# 平成２１年度　成果報告書

「PaaS-CAE 基盤技術に関する研究開発」

目　次

## 1　研究開発課題の背景

(1) データ構造やプロセスの変化が激しく、またリモートアプリケーションサーバとのフレキシブルな連動が必要な R&D 系業務に効率化、高品質化、継承容易性を備えさせるためには、R&D 系業務のシステム化が必要である。しかしながら、従来のシステム化手法（Web プログラミングと DB 設計が必要な特注開発）では、R&D 系業務の非定常、非定型性のため十分なシステム化が容易ではない。

(2) 研究開発者には、できる限り研究開発に直接つながる業務、つまりクリエイティブな業務に集中できる環境を準備することが、会社として、また、研究開発部門として重要な課題である。しかしながら、実験計測器やコンピュータの性能が向上し、研究開発内の様々な雑務（ファイル探査やグラフ化処理等）が多く、研究開発者がクリエイティブな業務に割ける時間が少なくなってきている。

(3) いかなるクリエイティブな業務においても、適切な批判の目に晒されなければ、その正確性および洞察力に曇りが生じる。R&D 系業務に関してもその内容および進捗に関し、様々な（特に、経営層や研究開発責任者）人へ意見を求めたり、説明を行うことが重要である。そのためには、研究開発の進展状況等を研究者本人以外も簡単にトレースできる必要がある。しかしながら、現状では、研究開発が蛸壺型で、個人に委ねられており、適切な批判の目に晒された R&D 系業務にはなっていない。

(4) CAE、実験データ解析に上記要求を満たす機能を付加したシステムを社内で構築するには多くの問題が存在する。
- 閑散期にあわせると必要なときに使えない。
- 繁盛期にあわせて設備投資すると過剰投資になる。
- 新しいソフトを試したいのだが、気軽に試せない。
- 高度なシステム管理者が必要である。
- システム構築自体が特注開発であり、コストが膨大になるだけでなく、特注システムは納品をしてしまうと開発プロジェクトが終了してしまうため、製品システムに比べソフトウェア品質、保守性の点で遥かに劣ってしまう。つまり、システムを使う側としては、ハイリスク・ローリターンになってしまう。

(5) SaaS 化は、上記問題を解決しうるが、CAE は解析課題ごとに異なる自由度が要求され、GUI 画面や DB 化は、利用者ごと解析課題ごとに異なるものにしなければならない。したがって、従来の SaaS 化手法では、対応できなく、PaaS である必要がある。

(6) CAE などのシミュレーション系業務は、社外のシステム(Web の向こう側のシステム)でも良く、商用 PaaS サービスを活用することができる。しかし、実験データなどは、装置のある社内側にシステムを構築しなければならなく、商用 PaaS サービスで業務を行うことはできない。社外の商用 PaaS サービスと社内側システムの双方を FireWall を超えてシームレスに連携できる機構が実現されなければならない。

## 2　研究開発の全体計画

### 2－1　研究開発課題の概要

PaaS-CAE 基盤技術に関する研究開発

経理や営業等の事務業務系システムは、特注開発システムから SaaS、PaaS 等のクラウドコンピューティング化が進んでいる。一方、CAE や実験データ解析等の R&D 系業務は、取り扱うべき対象や処理プロセスの変化が激しく、特注開発でさえシステム化が容易ではない。RCM は、上記困難を克服し、R&D 系業務のシステム化を可能にした。

本開発は、RCM を拡張し、R&D 系業務の PaaS 化を可能にすること(FireWall 透過性、

信頼性、機密性、利便性）を目指すものである。ただし、R&D 業務では社外 PaaS 化になじまない実験装置等が存在するため、PaaS と社内 R&D システムとを連携する仕組みの開発を行う。これらの目標を達成するために具体的には、以下の 6 つの機構を開発し、実証システムの構築及び運用を行う。

(1) HTTPS ベースでリモートアプリケーションサーバの画面を高速に伝送する機構
(2) RCM システムの負荷分散、冗長機構
(3) データベースの高機密化（排他的記録）機構
(4) Workflow や高品位な UI を GUI で設定、変更を可能にする機構
(5) RCM システム間（WebServer-WebServer）の連携機構
(6) 既存（非 RCM）社内 R&D システムとの連携機構
(7) 実証システムの構築と運用

(1) HTTPS ベースでリモートアプリケーションサーバの画面を高速に伝送する機構

現状の RCM システムソフトウェアは、Workflow 機能によりリモートのアプリケーションサーバの画面をクライアントに表示させ、インタラクティブな操作を可能にすることを実現できている。この方式では、SSH トンネリング機能及び VNC サーバ機能を利用しているため、クライアントとリモートアプリケーションサーバ間が直接 SSH で接続できる必要がある。

社内システムの場合は、上記接続方式で大きな問題は生じないが、SaaS や PaaS 環境のように社外にシステムがある場合は、上記接続方式は、Firewall 等に阻まれることになる。したがって、リモートアプリケーションサーバは、コントロールサーバに SSH のみで接続し、クライアントはシステムの Web サーバに HTTPS のみで接続することができる機構を開発する必要がある。

本開発は、RCM からリモートアプリケーションサーバのインタラクティブアプリケーションを利用する場合における FW 透過性を獲得することを目指すものである。

(2) RCM システムの負荷分散、冗長機構

現状の RCM は、リモートアプリケーションサーバの負荷分散、冗長機構を有しているが、システムコアである WebServer、ControlSever、DataBaseServer は、それぞれ、1 システム 1 サーバとなっており、負荷分散、冗長機構を持っていない。既存の Web システムの負荷分散や冗長化関連製品は、セッション情報の冗長化しか行わないが、RCM は、Workflow 機能等の複雑な状態遷移処理機構により、連続的な処理やユーザインタフェース構築を自動的に行っているためセッション情報の冗長化だけでは、システム全体の冗長化を達成できない。したがって、既存の負荷分散、冗長化製品では RCM の連続稼働性を保証できない。

本開発は、RCM を拡張し、WebServer、ControlSever、DataBaseServer の負荷分散、冗長機構を付加し、商用 PaaS サービスに耐えうる信頼性を獲得することを目指すものである。

冗長化機能を考えると、データベース部は、永続的なデータを格納しているため完全な多重化システム（全ての状態が同一）が必要となるが、Web、Control 部は、Workflow 単位での永続性のみが必要であることから、Workflow 単位での多重化で十分である。したがって、本機能の負荷軽減のため Master-Slave ペアを確立し、どちらか一方に障害があった場合は、自らがマスターになり動的に Slave を割り当てる動的 Master-Slave ペア方式を採用する。

(3) データベースの高機密化（排他的記録）機構

現状の RCM は、データベース部の並列化が可能であるが、この並列化はあくまで高速化としての側面である。しかしながら、データの機密性を重んじるシステムの場合は、データの機密度ごとに異なるハードウェアを準備し、そこに格納することが条件づけられる。例えば、PaaS サービスの場合、より高い価格を支払うことである情報部分のデータのみを別ハードウェアへ割り当てることを要求する場合も考えられる。事務業務系 PaaS サービスであっても、この条件をクリアできないがために PaaS サービスが採用されない場合も多い。

本開発は、データベースの排他的記録機構を開発し、任意のデータおよびメタデータを別ハードウェアに格納することを可能にする機能の獲得を目指すものである。

(4) Workflow や高品位な UI を GUI で設定、変更を可能にする機構

現状の RCM システムソフトウェアは、XML-Workflow 内に変数化の設定や UI の空間配置等のデザインを XML で記述している。入力系 UI を簡易 UI と呼び、出力系 UI を XML-Viewer と呼んでいる。これら UI は、汎用的な XML を表示するための強力なエンジンではあるが、特注開発や商用アプリケーションと比べると、装飾性が非力であり、画面表現力が乏しい。RCM では、入力系 UI、出力系 UI とも外部で記述した HTML と差し替えることができるように実装されているが、PaaS 化を考えると GUI で UI のデザインを設計し、変更できる程度にまで利便性を向上させる必要がある。また、XML-Workflow 全体に関しても、現状は、特別なツールがないため通常の XML エディターで入力・編集を行っているが、これも GUI インタフェースで入力・編集できるようにする必要がある。

本開発は、GUI で XML-Workflow および入力系 UI、出力系 UI のデザインを設計し、変更できる機能の獲得を目指すものである。

(5) RCM システム間（WebServer-WebServer）の連携機構

現状の RCM は、1 システムで完全に閉じた形となっており、システム同士は連携できない。商用 PaaS サービスと社内システムのように 2 つのシステム配下のリモートアプリケーションサーバの双方を使った処理プロセスは、研究開発者が手動で 2 つのシステムを切り替えながら実行するしかない。つまり、双方跨って利用するサービスなど強く連携したサービス構築は不可能である。

したがって、装置のある社内側に存在する実験データは、社内 R&D システムで処理し、シミュレーションは、商用 PaaS サービスを使いそれらを比較するような系統的な処理を実現する場合、社内 R&D システムと商用 PaaS サービスの双方のシステムがシームレスに連携できる必要がある。

本開発は、WebServer-WebServer の連携機構を開発し、RCM を拡張し、複数の（RCM を使った）システムをシームレスに連携することができる機能の獲得を目指すものである。

(6) 既存（非 RCM）社内 R&D システムとの連携機構

現状の RCM システムソフトウェアは、Workflow 機能に含まれている SSH 接続機能により、リモートアプリケーションサーバへのコマンド実行、つまりアプリケーション利用を可能にしている。その他、REST によるコマンド発行も可能である。しかしながら、独自に作られた既存システム（ほとんどのシステムは、特注開発

であり、仕様で決められた機能を実現するための独自実装である）と連携することは、困難である。

本開発は、既存システムとの連携を行うために、任意の通信プロトコルを受信し、応答するためのブリッジモジュール機能の獲得を目指すものである。また、拡張性の高い汎用通信プロトコルである SOAP や REST 等は、標準装備する予定である。

(7) 実証システムの構築と運用

本研究開発で作成した機構をインプリメントした RCM を使って、実証システムを構築し、CAE における仮想商用 PaaS サービス＋社内 R&D システムの連携システムが有効に機能することを確かめる。また、実際のシステム運用によって運用モデルを明確化するとともに、問題点等を改善し、より使いやすいシステムへと改良を行う。

## ２－２　研究開発の最終目標（平成 23 年 10 月末）

(1) HTTPS ベースでリモートアプリケーションサーバの画面を高速に伝送する機構

1.1 クライアントと別ネットワークにあるリモートアプリケーションサーバに関して、クライアントはシステムの RCM-Web サーバに HTTPS のみで接続するものとし、リモートアプリケーションサーバは、RCM コントロールサーバに SSH のみで接続することで、クライアントからリモートアプリケーションサーバのインタラクティブアプリケーションを操作できること。

1.2 現状の接続方式（クライアントと別ネットワークにあるリモートアプリケーションサーバが direct に SSH でつながっている）に比べ、中継が 2 段になり、伝送路が 3 倍になるが、現状の 1/5 以内のインタラクティブ性能（FPS）を確保する。

