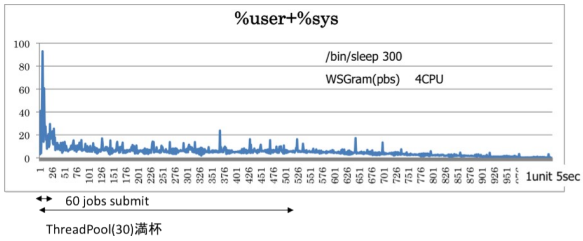


図5 LLS PortalのCPU負荷

３．３　OpenFOAMへの適用

RENKEI-WFTを用いてLLSとNISシステムの連携例を示すために，流体力学のアプリケーションへの適用を試みた．OpenFOAM(Field Operation and Manipulation)[14]と呼ばれる，流体力学の多数のソルバや，可視化のための単純データ操作から，メッシュ生成や可視化に至る様々な前・後処理のユーティリティが含まれるオープンソースをLLS環境とNIS環境にインストールを行った．

a)　OpenFOAMのケースファイル

OpenFOAMの１つのシミュレーションで必要な基本的なディレクトリ構造を図6に示す．このトップディレクトリを，今後，ケースディレクトリと呼ぶ．OpenFOAMのあらゆるコマンドにおいて，このディレクトリで実行する場合には引数は不要であるが，他のディレクトリで実行する場合は，引数で指定する必要がある．

```
<case>
  | system
  |      | controlDict
  |      | fvSchemes
  |      | fvSolution
  | constant
  |      |.....Properties
  |      | polyMesh
  |            | blockMeshDict
  |            | points
  |            | cells
  |            | boundary
  |            |.....
  |
  |  時刻ごとのdirectoryとその中に物理量ファイル
```

図6 ケースディレクトリとファイル

constantディレクトリの中には，メッシュと，物理的性質を指定するpolyMeshサブディレクトリとtranportPropertiesなどのファイルがある．polyMeshディレクトリには，メッシュを指定するblockMeshDictファイルがある．図において，points,cells,..などのファイルは，メッシュ生成コマンドがblockMeshDictファイルを入力として生成したものである．systemディレクトリは，解析手順のためのパラメータ設定をするもので，以下のようなファイルがある．シミュレーション開始，終了時刻，時間ステップや出力をコントロールするcontrolDictファイル，解析に使われるスキームを記述しているfvSchemes,ファイル，そして，方程式のソルバ，許容誤差などを制御するfvSolutionファイルである．時刻ディレクトリはcontrolDictで出力を指定された時刻ごとディレクトリであり，計算領域における物理量の計算結果についてのデータファイルをもっている．

b)　ケースファイルの配置

このケースファイルを，前処理，シミュレーション処理，後処理で参照して計算を進めることになる．今回の実験では，著者の研究室のある神保町にLLSシステムを，40km東にある西千葉の国立情報学研究所の計算機室にNIS(NAREGI)を配置したものを利用している．前，後処理をLLSで，シミュレーションをNISシステムで実行する

ことから，ケースファイルをどのように配置するかが問題となる．案１は，共有ファイルシステムであるgfarm上にケースファイルを一つだけ置くものである．案２は，LLSとNISの計算のためにそれぞれのローカル領域を割当て，前後処理とシミュレーション処理の間でケースファイルをtarで固めてgfarm経由で転送し，それぞれのケースファイル上で展開する方式である．両方法でワークフローを作成して動作することは確認した．どちらの方法が性能上，望ましいかはプログラムやシステム環境に依存する．

c) ワークフローの作成

OpenFOAMのパッケージのチュートリアルにある，Lid-driven cavity flowを例題にとりあげる．前処理であるメッシュ生成と，計算結果の可視化をLLS計算資源で行い，流体計算をNIS計算機で実施する．なお，以下ではケースファイルは案2の持ち方の場合を説明する．

その際のユースケースとして，ユーザは使用するアルゴリズムは変えないが，以前に実行したシミュレーションの次の時刻からシミュレーションを継続実行したい，あるいは，最初は粗いメッシュで実行し，その次は細かいメッシュで実行したいなど，粗いデータから内挿したデータを初期値として実行することが簡単にできるようにワークフローを作成した．図8にGUIで記述したワークフローと実行結果のスナップショットを示す．

以上のユースケースでは，ケースファイルのcontrolDictやblockMeshDictファイルをワークフロー上から容易に変更できるようにすればよい．そこで，図8の①②のようにデータアイコンとして表現した．このようにすることでそれぞれのcontrolDictやblockMeshDictファイルをワークフローツール上で容易に修正できる．③はプログラム実行を表現するプログラムアイコンであるが，このアイコンが指定するExecutableとしては，図7のようなシェルファイルをWFTから作成し，計算機上に配置したファイルパスを指定する．ここで， a.のblockMeshコマンドはOpenFOAMのメッシュ生成コマンドである．ケースファイルのファイルパスとして，プログラム実行ディレクトリ直下のdataディレクトリに配置した．その後の処理は，data直下のファイルを纏め(b)，gfarmファイルをマウントし(c, d)，纏めたデータをgfarmファイルに転送し(e, f),その後，アンマウントをする（g, h）という処理からなる．なお，①②から③への黒線はファイルのデータステージングを表しており，アイコン③の中で指定されたファイルパスに，プログラム実行前に転送することを意味している．④のアイコンでは，メッシュの生成結果を見るためDISPLAY変数の設定の後，可視化コマンドparaFoamを実行するシェルプログラムを指定している．即ち，LLS内の計算資源がX-clientとなっているが，通常ユーザ端末は同じネットワーク内にあることが多いので，firewall上の問題は少ないと考えられる．同様に，OpenFOAMの層流流れの解析のicoFoamコマンドを⑤で使用し，コマンド実行前後で，gfarmファイルへのケースファイルの読み込みと書き出しを実施する．⑥は，⑤で計算された計算データを読み込み可視化するためのコマンドである．④と，gfarmからの結果データのファイルのコピー処理があることだけが異なる．

d) ワークフローの利用の仕方

c)で作られたワークフローを実行することで，LLS/NISシームレスにジョブを実行することができた．一度ジョブが実行されると，ケースファイルにはそれまでのデータが存在していることから，①のデータのシミュレーション開始時刻と終了時刻を修正してワークフローを投入すれば，継続実行ができる．メッシュを細かくしたり，勾配メッシュにしたりするのであれば，②のメッシュの指定を変えればよい．