(2) RCM システムの負荷分散、冗長機構

2.1 RCM-Web サーバの分散化ができ、ロードバランス機能により負荷を分散できること。

2.2 RCM-Web サーバの一部に障害が発生した場合は、他の RCM-Web サーバが処理を継続できること。

2.3 RCM- Control サーバの分散化ができ、ロードバランス機能により負荷を分散できること。

2.4 RCM- Control サーバの一部に障害が発生した場合は、他の RCM- Control サーバが処理を継続できること。

2.5 RCM-DB サーバの分散化ができ、ロードバランス機能により負荷を分散できること。

2.6 RCM-DB サーバの一部に障害が発生した場合は、他の RCM- DB サーバが処理を継続できること。

(3) データベースの高機密化（排他的記録）機構

3.1 任意のデータおよびメタデータを別ハードウェアに格納するための DB リクエストを開発し、それ以外のデータとハードウェア的に分離した管理が可能であること。

(4)Workflow や高品位な UI を GUI で設定、変更を可能にする機構

4.1 XML-Workflow（入力系 UI、出力系 UI を含む）を GUI 画面で設定できること。

4.2 XML-Workflow 設定 GUI 画面では、XML のタグ名入力は不要にすること。

4.3 XML-Workflow 設定 GUI 画面では、固定的な複数選択しは、ドロップダウンで選べるようにすること。

4.4 XML-Workflow 設定 GUI 画面では、数値、日付など明らかなフォーマット指定がある場合は、入力時にそのフォーマットを誘導するとともにチェック機構を有すること。

4.5 XML-Workflow 設定 GUI 画面では、job 間の相関性（参照、重複不可等）を意識させながら入力できること。

4.6 入力系 UI、出力系 UI 設定は、レイアウト設定が簡単なように HomePage 作成ツールのようなレイアウト画面で設定できること。

(5)RCM システム間（WebServer-WebServer）の連携機構

5.1 RCM の Workflow 記述において、異なる RCM システムを跨いだ記述が可能であり、Workflow 内で異なる RCM システムで実行される job（Workflow 内の 1 つの作業単位で最小記述レベルでもある）は、RCM システムの WebServer 間連携機構より、他方の RCM システムに処理を依頼し、その戻値を受け取ることができること。

5.2 RCM の Workflow 記述において、1 つの job 内で複数の異なるサーバにアクセスを行うものに関しても、異なる RCM システムを跨いだ記述が可能であり、当該 job 内で異なる RCM システム支配下のサーバを利用する場合、RCM システムの WebServer 間連携機構により、他方の RCM システムと連携して 1 つの job 機能を果たすこと。

(6)既存（非 RCM）社内 R&D システムとの連携機構

6.1 任意の通信プロトコルを受信し、応答するためのブリッジモジュール機構を開発し、既存システムの通信プロトコルに合わせ専用ブリッジモジュールを開発でき、Controller に簡単に（システム全体のリコンパイルなしに）追加できること。

6.2 ブリッジモジュールにおいて SOAP プロトコルを使う場合、ESB を使った通信をサポートできるようにすること。

6.3 ブリッジモジュールの稼働・停止状態を RCM システムのメンテナンスモードから監視、制御できること。

(7)実証システムの構築と運用

7.1 2 つ以上の RCM システムを別ネットワークで構築、運用し、(1)～(6)の機能が正しく動作するかを 1 ヶ月以上検証すること。

7.2 (1)～(6)の機能の性能を評価し、性能、機能面で利用上問題がある部分に関して、ボトルネックポイントを同定し、改善プランを策定すること。

7.3 7.1 の検証時に運用モデルを明確化し、運用マニュアル、運用における注意点をドキュメントにまとめること。また、そのマニュアルに従って、運用を実際に行って、問題点がないかを確認すること。

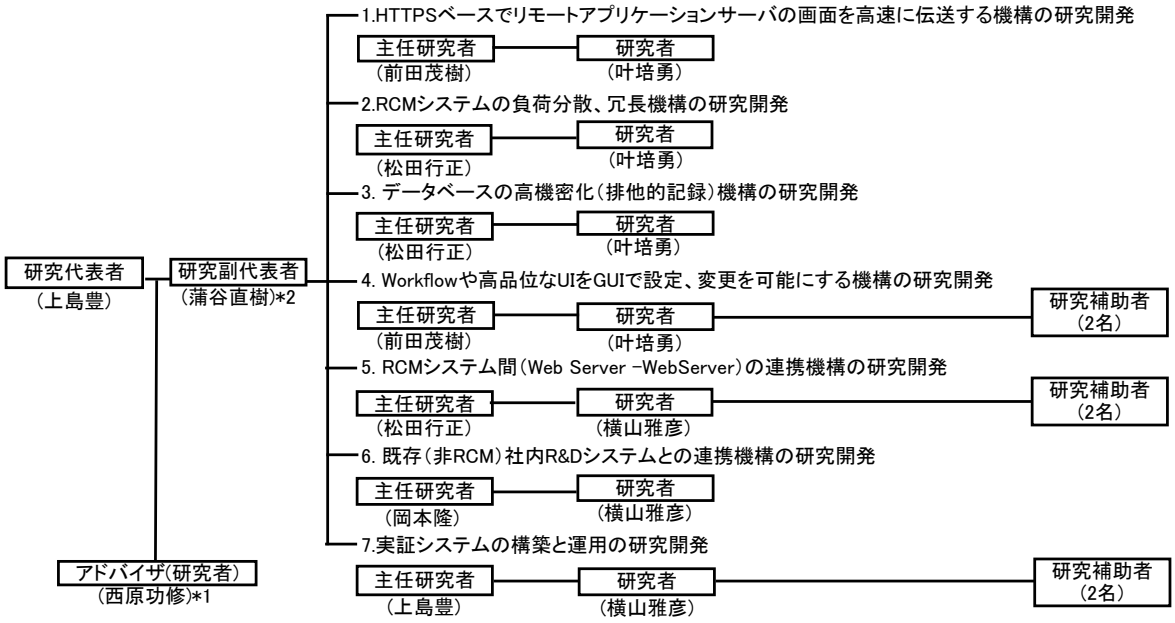
## ２－３　研究開発の年度別計画

金額は非公表

| 研究開発項目 | 21 年度 | 22 年度 | 23 年度 | 計 | 備　考 |
|---|---|---|---|---|---|
| PaaS-CAE 基盤技術に関する研究開発<br>(1)HTTPS ベースでリモートアプリケーションサーバの画面を高速に伝送する機構の研究開発 | － | － | － | － | |
| (2)RCM システムの負荷分散、冗長機構の研究開発 | － | － | － | － | |
| (3)データベースの高機密化（使用領域指定）機構の研究開発 | － | － | － | － | |
| (4)Workflow や高品位な UI を GUI で設定、変更を可能にする機構の研究開発 | － | － | － | － | |
| (5)RCM システム間(WebServer-WebServer)連携機構の研究開発 | － | － | － | － | |
| (6)既存システムとの連携機構の研究開発 | － | － | － | － | |
| (7)実証システムの構築と運用の研究開発 | － | － | － | － | |
| 間接経費 | － | － | － | － | |
| 合　　計 | － | － | － | － | |

# 3　研究開発体制

## 3－1　研究開発実施体制




*1) 1～7の研究課題全般にわたり、横断的な観点でシステム全体が最良な進展をするように、研究代表者、各課題の主担当者(主任研究者)と合議を行い、研究方針の決定を行う。
*2) 1～7の研究課題全般にわたり、研究代表者である上島の補佐的役割を果たす。

## ４　研究開発実施状況

### ４－１　HTTPS ベースでリモートアプリケーションサーバの画面を高速に伝送する機構の研究開発

#### ４－１－１　研究開発内容

(1) 現状の SSH トンネリングベースの「リモートアプリケーションサーバ画面の伝送」機構を HTTPS ベースによる伝送機構に切り替える試作開発を行なった。

(2) 試作開発と並行して、リモートアプリケーションサーバ上で動作する評価対象アプリケーションが評価版などで無償貸与をしてもらえるか、リースが必要か、購入が必要かなどを調査し、選定および導入を行なった。

(3) 試作開発をベースに Firewall 透過性や通信性能の基礎性能を評価した。

#### ４－１－２　実施状況

(1) SSH および HTTPS の性能測定を実施し、通常の HTTPS 転送ではアプリケーションにより性能が確保できないことを確認した。

実施内容 1

SSH プロトコルを使って 3D アプリケーションを VNC でリモートマシンから操作する。

パケット遅延時間とパケットロス率をネットワーク状態パラメーターとして、体感上の操作性について、評価を行った。

（ 注 ） 下表の RTT はパケット遅延時間、PLR はパケットロス率を示す。
RTT および PLR は、ネットワーク負荷装置を試験環境内に設置し、意図的に RTT や PLR を発生させた。

各負荷設定時に、VNC を SSH でトンネリングした場合のアプリケーションの応答性を 5 段階評価し、表 1 に示す。

一般的なインターネットの場合、RTT は 20ms 以下、PLR 0%を想定でき、その範囲は表 1 の 背景色が水色のセルになる。
インターネットを想定した負荷では、評価値は 4 または 3 となり、VNC を SSH でトンネリングした場合の操作性は、インターネットを想定すると、実施可能であることが確認できた。

表 1　VNC over SSH での、PLR, RTT と操作性の評価

| 設定 RTT(ms) | PLR 0% 評価 | PLR 2% 評価 | PLR 4% 評価 |
|---|---|---|---|
| 1 未満 | 4 | 3 | 2 |
| 10 | 4 | 2 | 2 |
| 20 | 3 | 2 | 1 |
| 50 | 2 | 1 | 1 |
| 100 | 2 | 1 | 1 |
| 200 | 1 | 1 | 1 |

段階別評価
１：数十秒以上応答が帰ってこない。
２：数秒以上応答が帰ってこない。動的画像では、利用が不可能。静的な画像であれば、利用可能。
３：0.1 数秒毎にコマ落ちがあるが利用できないレベルではない。（動的画像の限界）
４：ほとんどコマ落ちせず処理できている
５：ローカルアプリケーションと同等である

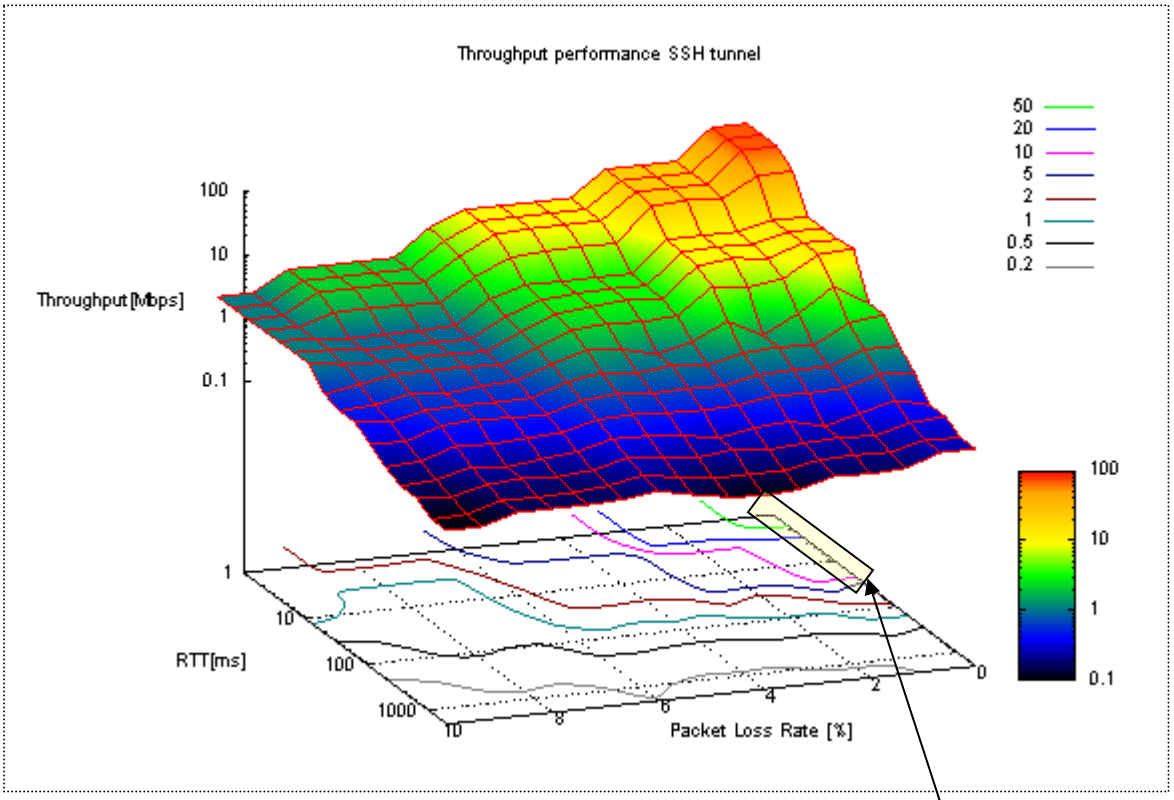



図 1　RTT, PLR の値と SSH トンネルのスループット

次に、表 1 で実際に「4:ほぼ問題なく使用できる」または
「多少難はあるが使用できる」は、 図 1 を参照すると、SSH トンネル時に
5Mbps 以上のエリアを示すことがわかる。
(注) 図 1 の見方
RTT20ms 以下かつ PLR 0%のエリアは、図 1 の薄い黄色のエリアで、
その設定値における SSH トンネル時のスループットは Z 軸を見ます。
すると、Z 軸の等高線で 5Mbps から 100Mpbs を示している。

従って、3D アプリケーションの画面転送では、少なくとも 5Mbps を越える
実行スループットが必要であることがわかった。

実施内容 2

SSH の 3 連結時における実行スループットの調査。
RCM では、VNCViewer と VNCServer の間に、RCM-Web サーバと RCM-CTL サーバ
の 2 台を経由するため、SSH を 3 連結して VNC を画面転送することになる。
(図 2 の RCM の接続構成図を参照)

本調査では、SSH の 3 連結時におけるスループット計測を行った。

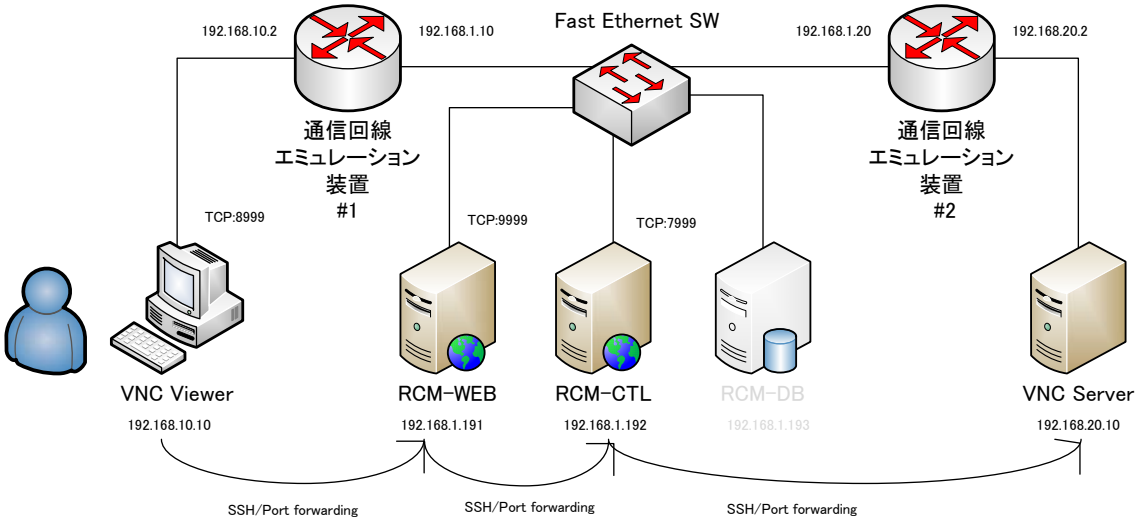



図 2　RCM の接続構成図（SSH の 3 連結時）

表 2　SSH（3 連結）による VNC 転送の実効スループット計測結果

| 設定 RTT (単位:ms) | スループット PLR 0% （単位：Mbps） | スループット PLR 2% （単位：Mbps） | スループット PLR 4% （単位：Mbps） | スループット PLR 8% （単位：Mbps） | スループット PLR 10% （単位：Mbps） |
|---|---|---|---|---|---|
| 1 未満 | 50.3 | 21.7 | 14.4 | 2.83 | 0.780 |
| 10 | 15.2 | 5.73 | 2.86 | 0.940 | 0.780 |
| 20 | 11.1 | 3.09 | 2.83 | 1.04 | 0.620 |
| 50 | 2.88 | 1.29 | 1.09 | 0.680 | 0.470 |
| 100 | 1.40 | 0.880 | 0.620 | 0.410 | 0.420 |
| 200 | 0.780 | 0.570 | 0.410 | 0.260 | 0.320 |
| 500 | 0.370 | 0.320 | 0.260 | 0.220 | 0.160 |
| 1000 | 0.260 | 0.210 | 0.210 | 0.110 | 0.160 |
| 2000 | 0.160 | 0.110 | 0.160 | 0.110 | 0.160 |

この結果から、SSH の 3 連結時、RTT 20ms 以下、PLR 0%における TCP の実行スループットは、3D アプリケーションの「全く問題なく使用できる」を満たす 10Mbps を確保できていることが分かった。

(2) 評価対象アプリケーションの選定を終了し、各ベンダーにアプリケーションの無償貸与の可否などを確認した。

表 3　評価対象アプリケーション無償貸与の可否等の確認状況

| アプリケーション | 動作環境 | 許可依頼先 | 状況 |
|---|---|---|---|
| Qm（動磁場解析コード） | Windows XP | 株式会社シフトロック | 許可済み |
| AVS/Express | Windows XP | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| AVS/Express | Redhat5.3 | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| Ensight | Redhat5.3 | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| RealIntage | Windows XP | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| HexaGrid | Redhat5.3 | 株式会社計算力学研究センター<br>(独)宇宙航空研究開発機構 | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| ADVC | Redhat5.3 | 株式会社アライドエンジニアリング | これからアプローチ |
| 依頼先が販売可能なソフトウェア全般 | --- | 株式会社アドバンスソフト | 依頼済み。回答待ち |
| 依頼先が販売可能なソフトウェア全般 | --- | 株式会社計算力学研究センター | 依頼済み。回答待ち |
| 依頼先が販売可能なソフトウェア全般 | --- | 株式会社ソフトウェアクレイドル | まだ、アプローチしていない。 |
| ANSYS | Redhat5.3 | サイバーネットシステム株式会社 | ライセンスの問題で最終ユーザしか購入ができないので、実証試験には利用できない可能性が高い・ただし、１ヶ月のデモライセンスで対応可能か協議中。 |

(3) (1) の課題解決手段を組み込んだ性能評価を実施した。

実施内容３

次に、本課題の核心である HTTPS による VNC 転送の実行スループット計測について報告する。

Firewall 透過性を確保するためには、VNCViewer と RCM-Web サーバ間は、SSH ではなく、HTTPS の接続が要求される。
また、RCM-Web サーバと RCM-CTL サーバ間も、セキュリティ的には HTTP 接続が望ましい。
( 図 3 を参照 )

計測の結果、最大スループットは 38.9Mbps であるが、RTT=20ms, PLR=0%では 16.8Mbps となり、通信回線ロスに対しては SSH 3 連結よりもロバストな性能を示すことがわかった。

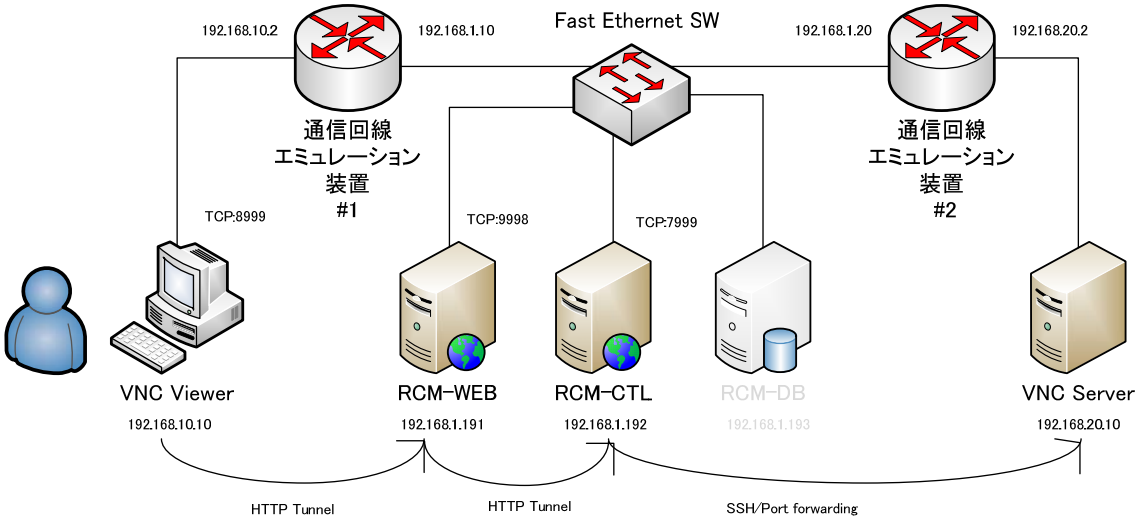



図 3　RCM の接続構成図 (HTTPS 接続時)

表 4　HTTPS による VNC 転送の実効スループット計測結果

| 設定 RTT (単位:ms) | スループット PLR 0% (単位：Mbps) | スループット PLR 2% (単位：Mbps) | スループット PLR 4% (単位：Mbps) | スループット PLR 8% (単位：Mbps) | スループット PLR 10% (単位：Mbps) |
|---|---|---|---|---|---|
| 1 未満 | 38.9 | 14.4 | 7.12 | 2.85 | 0.834 |
| 10 | 23.1 | 4.51 | 1.88 | 1.01 | 0.656 |
| 20 | 16.8 | 2.84 | 1.92 | 0.810 | 0.605 |
| 50 | 8.00 | 1.20 | 0.838 | 0.471 | 0.397 |
| 100 | 4.36 | 0.692 | 0.472 | 0.306 | 0.261 |
| 200 | 2.11 | 0.404 | 0.267 | 0.178 | 0.131 |
| 500 | 0.846 | 0.143 | 0.102 | ---- | ---- |
| 1000 | 0.352 | ---- | ---- | ---- | ---- |
| 2000 | 0.180 | ---- | ---- | ---- | ---- |