また，既存の計算結果の可視化だけを行いたいのであれば，アイコン⑥の部分だけを実行させればよい．しかし，RENKEI-WFTには，現在のところ選択したアイコンだけを実行させる機能はない．その代わりに，図8のワークフローから必要な部分をGUI上でCopy & Pasteして，⑥だけのアイコンをもつワークフローを新たに作って実行すれば同等のことができる．

```
    #!/bin/sh
a.  blockMesh  -case data

b.  tar cfz mesh.tar.gz data
c.  mkdir /tmp/demo$$
d.  gfarm2fs /tmp/demo$$
e.  cp mesh.tar.gz
      /tmp/demo$$/gfarmhome/usrwft03/openfoam/mesh.tar.gz
f.  rm mesh.tar.gz
g.  fusermount -u /tmp/demo$$
h.  rmdir /tmp/demo$$
```

図7 シェルスクリプト（メッシュ生成）

e) 並列実行に関する問題点

OpenFOAMには，計算領域分割による並列実行がサポートされている．領域分割のためには，LLSにおいて，メッシュ生成後に，設定ファイルdecomposeParDictファイルに分割方法を設定し，decomposeParコマンドを実行すればよい．そうすると，プロセッサごとのメッシュ情報がケースファイルに生成される．シミュレーションは，NISにおいてmpirunコマンドで実行すれば領域ごとに並列処理ができる．計算結果はプロセッサごとのファイルに得られるので，結果の融合にはLLSにおいて，可視化のコマンド実行の前にreconstructParコマンドを用いる必要がある．

現在のところ，RENKEI-WFTでは前後処理は問題なく動作するが，NISにおけるmpi実行ができないという問題がある．これは，OpenFOAMはmpiの実行としてパブリックな実装であるOpenMPI[16]を利用しているが，標準のNAREGIではmpiとしてGridMPIしかサポートされていないことによる．今後，OpenFOAMをGridMPI[15]で構築できないか検討する必要がある．

５．まとめと今後の課題

本研究では，研究室（LLS）および情報基盤センター（NAREGI）の計算機間でジョブを連携して実行するワークフローに関する設計と実装を行い，図 1 に示す環境を神保町(LLS)と西千葉（NIS）に構築して評価を行った．評価においては，流体力学のツールキットとして著名な OpenFOAM のチュートリアルにある例題を選択した．メッシュ生成の前処理を LLS 資源で，シミュレーションを NIS 資源で，可視化の後処理を LLS 資源で処理するワークフローを WFT で記述し，実行，監視を行う実験を行い RENKEI-WFT の有効性を確認した．

実験において，ワークフロー内に単一ジョブがあるワークフローを多数投入し実行できることは確認した．しかし，流量制限のロジックはワークフロー単位に行われているため，１つのワークフローの中に並列に実行できるジョブがある場合の対応は行われていない．この点について，今後の検討と実装が必要である．また，直接，本研究とは無関係であるが，OpenFOAM において NAREGI 資源で並列実行が可能とするため，mpi として GridMPI[15]が使えるように構築することも課題である．

謝辞：本研究は，文部科学省の次世代IT基盤構築のための研究開発「e-サイエンス実現のためのシステム統合・連携ソフトウェアの研究開発」（研究コミュニティ形成のための資源連携技術に関する研究）の成果である．

参考文献

1) UNICORE,http://www.unicore.eu/
2) GridSAM http://www.www.omii.ac.uk/wiki/GridSAM/
3) Condor http://www.cs.wisc.edu/condor/
4) Globus Toolkit,http://www.globus.org/toolkit
5) G.von Laszewski, I.Foster, J.Gawor and P.Lane, “A Java Commodity Grid Kit,” Concurrency and Computation: Practice and Experience, vol.13, no.89, pp.643-662, 2001
6) Taverna: workbench http://taverna.sourceforge.net/
7) I.Taylor, M.Shields, I.Wang and A.Harrison, “Visual Grid Workflow in Triana,” Journal of Grid Computing, vol.3, no.3-4, pp.153-169,2005
8) NAREGI http://www.naregi.org/
9) OGF(Open Grid Forum) http://www.ogf.org/
10) HPCBP(HPC Basic Profile) http://www.ogf.org/documents/GFD.114.pdf
11) JSDL(Job Submission Description Language) http://www.ogf.org/documents/GFD.56.pdf
12) BES(OGSA Basic Execution Service) http://www.ogf.org/documents/GFD.108.pdf
13).....Gfarm http://datafarm.apgrid.org/index.ja.html
14).....OpenFOAM http://www.openfoam.com/
15).....GridMPI http://www.gridmpi.org/
16).....Open MPI http://www.open-mpi.org/
17). 宇佐見仁英　他：異種グリッドミドルウェアに跨るアプリケーションホスティングサービス（AHS）の設計と実装　SWoPP 仙台 2009
18). 田中義一　他：異種グリッドミドルウェア間に跨るワークフロージョブの実行方式と実装　SWoPP 仙台 2009

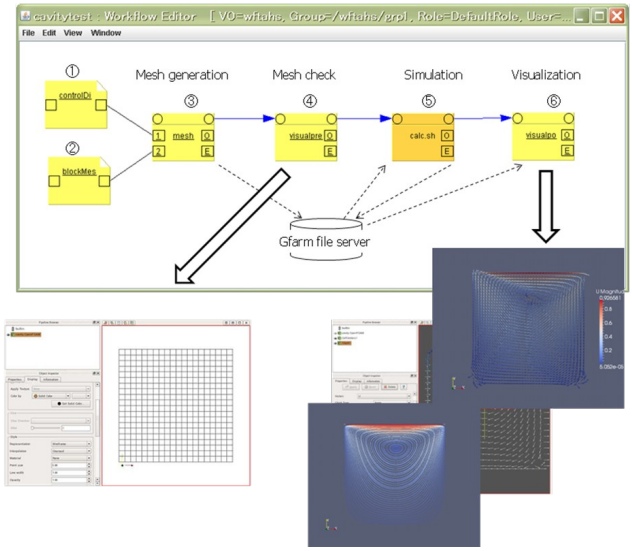


図 8 Cavity 流れのスナップショット

異種グリッド環境におけるアプリケーションホスティングサービス（ＡＨＳ）の試作

PROTOTYPE OF APPLICATION HOSTING SERVICES(AHS)
ON MULTIPLE GRID ENVIRONMENTS

大西尚樹[1)]，水澤有里[1)]，金澤宏幸[1)]，恒川隆洋[1)]，宇佐見仁英[2)]
Naoki Onishi, Yuri Mizusawa, Hiroyuki Kanazawa, Takahiro Tsunekawa and Hitohide Usami
1)富士通株式会社 ＴＣソリューション事業本部(onishi.naoki, yuri, kanazawa.h, tsunekawa.takah@jp.fujitsu.com)
2)玉川大学（usami@lab.tamagawa.ac.jp）/ 国立情報学研究所(usami@grid.nii.ac.jp)

This paper describes the implementation and evaluation of Application Hosting Services (AHS) capable of covering multiple grids composed of different grid middleware systems. A service which covers heterogeneous grid middleware environment must be capable of communicating with protocols that each middleware understands. AHS is a software environment designed for deploying and sharing scientific applications on NIS (National Infrastructure System) resources, and on LLS (Laboratory Level System) resources. The NIS is based on the NAREGI, while many of the LLS is configured using GridSAM. Hence, in order to communicate with both grid systems, the AHS has implemented a scheme for translating application's information from NAREGI to GridSAM, and vice-versa. The evaluation of AHS was done not only on functionalities, but also on the network security issues. The tests were conducted by locating at two distant locations connected by WAN, with firewalls in between.