実施内容４

参考までに RCM-Web サーバと RCM-CTL サーバ間を HTTP から SSH に戻して比較計測を行った。

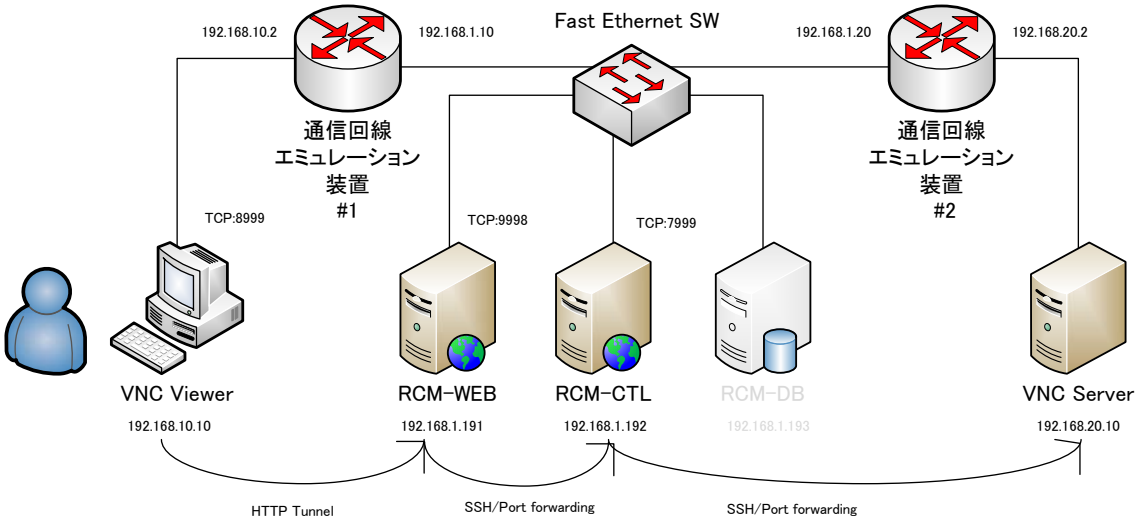



図 4　RCM-Web - RCM-CTL 間を SSH に戻した場合の RCM の接続構成図

表 5　RCM-Web - RCM-CTL 間を SSH に戻した場合の VNC 転送実効スループット計測結果

| 設定 RTT (単位:ms) | スループット PLR 0% (単位：Mbps) | スループット PLR 2% (単位：Mbps) | スループット PLR 4% (単位：Mbps) | スループット PLR 8% (単位：Mbps) | スループット PLR 10% (単位：Mbps) |
|---|---|---|---|---|---|
| 1 未満 | 35.5 | 12.9 | 6.95 | 1.66 | 1.27 |
| 10 | 21.4 | 3.99 | 2.69 | 1.36 | 0.922 |
| 20 | 16.5 | 2.80 | 1.73 | 0.733 | 0.456 |
| 50 | 8.06 | 1.32 | 0.838 | 0.566 | 0.356 |
| 100 | 4.33 | 0.782 | 0.486 | 0.315 | 0.227 |
| 200 | 2.24 | 0.347 | 0.245 | 0.147 | 0.146 |
| 500 | 0.796 | 0.166 | 0.102 | ---- | ---- |
| 1000 | 0.381 | ---- | ---- | ---- | ---- |
| 2000 | 0.177 | ---- | ---- | ---- | ---- |

この接続状態でも 調査結果 3 とほぼ同等のスループットを示す。

結論

SSH で VNC を転送する現行の RCM の方式に対して、FireWall 透過性を付与するために、HTTPS 通信路で VNC を転送する本課題については、実現可能性を示すことができた。

しかしながら、実際に RCM システム上で、他の負荷の高い処理が稼動している場合などは、更なる調査を実施する必要がある。

### ４－１－３　達成状況及び今後の課題

(1) 平成 21 年度は、現状の SSH トンネリングベースと試作した HTTPS ベースの簡易的な基礎性能評価を実施した。これに引き続き平成 22 年度は、性能評価および試作機構の改善を進めながら、RCM を HTTPS ベースによる伝送機構に切り替える。具体的には、クライアントからリモートアプリケーションサーバのインタラクティブアプリケーションの操作を可能とする。

(2) 現状の接続方式（クライアントと別ネットワークにあるリモートアプリケーションサーバが direct に SSH でつながっている）に比べ、中継が 2 段になり、伝送路が 3 倍になるが、現状の 1/5 以内のインタラクティブ性能（FPS）を確保する。

## ４－２　RCM システムの負荷分散、冗長機構の研究開発

### ４－２－１　研究開発内容

(1) 最新のアプリケーションサーバ製品で負荷分散、冗長機構がどこまでどのような方式で実装されているのかを実際の動作テスト、性能測定を含め調査した。

(2) 既存アプリケーションサーバ製品で負荷分散、冗長機構において実装されていないことが確実な一時ファイルに関する取り扱い機構の案を複数検討し、将来性、性能面を考慮して、開発機構の優先順位を評価した。最も優先順位の高い方式に致命的問題が出た場合に迅速に次点案に切り替えられるように開発手順を明確化した。

(3) (2) の一時ファイルに関する取り扱い機構を試作した。
(4) 試作機構の基礎性能を評価した。

## ４－２－２　実施状況

(1) 代表的なアプリケーションサーバ 4 種類について機能比較を実施した。

実施内容 1

下記のアプリケーションサーバ 4 種類が提供する冗長化機能の機能比較を行った。

- Jakarta Tomcat6
- JBoss EAP5
- WebLogic 11gR1
- WebSphere v7.0

表 6　アプリケーションサーバ機能比較①

| | AP サーバ | Tomcat 6 | | JBoss EAP 5 | | WebLogic 11gR1 | | WebSphere v7.0 | |
|---|---|---|---|---|---|---|---|---|---|
| | クラスタ方式 | All-to-All | Pri-Sec | All-to-All | Pri-Sec | All-toAll | Pri-Sec | All-toAll | Pri-Sec |
| オブジェクト | Collection | ○ | ○ | ○ | ○ | × | ○ | × | ○ |
| | Map | ○ | ○ | ○ | ○ | × | ○ | × | ○ |
| | File | × | × | × | × | × | × | × | × |
| シリアライズ | シリアライズ化 | ○ | ○ | ○ | ○ | × | ○ | × | ○ |
| | 非シリアライズ化 | × | × | × | × | × | × | × | × |

○は、機能が存在し、×は機能が存在しないことを示している。

4 種類のアプリケーションサーバとも、当初の想定通り、一時ファイルの冗長化機構を有していないことを、実機動作テストも含め、確認した。
また、商用アプリケーションサーバである、WebLogic と WebSphere は冗長クラスタ構成において、All-to-All のクラスタをサポートしていないことが分かった。

非シリアライズ化オブジェクトをクラスタ転送した場合の挙動(下表の最右列)については、JBoss の挙動が、他の 3 つと異なることも実機で確認した。

表 7　アプリケーションサーバ機能比較②

| No | AP サーバ | クラスタ方式 | Collection・Map に下記の Object が含まれる場合 | | | | |
|---|---|---|---|---|---|---|---|
| | | | Serializable | int[] | Null | Serializable[] | 非 Serializable |
| 1 | Tomcat | All-to-All | ○ | ○ | ○ | ○ | C |
| 2 | | Primary-Secondary | ○ | ○ | ○ | ○ | C |
| 3 | JBoss | All-to-All | ○ | ○ | ○ | ○ | D |
| 4 | | Primary-Secondary | ○ | ○ | ○ | ○ | D |
| 5 | WebLogic | Primary-Secondary | ○ | ○ | ○ | ○ | C |
| 6 | WebSphere | Primary-Secondary | ○ | ○ | ○ | ○ | C |

○： 受信サーバで Collection から値を取り出し、value が一致している
B ： 受信サーバで Collection を取り出せるが、value は Null になっている
C ： 送信サーバでエラーが発生するがセッションレプリケーション機能は継続し、
受信サーバではセッションは Null にならないが、Collection 自体が Null になっている
D ： 送信サーバで送信エラーになり、受信サーバで HTTP セッション自体が Null になっている

実施内容 2

4 種類のアプリケーションサーバについて、クラスタ間の冗長化機構の性能評価を行った。

表 8 から表 11 の説明

(1) 表中の単位のない数値は、実行に要した時間(ms)を示す。

(2) データ容量は、1 台のアプリケーションサーバに登録したデータが残りのデータサーバに複製されるまでの、登録データ容量と、所要時間の対応を示す。

(3) データ登録回数は、8 バイトのデータを 1 台のアプリケーションサーバに、1 秒間にデータ登録回数登録した際の、1 登録あたりのデータ複製完了時間を示す。

表 8　アプリケーションサーバ機能比較③

| | アプリケーションサーバ | Tomcat | | | | | |
|---|---|---|---|---|---|---|---|
| | クラスタ方式 | All-to-All | | | Pri-Sec | | |
| | クラスタ構成台数 | 2 台 | 3 台 | 4 台 | 2 台 | 3 台 | 4 台 |
| データ容量 | 1byte | 3 | 4 | 4 | 4 | 4 | 3 |
| | 1MB | 99 | 191 | 269 | 99 | 102 | 127 |
| | 10MB | 1316 | 2013 | 2826 | 1353 | 1188 | 1071 |
| | 100MB | 35934 | 61663 | 90163 | 36310 | 36044 | 35823 |
| データ登録回数 | 1 回 | 4 | 3 | 3 | 3 | 4 | 3 |
| | 10 回 | 4 | 5 | 4 | 3 | 3 | 3 |
| | 100 回 | 5 | 5 | 7 | 3 | 4 | 4 |
| | 1000 回 | 17 | 30 | 50 | 9 | 11 | 35 |

表 9　アプリケーションサーバ機能比較④

| | アプリケーションサーバ | JBoss | | | | | |
|---|---|---|---|---|---|---|---|
| | クラスタ方式 | All-to-All | | | Pri-Sec | | |
| | クラスタ構成台数 | 2 台 | 3 台 | 4 台 | 2 台 | 3 台 | 4 台 |
| データ容量 | 1byte | 7 | 8 | 8 | 7 | 7 | 8 |
| | 1MB | 105 | 108 | 158 | 105 | 126 | 108 |
| | 10MB | 1164 | 1267 | 1415 | 1548 | 1245 | 1306 |
| | 100MB | 21341 | 20298 | 21664 | 17872 | 16117 | 18428 |
| データ登録回数 | 1 回 | 7 | 8 | 8 | 7 | 7 | 8 |
| | 10 回 | 8 | 9 | 9 | 7 | 7 | 8 |
| | 100 回 | 9 | 10 | 11 | 8 | 11 | 11 |
| | 1000 回 | 25 | 29 | 36 | 23 | 29 | 51 |

表 10　アプリケーションサーバ機能比較⑤

| | アプリケーションサーバ | WebLogic | | |
|---|---|---|---|---|
| | クラスタ方式 | Pri-Sec | | |
| | クラスタ構成台数 | 2 台 | 3 台 | 4 台 |
| データ容量 | 1byte | 5 | 10 | 10 |
| | 1MB | 100 | 115 | 115 |
| | 10MB | 1014 | 1014 | 1031 |
| | 100MB | 16449 | 16402 | 17028 |
| データ登録回数 | 1 回 | 9 | 8 | 9 |
| | 10 回 | 8 | 9 | 10 |
| | 100 回 | 9 | 10 | 14 |
| | 1000 回 | 14 | 14 | 19 |