***Key Words** : Problem Solving Environment, Grid Provisioning, Application Deployment, Workflow*

1. はじめに

サイエンスグリッドは，ナノテクノロジー，バイオテクノロジーなど，これからの日本を支える先端科学技術のプロセスを大きく変える可能性を持っている．我が国のサイエンスグリッドを推進するプロジェクトとして「e-サイエンス実現のためのシステム統合・連携ソフトウェアの研究開発～研究コミュニティ形成のための資源連携技術に関する研究～RENKEI(REsources liNKage for E-scIence)プロジェクト」[1)]がスタートしている．一方，グリッド先進国の米国では，グリッド上に特定領域の研究コミュニティを一つの仮想組織として形成し，その領域で有効な様々なサービスの提供やアプリケーション，ノウハウの共有を可能とすることによって研究者の利便性を高めると共に，グリッドの煩わしさを解消させようとするサイエンスゲートウェイという考え方がテラグリッドプロジェクトを中心に推進されている．

我々はRENKEIプロジェクトの１グループとして，サイエンスゲートウェイを実現するため，AHS(Application Hosting Services) [2-4)]を研究開発している． AHSは，GridSAM[5)]等の軽量級のグリッドミドルウェアを使った研究室レベルの計算資源を中心とした小規模なグリッド環境(以下，LLS: Laboratory Level System)やNAREGI[6)]等のheavy-dutyなグリッドミドルウェアを使った情報基盤センター等の大規模なグリッド環境(以下，NIS: National Infrastructure System)など，異種のグリッド環境を跨って形成される研究コミュニティVO(Virtual Organization)におけるアプリケーションのホステイングサービスを提供する．

本論では，AHSの試作として，NAREGI-PSE[7-10)]からの拡張における２つの課題とその解決を中心に報告する．１つ目の課題はLLSに対応したシステム拡張におけるポイントとなるRENKEI-WFT[11)]との連携方式，２つ目の課題は実運用環境を構築する際にポイントとなる拠点を跨った異種グリッド環境の検証である．

第２章では，AHSの概要と試作における２つの課題について説明し，第３章でAHSの実現方式について２つの課題の解決を中心に説明し，第４章で評価を行い，第５章でまとめを述べる．

２．アプリケーションホスティングサービス(AHS)

２．１　AHSの目的

AHSは，研究室レベルのLLS環境と情報基盤センター等によるNIS環境が混在する環境において，コミュニティに所属する研究者が，VO属性によるアクセス制御のもとで開発したアプリケーションを共有しながら，コンピュータシミュレーションによる研究開発を進められる環境の提供を目的としている．そのイメージを図１に示す．

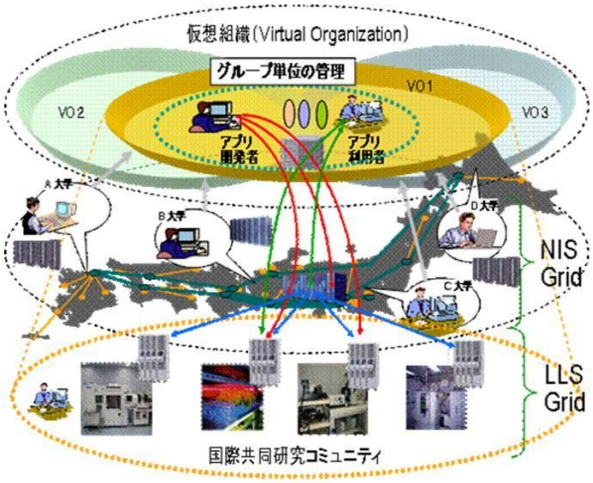


図１　ＡＨＳの目的

アプリ開発者は，研究コミュニティで利用可能なNIS資源やLLS資源にアプリケーションを配置しておく．アプリ利用者は，例えば，小規模解析ならLLSで，大規模な場合はNIS資源でとのように適時使い分けて同じアプリケーションを同じ操作環境で実行することができる．

２．２　RENKEI プロジェクトと AHS

RENKEIプロジェクトにおけるe-サイエンス計算連携技術の全体概念図を図２に示す．我々はe-サイエンス計算連携技におけるアプリケーションのコミュニティでの共有環境としてAHSを担当している．

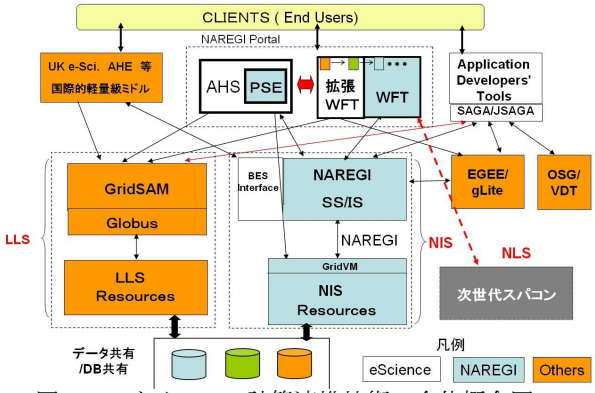


図２　e-サイエンス計算連携技術の全体概念図

e-サイエンス計算連携技術は，NAREGIコンポーネントの内，基本的にはNAREGI-PSEとWFT(Workflow Tool)を複数のグリッド環境に対応できるように拡張するものである．NIS環境に関しては，スーパーコンピュータセンター等における大規模・高信頼性のNAREGI Middlewareが導入された環境を，LLS環境での資源管理に関しては，英国のOMIIプロジェクトで開発されたGridSAMをベースに全体の仕様を検討する．また，NIS及びLLSでの全体のジョブの投入処理に関しては，国際標準として検討されているBES(Basic execution system)に準拠した仕様とする．

２．３　NAREGI-PSE の機能

AHSの機能のベースとなるNAREGI-PSEの機能について説明する．NAREGI-PSEは，複数のユーザが協同して研究を行うため，アプリ開発者がアプリケーションの登録やデプロイ等を行い，アプリ利用者がカタログ化された情報からジョブを実行のための情報をインポートできる機能を提供する．NAREGI-PSEはNAREGI Middlewareが導入されたNIS環境において，以下の機能がある．

１）アプリケーションの登録・管理
２）アプリケーションの実行前準備（コンパイル）
３）仮想組織(VO)内でのアプリケーションの共有
４）アプリケーションのグリッド環境への配置と実行確認

なお，NAREGI-PSEのアプリケーション管理機能はグリッドの標準化団体であるOGF[12]のACS(Application Contents Service)-WG[13]の仕様に準拠し開発されている．

２．４　NAREGI-PSE から AHS への拡張における課題

NAREGI-PSEからAHSへのシステムの拡張を進めるにあたり，主に２点の課題があった．

一つ目は，「RENKEI-WFTとの連携方式」である．NAREGI-PSEは，開発者が配置済みのアプリケーションを利用者がジョブ実行できるように，OGF標準のジョブ記述形式JSDL(Job Submission Description Language)[14]を含む情報をNAREGI-WFTからインポートできる仕組みを提供している．NAREGI-WFTは受け取ったJSDLの情報を元にNAREGI-SS[15]へジョブの投入を依頼する仕組みとなっている．しかし，LLSで利用するグリッドミドルウェアGridSAMではJSDLのうちサポートしている範囲がNAREGI-SSとは異なっている．またLLS環境には情報サービスが存在しない点も異なっている．この違いに起因する理由から，LLS環境へのジョブ投入に関してもNAREGI-PSEと同様の方式にて対応する場合，AHSからインポートしたアプリケーションは実行先のホスト名を解決できないためジョブ投入に失敗するという問題がある．

二つ目は，「拠点を跨った異種グリッド環境の検証」である．LLSのグリッド環境とNISのグリッド環境を接続した環境を構築する場合に，拠点間にはファイアウォールが存在するため，導入時に問題が発生することが想定される．そのため，試作段階において検証・評価を行うこととした．

３．実現方式

３．１　システム構成

RENKEIプロジェクトのシステムの全体構成をもとにNAREGI-PSEを拡張し，図３に示すシステム構成にてAHSを試作した．

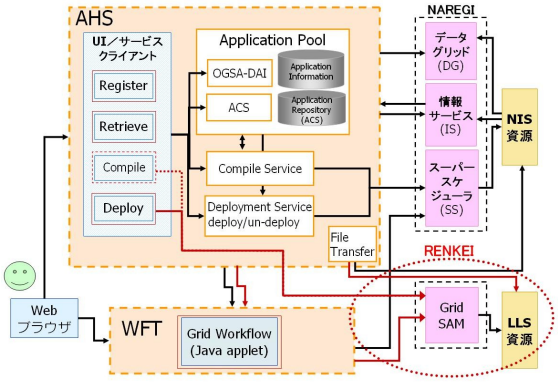


図３　試作したＡＨＳのシステム構成

試作したAHSのシステムは，NAREGI-PSEと同様にミドルウェアにはGlobus Toolkit4を使用し，認証にはNAREGIのGSI(Grid Security Infrastructure)ベースの方式を用いた．Deploy機能に関しては，NAREGI-SS経由でアプリケーションを配置する機能と同様の仕組みで，GridSAM経由でLLS資源へアプリケーションを配置できるようシステムを拡張した．LLS環境には，情報サービスが存在しないため，ファイルベースで計算資源の情報を保持しておく方式とした．コマンドラインインターフェースに関してもLLS環境に対応した拡張を行った．

３．２　解決１：RENKEI-WFTとの連携方式

AHSとRENKEI-WFT間のインタフェースとなる，JSDLを含むXML Schemaを検討した．一般的にJSDLは，計算ジョブをサブミットする際の要件として，アプリケーション情報であるExecutableやArgument等や，資源要求情報であるOSやトータルCPU数等や，データステージング情報を記述する．LLS環境のグリッドミドルウェアあるGridSAMは，JSDLの情報のうち資源要求情報であるResources要素をサポートしていないため，実行候補のホスト情報であるCandidateHostsを指定できない．そのためアプリケーションのデプロイ済みホスト名情報をRENKEI-WFTに伝達する方法を検討した．また，GridSAMはDataStaging要素をサポートしているため，データステージングに関する情報の取り扱いついて検討を行った．検討結果の例を図４に示す．

```
<?xml version="1.0" encoding="UTF-8"?>
<ns5:application
  xmlns:ns5="http://www.e-science.org/ws/2009/08/ahs-wf-application"
  no="PSE_xxxxxxx.naregi.org_1249974403510" type="LLS">
  <ns5:submittingServer>xxxxxxx.naregi.org</ns5:submittingServer>
  <JobDefinition xmlns="http://schemas.ggf.org/jsdl/2005/11/jsdl"
    xmlns:ns2="http://www.naregi.org/ws/2005/08/jsdl-naregi-draft-02"
    xmlns:ns4="http://schemas.ggf.org/jsdl/2005/11/jsdl-posix">
    <JobDescription>
      <Application>
        <ApplicationName>SampleApp2</ApplicationName>
        <Description>AppAbs</Description>
        <ns4:POSIXApplication>
          <ns4:Executable>/home/onishi/a.out</ns4:Executable>
          <ns4:Argument>arg1 arg2</ns4:Argument>
          <ns4:Output>stdout.txt</ns4:Output>
          <ns4:Error>stderr.txt</ns4:Error>
          <ns4:WorkingDirectory>/tmp</ns4:WorkingDirectory>
          <ns4:Environment name="ENV1">aaa</ns4:Environment>
          <ns4:Environment name="ENV2">bbb</ns4:Environment>
          <ns4:WallTimeLimit>300</ns4:WallTimeLimit>
          <ns4:MemoryLimit>1024000</ns4:MemoryLimit>
          <ns4:CPUTimeLimit>300</ns4:CPUTimeLimit>
        </ns4:POSIXApplication>
      </Application>
    </JobDescription>
  </JobDefinition>
  <ns5:input no="1" path="input1.dat" />
  <ns5:input no="2" path="input2.dat" />
  <ns5:output no="3" path="output1.dat" />
</ns5:application>
```

図４　RENKEI-WFTとやり取りするJSDLの例

アプリケーションのデプロイ済みホスト名情報は，JSDLのスキーマの外で１つのホスト名情報を受け渡し，実行パスの情報はJSDLのExecutable要素で情報受け渡すよう設計した．また，DataStaging要素については，NAREGIと同様の機能を維持する方針とし，JSDLのスキーマの外で入出力のワークフローのデータアイコンの情報を受け渡す方式とした．

３．３　解決２：拠点を跨った異種グリッド環境の検証

二つ目の課題の解決のため，拠点を跨った環境構築を実施した．玉川Lab（東京）に小規模構成のクラスタを導入し実証用のLLS環境とし，国立情報学研究所（以下，NII）のNAREGI Center（千葉）に既存のサーバ群をNIS環境と見立てた実証評価環境を構築した．実証評価環境の構築を図５に示す．