表 11　アプリケーションサーバ機能比較⑥

| | アプリケーションサーバ | WebSphere | | |
|---|---|---|---|---|
| | クラスタ方式 | Pri-Sec | | |
| | クラスタ構成台数 | 2 台 | 3 台 | 4 台 |
| データ容量 | 1byte | 9 | 22 | 19 |
| | 1MB | 14 | 24 | 24 |
| | 10MB | 73 | 52 | 79 |
| | 100MB | 2806 | 2079 | 1601 |
| データ登録回数 | 1 回 | 21 | 21 | 19 |
| | 10 回 | 19 | 22 | 18 |
| | 100 回 | 21 | 30 | 19 |
| | 1000 回 | 36 | 40 | 40 |

これらの測定から、データ容量 100MB のオブジェクトをクラスタ転送するには WebSphere の性能が 他の 3 種の 10 倍近い速度を計上した。

また、データ登録回数については、大きな差異は検出できなかった。

(2) および (3)　一時ファイルの冗長化機構を 3 種類検討し、実際に試作を行い、各方式の機能面での動作を確認した。

実施内容

File のレプリケーション機構を、下記の3 方式で試作を行った。

(試作1) Java のFile クラスを拡張した FileEx クラスを作成し、Serializa 化の読込/書込メソッド( readObject, writeObject ) をオーバライドして、参照先File を対象として 読込/書込を行う。

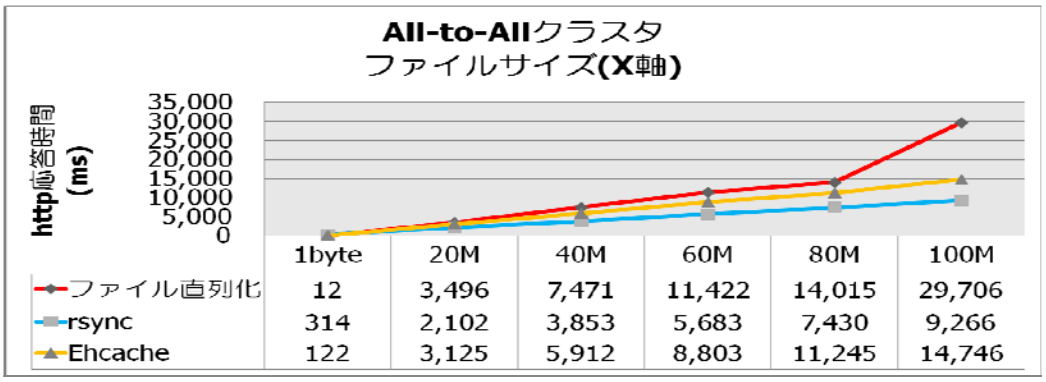
これにより、通常のアプリケーションサーバの クラスタリング機構でFile の実体そのもののレプリケーションを実現する。

(試作2) Linux rsync 機構を用いて、ディスクイメージの複写管理を行う。

(試作 3)分散キャッシュフレームワーク( Ehcache )により、キャッシュを用いてオブジェクトの複写管理を行う。

上記 3 方式のいずれも、機能的には一時ファイルのレプリケーション機能を実現することが実証できた。
( 表 12 に、3 種類の試作が、1byte から 100Mbyte までのファイルをレプリケーションした実績を示す。)

表 12　試作のレプリケーション実績




| | 1byte | 20M | 40M | 60M | 80M | 100M |
|---|---|---|---|---|---|---|
| ファイル直列化 | 12 | 3,496 | 7,471 | 11,422 | 14,015 | 29,706 |
| rsync | 314 | 2,102 | 3,853 | 5,683 | 7,430 | 9,266 |
| Ehcache | 122 | 3,125 | 5,912 | 8,803 | 11,245 | 14,746 |

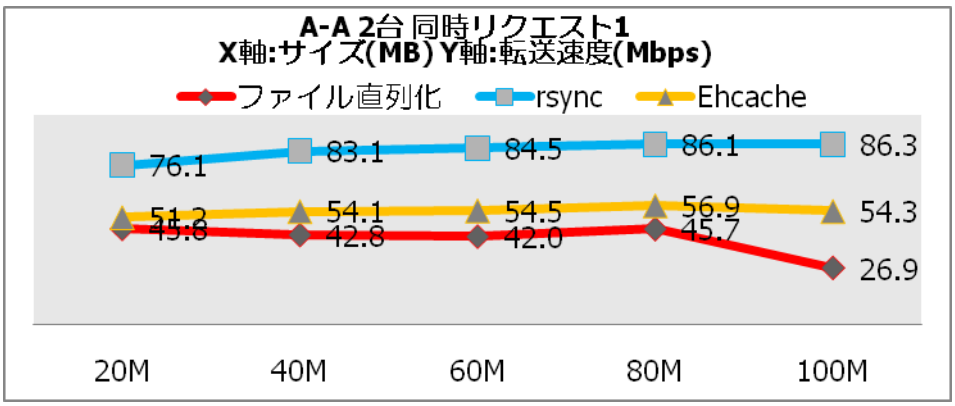
(4) 作成した 3 種類の試作に対し、性能評価を実施した。

実施内容

計測の一例として、All-To-All 2 台構成の転送速度を、3 方式間で比較したグラフを示す。

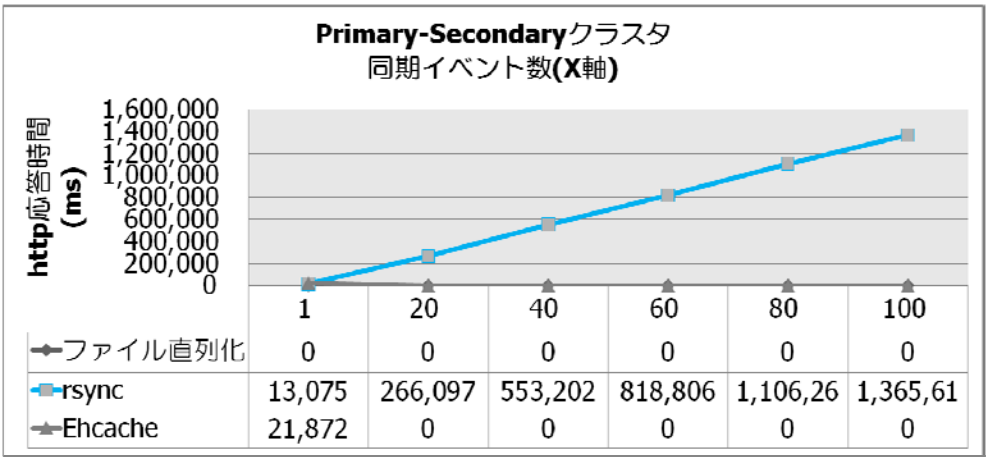
転送速度は、全般を通じて 試作 2( rsync 方式 )が優れている傾向を示している。

表 13　性能評価




また、高負荷時( Primary-Seconary 4 台構成、100Mbyte 同期 )において、試作 1 と 試作 3 では、メモリ不足に起因する実行エラーが発生し、実用面での問題を示した。

表 14　高負荷時におけるレプリケーション実績



## ４－２－３　達成状況及び今後の課題

平成 21 年度は、既存のアプリケーションサーバの負荷分散、冗長機構の調査および基礎性能を評価した。また、アプリケーションサーバが機能を有していない一時ファイルに関する取り扱い機構を複数のパターンで試作し、簡易的な基礎性能評価を実施した。これに引き続き平成 22 年度は、既存アプリケーションサーバ製品で負荷分散、冗長機構において実装されていないことが確実な一時ファイルに関する取り扱いの試作機構の基礎性能を評価する。評価項目は次のとおりとする。

①RCM-Web サーバの分散化ができ、ロードバランス機能により負荷を分散できること。

②RCM-Web サーバの一部に障害が発生した場合は、他の RCM-Web サーバが処理を継続できること。

③RCM-Control サーバの分散化ができ、ロードバランス機能により負荷を分散できること。

④RCM-Control サーバの一部に障害が発生した場合は、他の RCM- Control サーバが処理を継続できること。

⑤RCM-DB サーバの分散化ができ、ロードバランス機能により負荷を分散できること。

⑥RCM-DB サーバの一部に障害が発生した場合は、他の RCM- DB サーバが処理を継続できること。

具体的には、次の研究開発を行う。

(1) ①③⑤RCM-Web、Control、DB サーバの分散化を行い、ロードバランス機能により負荷を分散させることを可能とする。

(2) ②④⑥RCM-Web、Control、DB サーバの一部に障害が発生した場合は、他の RCM-Web サーバで処理が継続することを可能にする。

(3) ソフトウェアでなくハードウェアで本機構を実現することを検討し、それに伴う改良およびその基礎性能を評価する。評価自体は、平成 23 年度も継続調査とする。

## ４－３　データベースの高機密化（排他的記録）機構の研究開発

### ４－３－１　研究開発内容

(1) 支配下サーバを動的に追加できる機構を設計し、試作実装した。
(2) 試作機構の基礎性能を評価した。

### ４－３－２　実施状況

(1) 支配下サーバの動的追加機構を設計・試作実装した。

実施内容

支配下サーバは、図５に示すように、プロジェクト単位で、データを格納できる機構を設計・試作した。

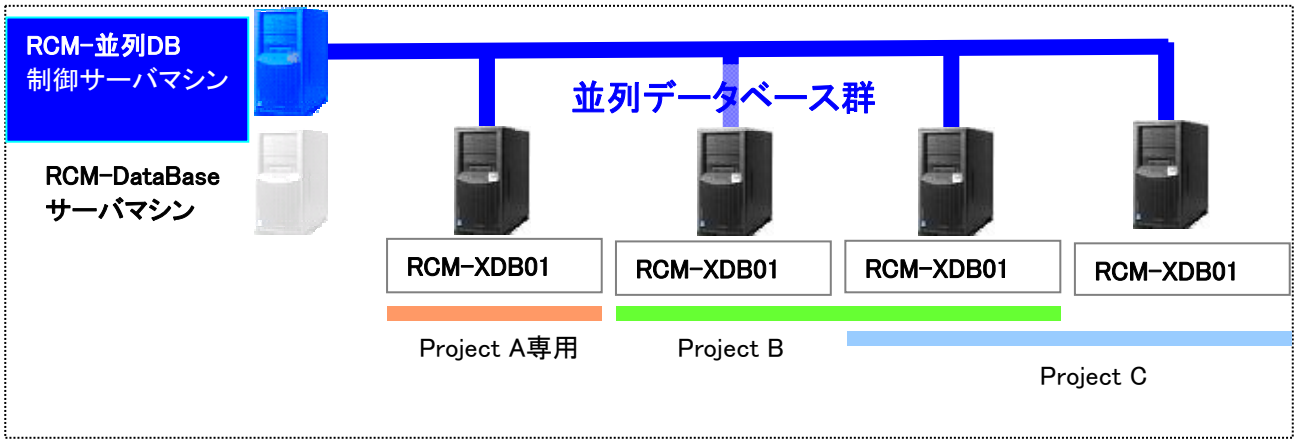



図 5　支配下サーバ構造図

RCM-並列 DB 制御サーバマシンでは、並列データベース群に対して、下記の設定ファイルで管理を行う機構を試作した。

```
<?xml version="1.0" encoding="Shift_JIS"?>
<ParallelSetting>
  <!-- default=roundRobin, option=greed -->
  <assignRule>roundRobin</assignRule>

  <DBServer>
    <name>DB1</name>
    <protocol>REST</protocol>
    <address>http://aaaa/RCM-DB/action/request.html</address>
    <maxNode>10000</maxNode>
    <project>ProjectA</project>
  </DBServer>

  <DBServer>
    <name>DB2</name>
    <protocol>REST</protocol>
    <address>http://bbbb/RCM-DB/action/request.html</address>
    <maxNode>10000</maxNode>
    <project>ProjectB</project>
  </DBServer>

  <DBServer>
    <name>DB3</name>
    <protocol>REST</protocol>
    <address>http://cccc/RCM-DB/action/request.html</address>
    <maxNode>10000</maxNode>
    <project>ProjectB</project>
    <project>ProjectC</project>
  </DBServer>

  (途中省略)

</ParallelSetting>
```