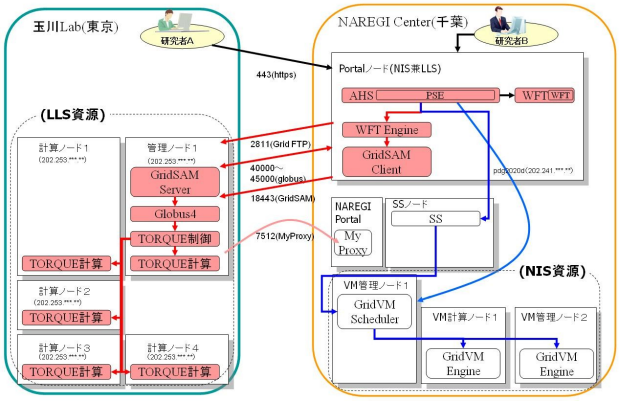


図５　拠点を跨った実証評価環境の構成

実証評価環境の構築を通し，いくつかの運用上の課題洗い出した．まず，拠点間のシステム構成について検討したところ，Portalノードを LLS環境側へ配置する場合，AHSとNAREGI Middlewareの結びつきが強いため，PortalノードとPCクラスタ毎に存在するGridVM管理ノード間に多数の穴あけが必要となるという問題があった．実証評価環境はPortalノードをNIS環境側へ配置することとしたが，今後拠点間のシステム構成について見直す必要がある．また，玉川LabとNAREGI Centerの拠点間でGridFTPの5000ポート以上のファイアウォールの穴あけが必要となった．実証評価環境ではファイアウォールの設定で対応したが，ネットワークのポリシーによっては受け入れられない組織がある事も考えられ，またシステム管理者にも作業負担をかけることとなるため，対策が必要となることが洗い出された．

４．実証評価

複数の研究者がLLSとNISの異なるグリッド環境・拠点間において，AHSの基本的な機能を一通り実行できることを確認するため，図６に示す構成・手順にて実証実験を実施した．

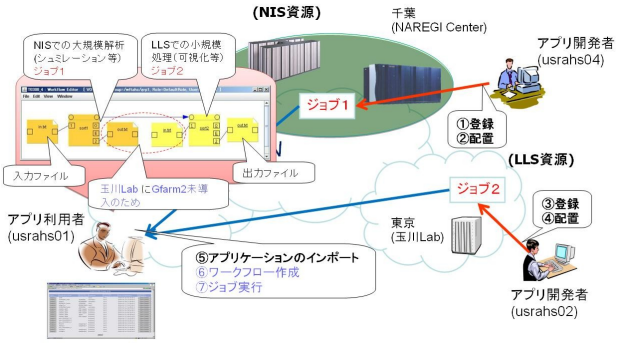


図６　実証評価

AHSの基本的な機能を確認するシナリオとして，研究コミュニティにおいて次の手順の動作を確認した．

１）NIS環境を利用しているアプリ開発者usrahs04が，自身が開発したアプリケーションをAHSへ登録し，NIS資源にデプロイする．

２）LLS環境を利用しているアプリ開発者usrahs02がアプリケーションを登録し，LLS資源にデプロイする．

３）２人のアプリ開発者と同じVOに所属するアプリ利用者usrahs02が，NISおよびLLSに配置されたアプリケーションをインポートし，ワークフローを作成し異種グリッドを跨って順にジョブを実行する．

本実証ではジョブの動作を確認するための試験的なアプリケーションを利用したが，例えば大規模解析をNIS資源で実施し，可視化処理等のポスト処理をLLS資源で実施することをシームレスに行う事が可能である．

５．まとめ

RENKEIプロジェクトでは，NAREGIの下方展開ということで，研究室レベルの計算資源LLSにもグリッド環境を拡張し，研究者の日々の研究にダイレクトに利活用できる環境を検討してきた．RENKEIプロジェクトにおいて我々は，NAREGI-PSEの拡張版として複数のグリッド環境に跨って仮想組織内でのアプリケーションを共有するアプリケーションホスティングサービス（AHS）の研究開発を推進している．

本論では，プロトタイプ版のAHSを開発し，実利用に近い異なった組織，異なったグリッドミドルウェアで運用されるLLSとNISとの分散環境で，AHSの基本機能を検証した．特にRENKEI-WFTとの連携方式の仕様を検討しRENKEI-WFTからインポートしてLLS資源へジョブ実行可能なJSDLを含むスキーマをまとめた．また，玉川LabとNAREGI Centerを跨った異種グリッド環境を構築して検証を行い，運用上の課題を洗い出す事により，最終年度に向けたシステム実現の目処を立てることができた．

今後は，プロトタイピング版を用いた実システムでの検証を実施しながら，2011年度の提供開始に向けた開発を実施して行く予定である．今後2年間のRENKEIプロジェクトを通して，より実用的なシステムの構築，および知識ベースを具備したグリッドPSEの世界標準化などを図って行きたい．

謝辞：本研究の一部は，文部科学省の平成20年度・21年度科学技術試験研究委託事業「ｅ―サイエンス実現のためのシステム統合・連携ソフトウェアの研究開発」のテーマ「研究コミュニティ形成のための資源連携技術に関する研究（アプリケーション共有方式の方式に関する研究）」の成果である．

参考文献

1)RENKEI http://www.e-sciren.org/

2)宇佐見，金澤：複数グリッド環境におけるアプリケーションホスティングサービス（AHS）の一考察，PSE Workshop2008，pp18-21，2008/9/1

3)宇佐見，大西，水澤，金澤:異種グリッドミドルウェアに跨るアプリケーションホスティングサービス（AHS）の設計と実装，SWoPP仙台2009，2009

4)宇佐見，大西，水澤，金澤，恒川:異種グリッドミドルウェアに跨るアプリケーションホスティングサービス（AHS）の実装と評価，計算工学講演会論文集，Vol.15，(2010年5月)

5)GridSAM http://www.omii.ac.uk/wiki/GridSAM

6)NAREGI http://www.naregi.org/

7)宇佐見，山田，宮原，藤崎，早勢，川田；サイエンスグリッドにおけるNAREGI-PSEのProvisioning方式，計算工学講演会論文集　Vo2.0（2005年6月）

8)金澤，山田，宮原，早勢，川田，宇佐見：グリッドサービスを基盤とした問題解決環境　“NAREGI-PSE”，計算工学講演会論文集　D-3-4（2006年6月）

9)宇佐見，金澤，宮原，藤崎，川田：研究コミュニティにおけるNAREGI-PSEのアプリ共有方式，計算工学講演会論文集　D-10-1（2007年5月）

10)大西，金澤，大津，川田，宇佐見：協同研究のためのNAREGI-PSEのアプリケーション共有・配置機能，PSE Workshop2008，pp15-18，2008

11). 田中，合田：異種ミドルウェア間に跨るワークフロージョブの実行方式と実装，情報処理学会HPC研究会SWoPP仙台2009，2009

12).....OGF(Open Grid Forum) http://www.ogf.org/

13).....ACS-WG https://forge.gridforum.org/projects/acs-wg/

14)JSDL(Job Submission Description Language) http://www.ogf.org/documents/GFD.56.pdf

15)畑中,中野,井口,大野,佐賀,秋岡,中田,松岡: OGSAアーキテクチャに基づくNAREGIスーパースケジューラの設計と実装，情報処理学会研究報告　Vol.2005 No.57(20050602) pp33-38, 2005

地上デジタル放送用シミュレータ開発における効率的な評価環境の検討

Study on an effective evaluation environment for the digital broadcasting simulator development.

梅田雅敬，古舘英樹，岩松隆則
Masataka UMEDA, Hideki FURUDATE and Takanori IWAMATSU
{umeda.masataka, furudate.hideki, iwamatsu }@jp.fujitsu.com
株式会社富士通研究所　（〒211-8588 神奈川県川崎市中原区上小田中 4-1-1）

We use Link Level Simulator (LLS) to develop for the digital broadcasting. The LLS need to read the transmission data during executing it. However, the problem occurred to execute the LLS on the cloud system because the transmission data is too large. In this paper, we explain the efficient evaluation environment made to solve the problem.

***Key words**: Simulator development, cloud system, Digital broadcasting, ISDB-T,*

1. はじめに

放送メディアの高画質化，高機能化を目的として，放送のデジタル化が世界各国で推進されている．国内では2003年12月より三大都市圏でサービスが開始され，その他の地域でも順次開始されている．日本の地上デジタル放送は，国際規格の一つになっているISDB-Tに基づいて行なわれている．

我々の部門では地上デジタル放送受信技術の開発作業効率化のために，シミュレータを用いて仕様検討や評価試験を行なっている．しかし， 地上波デジタル放送は演算するデータが膨大であるため，１ジョブ当たりのシミュレーション実行時間が膨大になる．この実行時間を短くすることが開発期間短縮のための課題と考える．この課題に対して，我々は富士通が開発したCAD-Gridを利用して，シミュレーション実行時間の短縮を行なってきた．CAD-Gridは，ユーザのニーズである「シミュレーションを容易かつ効率的に運用できる」システムを目指してきたクラウド環境とも呼べるシステムである．そのためクラウド上のノードに対するリソース管理やジョブ投入/実行などの操作をユーザに意識させることなく実行することができる．

これまで開発してきたシステム同様に，地上デジタル放送用シミュレータにCAD-GRIDを適用したところ，データファイルの転送負荷がセントラルサーバに集中する問題が生じた．これは，シミュレーションで使用する送信データを実行ノードとは異なる場所に格納しているデータファイルから読み込む仕様としていたためである．そこで，データファイルをジョブ実行するノードに事前に格納しておく運用に変更したが，ノード数が多くデータファイルの格納漏れが新たな問題となった．本稿では，このデータファイル格納漏れ問題が発生した背景とそれらを解決するために行なった施策について述べる．

2. 地上デジタル放送

2.1. 地上デジタル放送規格 ISDB-T

日本の地上デジタル放送であるISDB-Tは，１つのチャンネルを13セグメントに分割し，それらのセグメントを幾つか束ねて映像やデータ，音声などの送信を行う．図１に示すように，13セグメントの内，中央１セグメントをモバイル端末向けのワンセグ放送として，残りの12セグメントをハイビジョン放送用のフルセグ放送として利用している．なお，フルセグ放送は，移動受信時の電波の瞬断なども考慮した規格が定められているため，車載用カーナビなどの移動端末でも視聴することが可能である．

世界の標準化動向としては，日本のISDB-T以外に欧州のDVB-Tと北米のATSCがある．特に，欧州のDVB-Tは世界の多くの国が採用している．日本のISDB-Tは2006年6月22日に南米最大の経済大国であるブラジルが採用したことから，南米諸国を中心にその市場が広がりつつある．

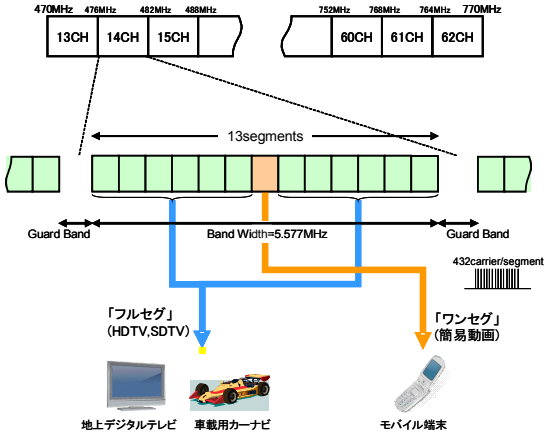


図 1. 地上デジタル放送のチャネル割当

2.2. 地上デジタル放送受信技術開発の課題

地上波放送では衛星放送と異なり，ビルなどの建物からの反射波を多く受信する．このため，放送局から受信端末に届くまでに多数の経路を経た電波同士が干渉し合うマルチパスフェージング環境となる．ISDB-T ではこのようなマルチパス妨害波対策として，変調方式に OFDM(Orthogonal frequency division multiplex)方式を採用している．OFDM 方式とは多数の搬送波(サブキャリア)を互いに直交するように並べて送信するマルチキャリア伝送の一種であり，ISDB-T では 1 セグメント当たり 432 本の搬送波を周波数方向に重ねて伝送する方法を取る．そのため，ワンセグ，フルセグ共に受信するためには，計 5616 本の搬送波分の復調処理を行う必要があり，膨大な演算量を必要とする．また更に，端末の高速移動による受信品質の劣化に対応するため，高度な符号化処理や誤り訂正処理などの機能を有している．その信号処理にも多大な演算量が必要となる．以上の理由から開発する技術の回路規模は膨大であり，検証評価を行うシミュレータの実行時間も長時間に及ぶ．

2.3. CAD-Grid の利用について

我々の部門では，開発期間短縮を目的として，弊社のグリッドコンピューティングである CAD-Grid を利用している．地上デジタル放送用受信技術開発においても CAD-Grid を利用した．図 2 は CAD-Grid の概略図である CAD-Grid では，まずユーザは Web ポータルサイトにアクセスし，自分の実行したいシミュレーションプログラムと，プログラムの実行を定義するスクリプトファイル(OJC ファイル)をセントラルサーバに投入する．OJC ファイルには各ノードに投入するプログラム名とジョブの投入条件が記述されており，OJC ファイルが実行されるとセントラルサーバはその投入条件に従い，最適なノードを選択し，シミュレーションの実行，結果出力が行なわれる．結果，ユーザはジョブがどこで処理されたかを意識することなく，効率的にシミュレーションを行うことができる．また，OJC ファイルは perl 言語や sh 言語で記述できるため，それらを利用してシミュレーション条件が異なる大量のジョブを一度に流すことができる．我々の部門では，この機能を利用して CAD-Grid ポータルに月平均 12,000 程度のジョブを投入している．