図 6　設定ファイル XML

各並列サーバ群には、<project>タグで明記されたプロジェクトに追加するデータのみが格納されるように、RCM-並列 DB 制御サーバマシンでデータ登録時に自動振り分けを行う。

(2) 支配下サーバの動的追加機構の試作に対し、実際に操作を行い、スムーズに追加が行えることを確認した。

実施内容

支配下サーバの動的追加機構の動作試験として、下記の試験を行った。

(1) 前ページの設定ファイルで、**<name>DB3</name>の定義を記載せずに運用を開始する。**
**→ この時点で、projectB のデータはすべて<name>DB2</name>の支配下サーバに記録される。**

(2) 前ページの設定ファイルで、**<name>DB3</name>の定義を追加する。**
**→ これ以降、projectB のデータ登録は、<name>DB2</name>と<name>DB3</name>の双方に、交互に登録される。**

### ４－３－３　達成状況及び今後の課題

(1) 平成 21 年度は、支配下サーバを動的に追加できる機構を試作した。これに引き続き平成 22 年度は、支配下サーバを動的に追加できる試作機構の基礎性能を評価する。
(2) 任意のデータおよびメタデータを別ハードウェアに格納するための DB リクエストを開発し、それ以外のデータとハードウェア的に分離した管理が可能となる機構を開発する。

## ４－４　Workflow や高品位な UI を GUI で設定、変更を可能にする機構の研究開発

### ４－４－１　研究開発内容

(1) 現在の RCM における Workflow 自動生成機構内の ASCII ファイルの XML 化支援ツールである RCM-Picker を拡張し、より大きなファイルや複雑なパターンに対して、Workflow 自動生成機構が有効となるように拡張試作を行なった。
(2) (1) の試作の性能を評価した。
(3) 現在の RCM における Workflow 自動生成機構のマクロ機構であるデータ解析ウィザードを拡張し、マクロ間連携、複数マクロを束ねたスーパーマクロを作成できるように拡張試作を行なった。
(4) (3) の試作の性能を評価した。
(5) 現在の RCM における Workflow 自動生成機構の変数置き換え機構である簡易 UI を拡張し、Workflow のプリミティブを UI から自動生成できるように拡張試作を行なった。
(6) (5) の試作の性能を評価した。

## ４－４－２　実施状況

(1) RCM-Pickerを拡張し、より大きなファイルや複雑なパターンに対して、Workflow 自動生成機構が有効となるよう拡張試作を行なった。

(2) (1) の試作の性能を評価した。

実施内容

大きなファイル( 例えば 1Gbyte のテキストファイル )を Load すると、本試作以前の RCM-Picker は メモリ不足のため、マウス操作に対するレスポンスが極端に遅くなることがあった。

本試作では、Load するテキストファイルのうち、処理したいエリアを定義し、関心領域のみをメモリにロードして、表示する拡張を行った。

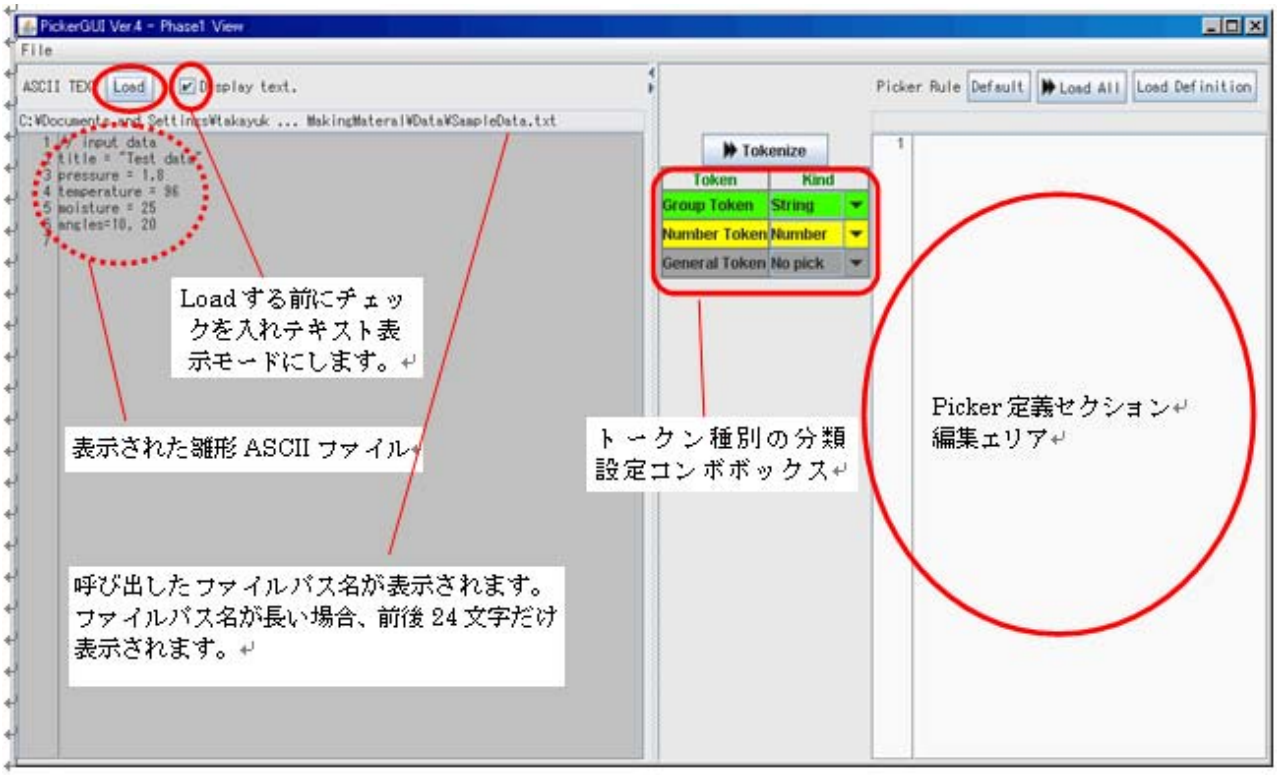



図 7　拡張 RCM-Picker テキスト表示モード初期画面

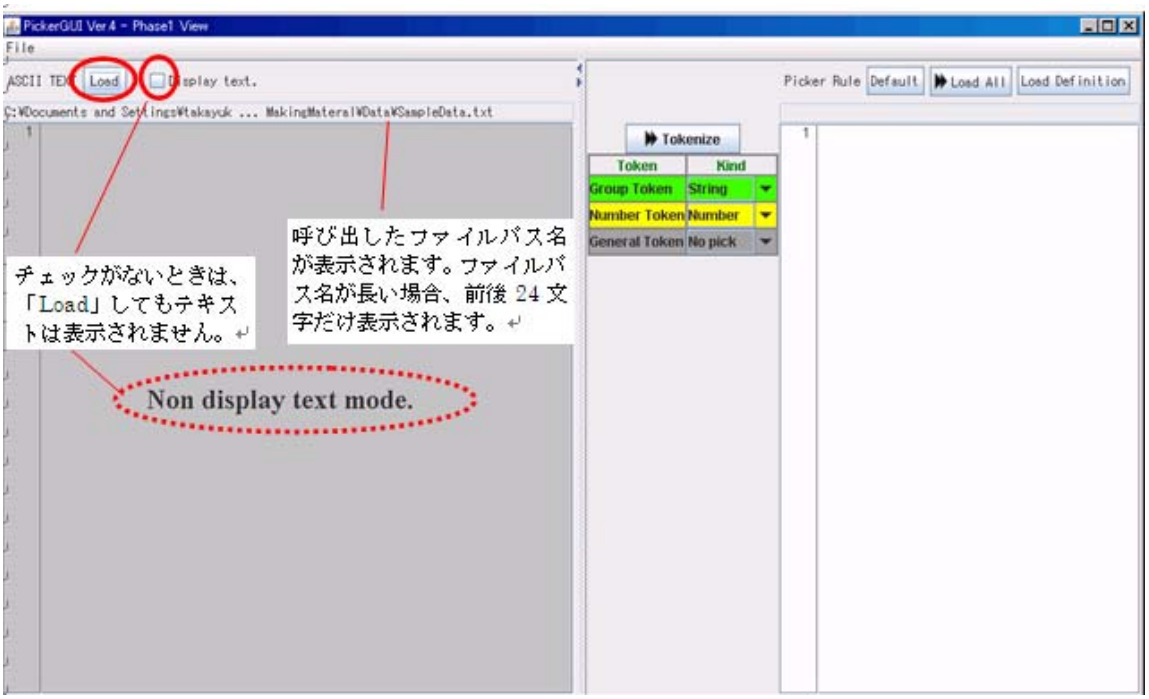



図 8　拡張 RCM-Picker テキスト非表示モード初期画面

トーカナイズ後は、テキスト非表示モードであっても、トークン pickup 領域のみ、トークンごとにトークン種別の色分けが行われたテキストがテキストエリアに表示される。