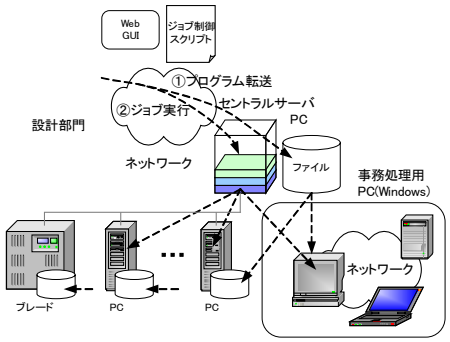


図 2. CAD-Grid 概略図

3. 地上デジタル放送受信技術の開発

本章では，地上波デジタル放送受信技術の性能検証のために新たに開発した LLS(Link Level Simulator)の概要と CAD-Grid に適用時に発生した問題点について述べる．

3.1. Link Level simulator

LLS とは，送受信部の処理と伝搬路を C++言語を用いてモデル化し，実際の環境に近い条件でシミュレーションを行うシミュレータである．LLS は主に受信部のベースバンド処理ブロックの特性評価を目的として，仕様検討から製品評価まで幅広く利用している．図 3 に LLS の概要を示す．LLS は送信部，伝搬路部，受信端末部の 3 つのブロックに分かれており，それぞれのブロックは実際の環境に近い条件になるように模擬している．

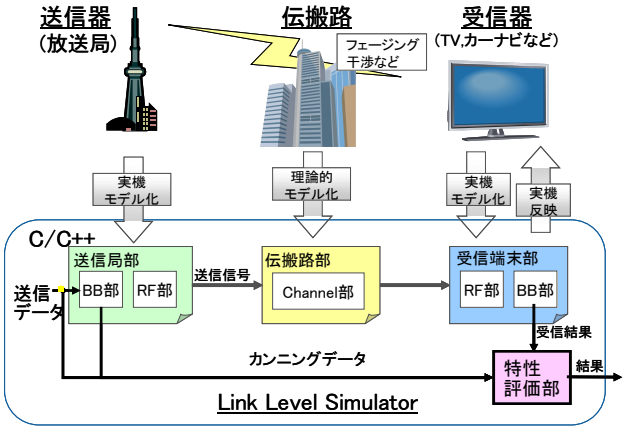


図 3. LLS の概要

3.2. 地上デジタル放送用受信技術の開発

ハードウェア開発では，システム全体および詳細な回路までの動作確認を行う必要があり，シミュレータは実機の動作を忠実に実現することが求められる．そのため受信端末部の各ブロックの動作がハードウェアの回路言語(RTL)と同じ動作となるように，プログラム化を行なっ

た．

一方，送信局部(放送局)のプログラム化については，下記の問題が生じた．

- プログラム化の参考になる実機を所持していないため送信器側の検証が容易でない．また，仕様書を解読しながらの作業であるため，シミュレータの開発期間が長期化する
- 図４のように送信局部の処理量はシステム全体の約15%を占めている．シミュレーション時間短縮のためには送信局側の処理軽減が必要不可欠
- 受信端末側のシミュレーションが主目的であるため，送信局側のプログラム化に利点を見出せない

以上の問題点を考慮し，市販されている ISDB-T 規格の波形発生器の出力波形を LLS の送信データとして代用した．図５に開発した LLS の構成図を示す．このシミュレータでは，シナリオファイルを用いて使用する入力ファイル名と伝搬路部/受信端末部の各種動作設定を行う仕組みにした．開発者はこのシナリオファイルの設定をシミュレーション条件に従い順次変更することで，各種特性評価を行うことができる．入力ファイルには送信信号のバイナリデータとビット誤りの評価などに用いるカンニングデータの２種類があり，LLS の送信データ読込部よりそれぞれのファイルのデータ読込みを行う．

以上の構成により，波形発生器の出力波形を，シミュレータの送信部分として代用した．市販の波形発生器を利用したため，多大な時間を要する動作検証を省略することができ，シミュレータ開発期間の短縮に繋がった．また，送信局側の処理はデータ読み込み時間だけになるため，シミュレーション時間の短縮にも繋がった．

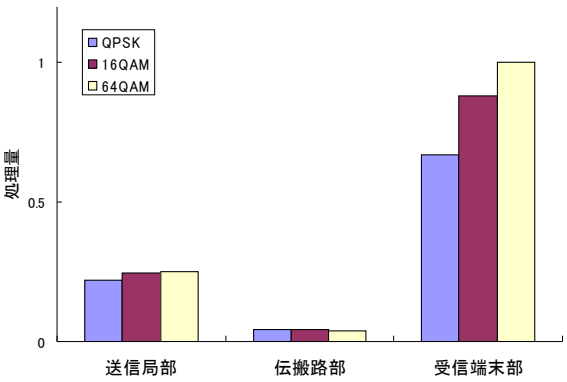


図 4．各ブロックの計算時間

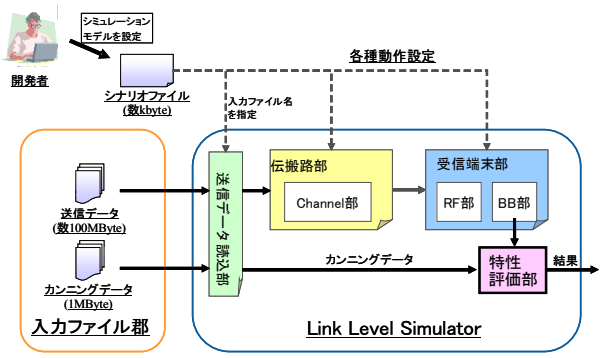


図 5．地上デジタル放送用 LLS ブロック図

3.3. CAD-Grid 適用時の問題点

前節で説明した送信データの代用を行なった場合に LLS へ入力する送信データをバイナリ化によりデータ圧縮を行なったとしても，そのファイルサイズが 100MByte を超過する．そのため，CAD-Grid へのジョブ投入時に，セントラルサーバ経由で各ノードに送信データを送るとセントラルサーバが過負荷状態となる可能性がある．また，セントラルサーバは各ノードの HDD の空き容量を確認することができないため，空き容量が少ないノードが存在した場合，送信データが転送できず，そのノードにおいて実行エラーが多発する．

以上の問題に対処するため，図６のようにジョブ実行前に各ノードの HDD に送信データを格納しておき，シミュレータからは実行しているノードの HDD の送信データを参照するように変更した．これによりセントラルサーバの過負荷問題は解決できたが，新たな問題も発生している．クラウド上のノード数が 50 を超えていたため，データ格納時に一部のノードに送信データ格納漏れのヒューマンエラーが時々発生した．そして，送信データ未格納のノードが存在したままジョブ実行を行うと，そのノードにおいて実行エラーが大量発生する問題が発生した また，一部のノードは事務用 PC と併用しているため，使用できる HDD 容量に制限があり，全ての送信データを格納して置くことができなかった．すなわち，ジョブ実行前に全ノードに送信データがあるかどうかを簡単に確認できるようにしておく必要があった．

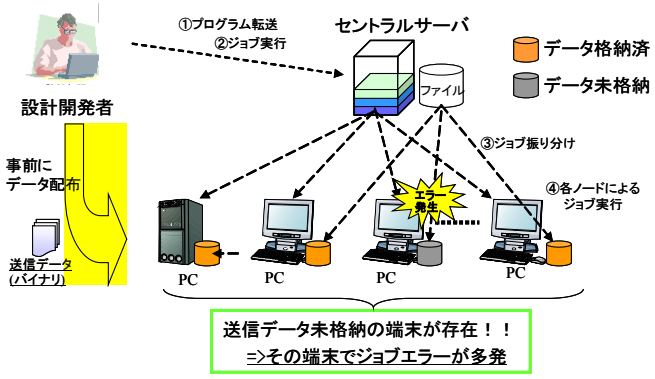


図６．送信データの未格納問題

4.　スクリプト活用による問題解決

3.3節で述べた送信データの格納漏れ問題を防止するためには，ジョブ実行前にノード一つ一つにアクセスし送信データの有無を確認する必要がある．しかしながらシミュレーションを行う実行ノード数が多いため，手動での確認には手間と時間を必要とする．我々はこの確認作業を簡略化するため，送信データの有無を自動的に確認するスクリプトを作成した．このスクリプトをジョブ実行前に実行することで送信データの確認漏れの防止を行った．

4.1. スクリプトによる確認作業の自動化

作成したスクリプトは，dirコマンドにより全ノードのノード情報を収集するプロセスと，perl言語の正規表現により，対象送信データが未格納のノードを抽出するプロセスにより構成される．

a)dirコマンドによるノード情報の収集

格納済み送信データ名とHDDの空き容量を収集するためdirコマンドを利用する．送信データを格納するディレクトリ名を[¥C$¥ISDB_data]とする場合，下記のようなコマンドを使用する．

```
dir ¥¥[IPアドレス]¥C$¥ISDB_data¥ >> out.txt
```

dirコマンドの出力結果例を図７に示す．dirコマンドの出力結果には実行ノードのIPアドレス，格納済みの送信データ名，Cドライブの空き領域などが記述される．このdirコマンドを全ノードに対して行い，その出力結果を一つのテキストファイルに纏める．

b)正規表現による未格納のノードの検出

dirコマンドの出力結果から，実行対象の送信データの有無を確認する手段として，perl言語で記述したスクリプトを用いる．このスクリプトでは，perl言語の正規表現の文字列検索/抽出機能を利用して，dirコマンドの出力結果から対象の送信データの検索を行う．その後，送信データが存在しないPCを抽出し，そのIPアドレスを出力する．これらの処理により，送信データが未格納のノードが抽出できる．また，各ノードの空き容量情報の確認を行い，IPアドレスと空き容量の出力も行う．

以上のようにスクリプト実行をジョブ実行前に行うことで，送信データの確認作業の自動化を実現できる．

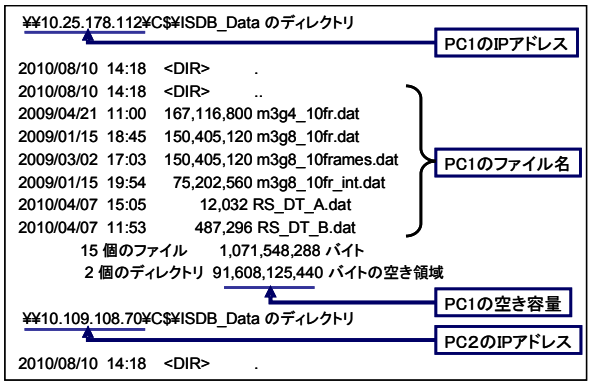


図７．dirコマンドの出力結果例

4.2. スクリプト適用の結果

前節で述べた，送信ファイルの確認作業の自動化により下記のような効果を得た．

- 送信データの格納作業/確認作業を手動で行った場合，全ノードに対し作業が完了するまでに1時間程度必要であった．その作業を自動化することにより20分程度に短縮することができた．この送信データの格納/確認作業は月平均で1.5回程度行うため，１月あたり約60分の時間短縮に繋がった．
- 全ノードのHDDの空き容量情報を用いることで，HDDの空き容量が少ないノードに対し，ジョブ投入禁止などの措置を行うことが可能になった．

5.　まとめ

地上デジタル放送用シミュレータの送信部を市販送信データで代用した背景について述べ，それにより発生したCAD-GRID適用時の問題点とその解決手段について説明した．特に，送信データ格納漏れ問題に関しては，その解決手段として実際に作成したスクリプトの詳細とその適用効果を明確にした．これらの対策により，シミュレータ開発およびジョブ実行の効率化を実現できた．

今回の事例では，送信データの格納漏れ問題に主眼を置いて検討を行ったが，ジョブ実行環境の効率化のためには，開発目的に合ったシミュレータを構築し，そのシミュレータに合った計算機環境を作っていくことが重要である．これからも積極的に活動を行い，開発効率ＵＰを目指していきたい．

参考文献

1)　社団法人電波産業界“地上デジタルテレビジョン放送の伝送方式”，ARIB STD-B31(May-2001)
2)　山下智規，中村武雄 etc.「グリッド環境「CAD-Grid」構築と移動通信システムシミュレーションへの適用」，

PSE Workshop in Sapporo(2003)
3) 松崎和浩, 山下智規 etc.「System CのMPI拡張と移動通信シミュレーションの適応」, PSE Workshop & Grid Seminar in Oita(2004)
4) 山本佳, 小橋博道 etc.「統計的設計支援システム・CyberGRIPを用いたグリッドシステムと移動通信システムシミュレーションへの適用」, PSE Workshop & Grid Seminar in Oita(2004)
5) 吉田大輔, 瓜生和也 etc.「最適化機能を強化したSmartGrid Systemの構築」, The 8th PSE Workshop & the3rdGrid Seminar’05
6) 梅田雅敬, 古舘英樹 etc.「利用者から見たCAD-Gridシステムの活用」, The 9th PSE Workshop’06
7) 小林崇春, 古舘英樹 etc.「移動通信シミュレータにおける実行時間最適化に関する検討」, The 10th PSE Workshop’07
8) 太田喜幸,岩松隆則「移動通信におけるシミュレーション技術の考察」, The 11th PSE Workshop’08
9) 岩松隆則, 古舘英樹 etc.「通信分野におけるハードウェア開発とシミュレーション」, The 12th PSE Workshop’09

第13 回 問題解決環境ワークショップ論文集

発行日 2010 年9 月15 日
編集発行 宇都宮大学 工学研究科 川田研究室
〒321-8585 栃木県宇都宮市陽東7-1-2
TEL(028) 689-6605