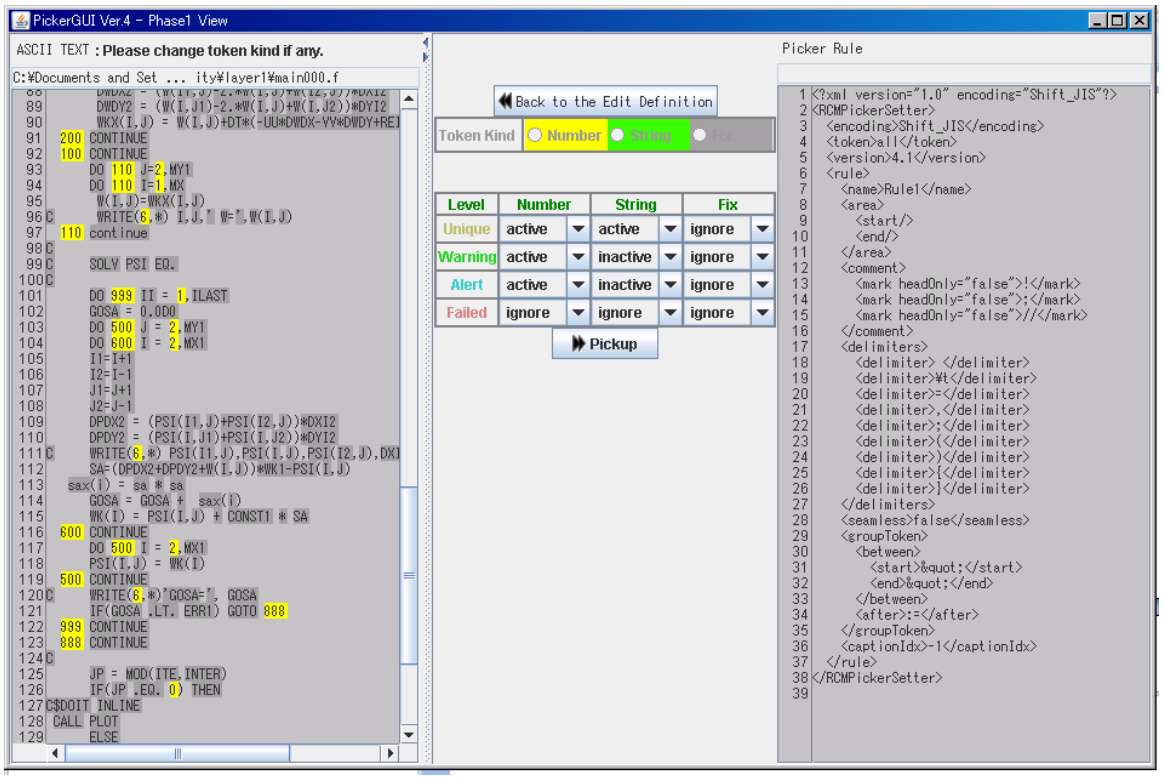



図 9　拡張 RCM-Picker トークン種別設定画面

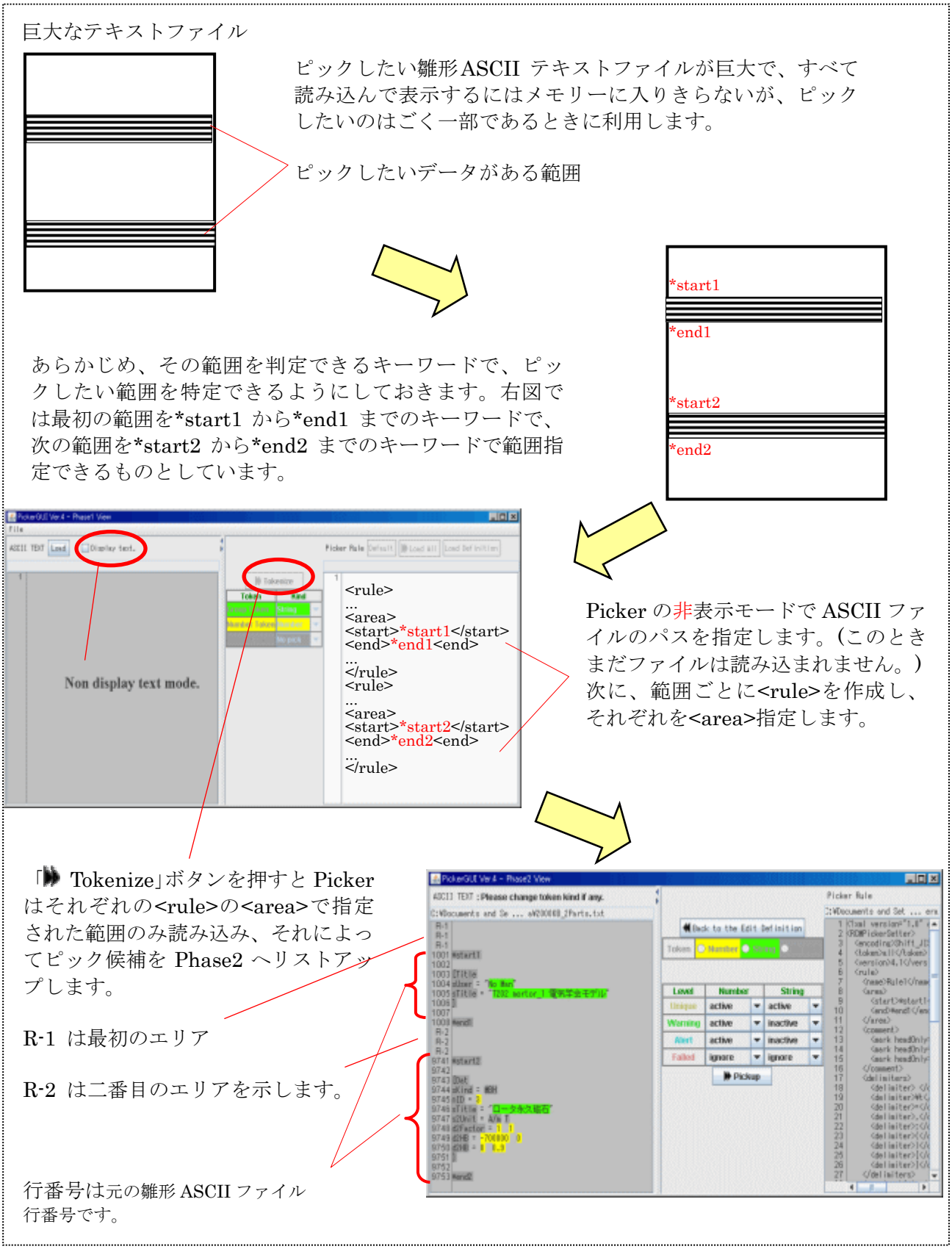



図 10　拡張 RCM-Picker の画面遷移について

(3) データ解析ウィザードを拡張し、マクロ間連携、複数マクロを束ねたスーパーマクロを作成できるよう拡張試作を行なった。

(4) (3) の試作の性能を評価した。

実施内容

処理コンポーネントを複数まとめて実行予約を行い、マクロ実行ができる機構を試作し、動作試験を行った。

具体的には、同じ 5 つのマクロ実行を、下記のようにマクロ間連携する。

(試作前)

1. 処理コンポーネント① を手動で実行する。
2. 1. の終了後、処理コンポーネント② を手動で実行する。
3. 2. の終了後、処理コンポーネント③ を手動で実行する。

(マクロ間連携機構)

1. 処理コンポーネント① → ② → ③ を実行予約する。
2. 処理コンポーネント① を実行する。

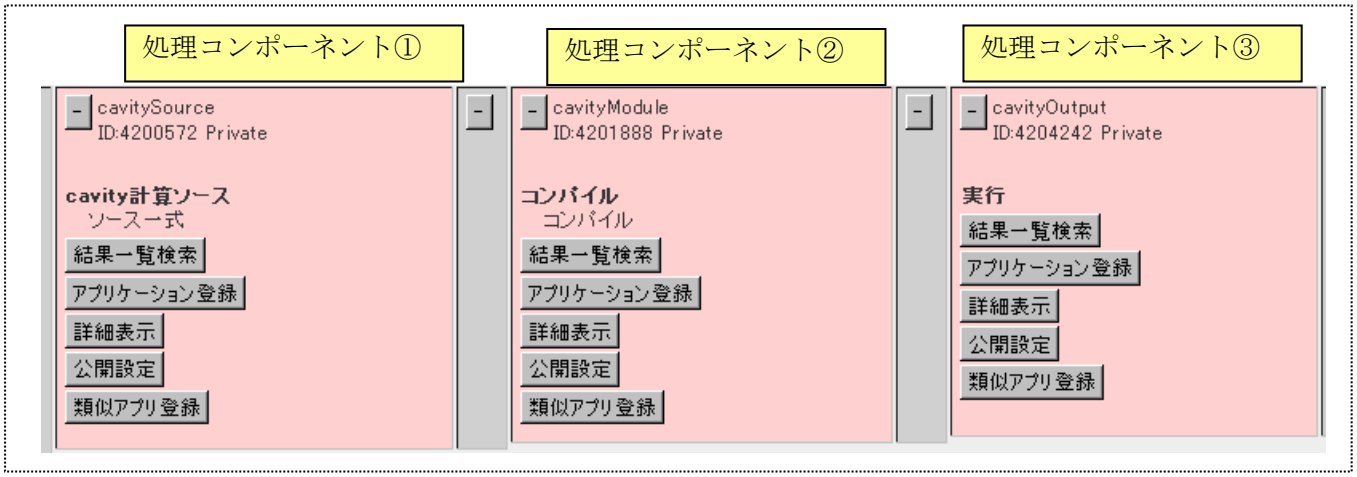



図 11　拡張データ解析ウィザードにおけるデータ処理連携コンポーネント

図 11 のマクロ間連携を実行すると、1 回のコンポーネント実行で、3 つの結果を自動生成できるようになった。

・試作前は 手作業で 3 回のコンポーネント実行を操作していた。

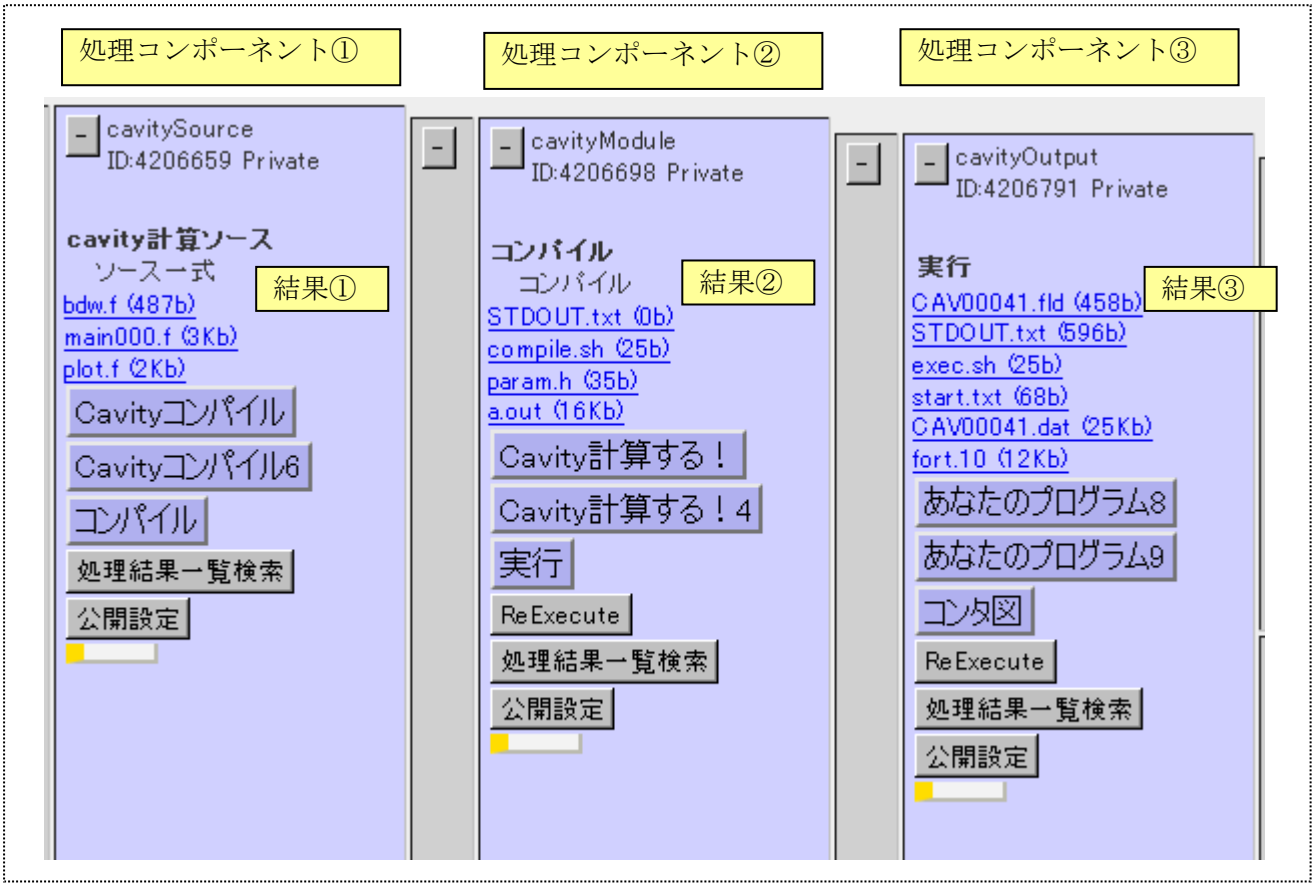



図 12　拡張データ解析ウィザードにおける連携コンポーネントの結果連携表示

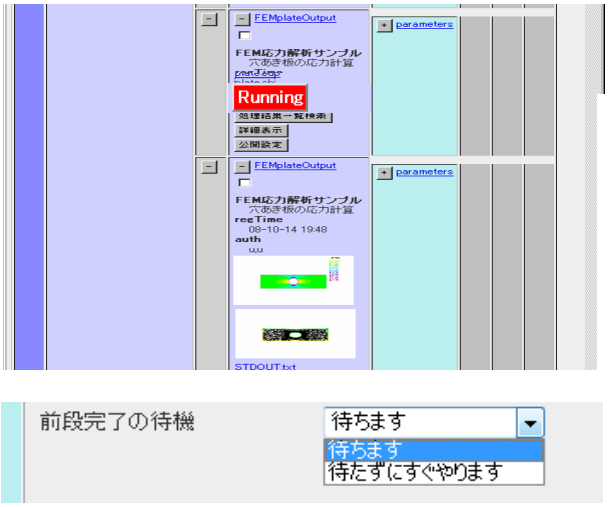



図 13　拡張データ解析ウィザードの連携機能モード

マクロモードでは、2 種類のモードを試作した。

1. 前段の処理コンポーネントの終了を待って連続実行するマクロモード（ 図 13 で「待ちます」で実行予約する）

2. 前段の処理コンポーネントの実行途中に、その時点での出力で解析を行うマクロモード（ 図 13 で「待たずにすぐやります」で実行予約する）

今後、利用者の利用状況などをヒアリングし、マクロモードの拡充および洗練を図っていく。

(5) 簡易 UI を拡張し、Workflow のプリミティブを UI から自動生成できるよう拡張試作を行なった。

(6) (5) の試作の性能を評価した。

実施内容

1)RCM-Picker が自動生成した XML を解析して、RCM 上でシミュレーションの入力パラメータを自動生成する試作を行った。

(試作前)

手作業で XML 内に、GUI 用のキーワードを記述していた。

(試作後)

1. RCM-Picker(図 7 から図 10)上で、マウス操作で入力欄の種別や初期値の設定を行い、設定値を XML で出力する。

2. 1.の出力 XML を RCM に登録することで、自動的に入力欄( UI )を自動生成する拡張試作を行った。
( 図 14 および図 15 に例示する UI を自動生成する )

この試作を利用することで、シミュレーションのポータル化に要する時間・労務コストが従来の 1/10 以下に削減できると評価している。

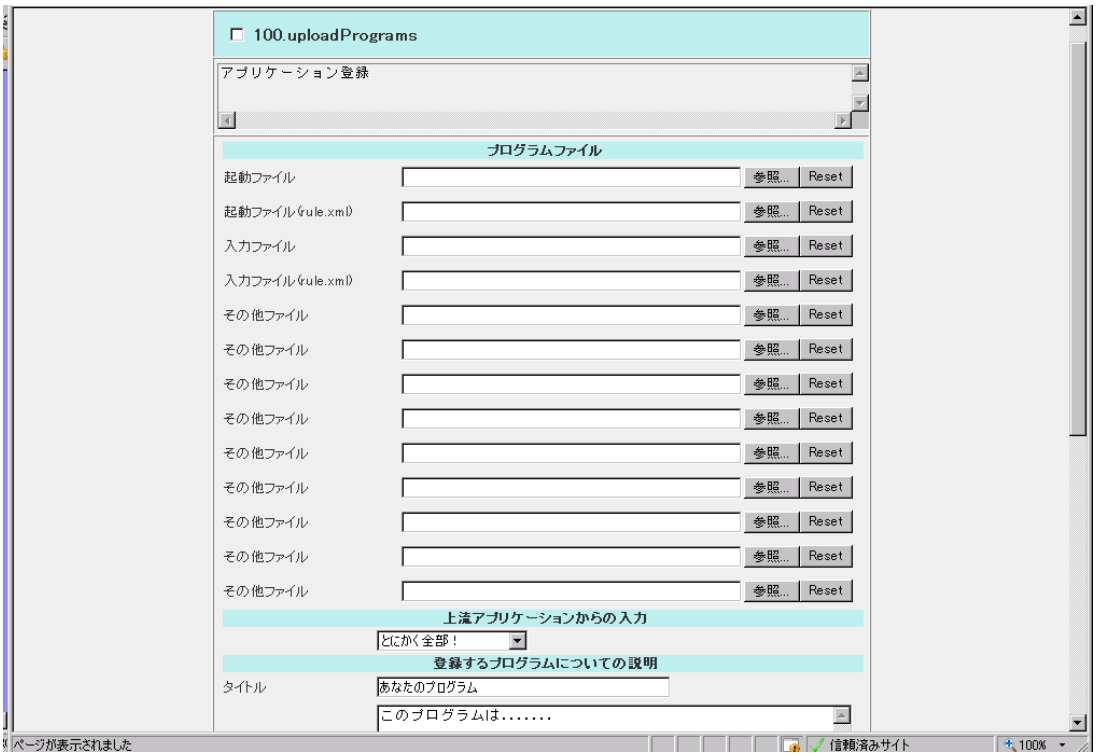



図 14 拡張簡易 UI 設定画面の例 1

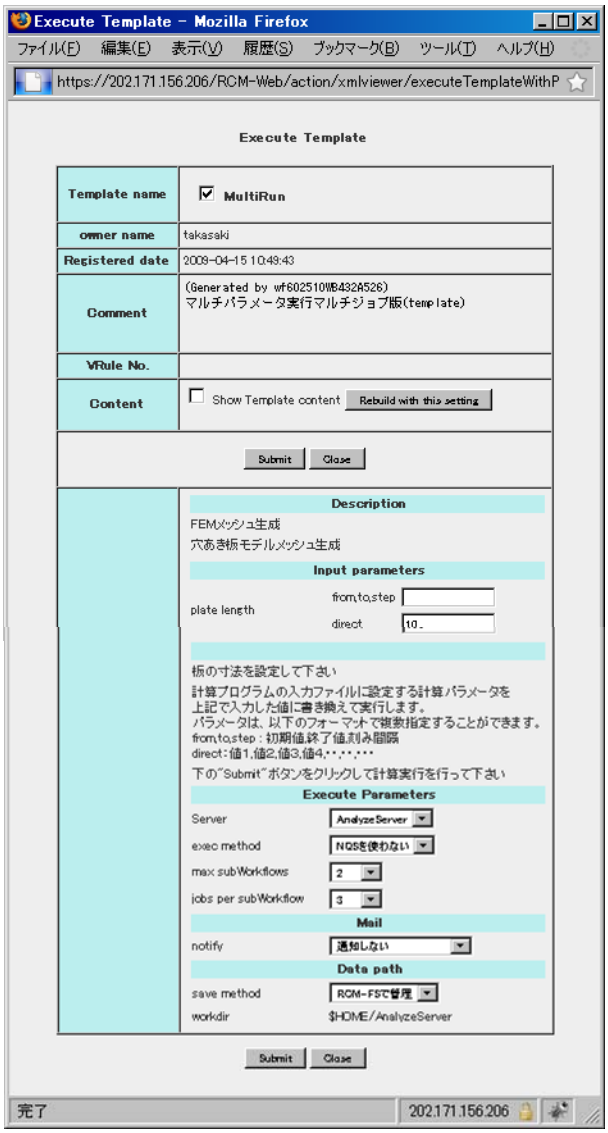



図 15　生成された拡張簡易 UI 画面の例 2

2) XSLT により、従来の出力系（XML-Viewer）表示画面をカスタマイズできるようにした。これにより、シミュレーションのポータル化のカスタマイズ能力、表現力が PaaS クラウドサービスにおいて大幅に確保できることになった。

次ページの 図 16 は、本試作前のデフォルトの XML-Viewer 画面である。従来は、この色合いや、表示情報の種類を変更することはできなかった。

図 17 は、本試作を用いて、XSLT を追加登録し、画面をデザインした例である。図 17 の文言やボタンなどは、XSLT を用いて、自由に作成することができる。

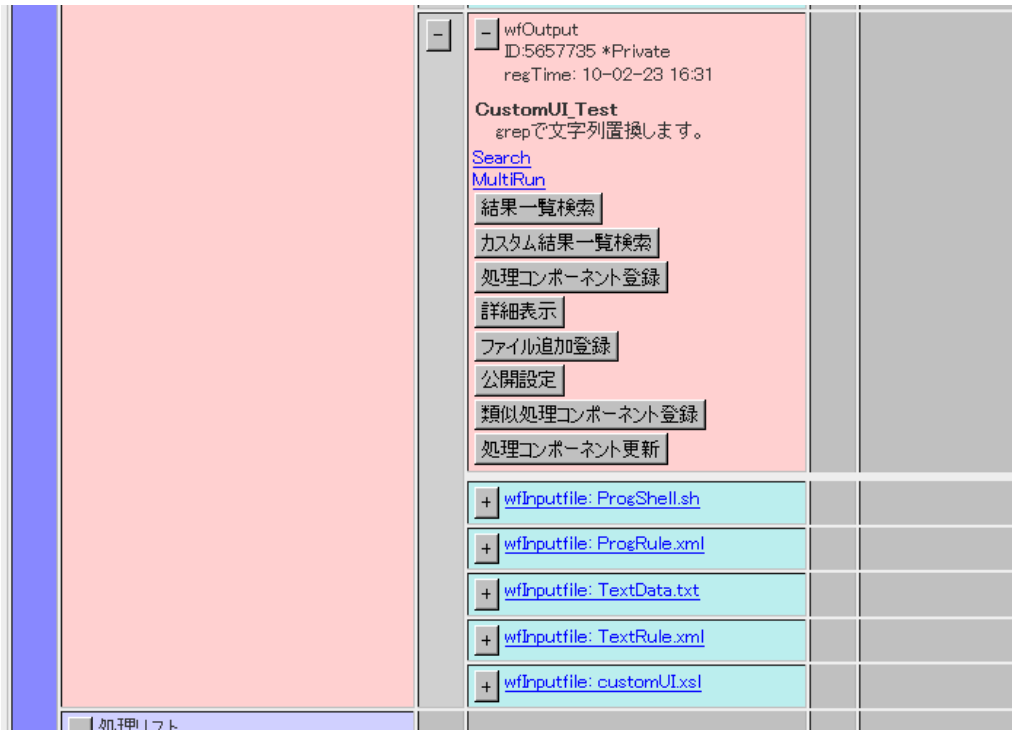



図 16　従来の固定的な XML-Viewer 表示

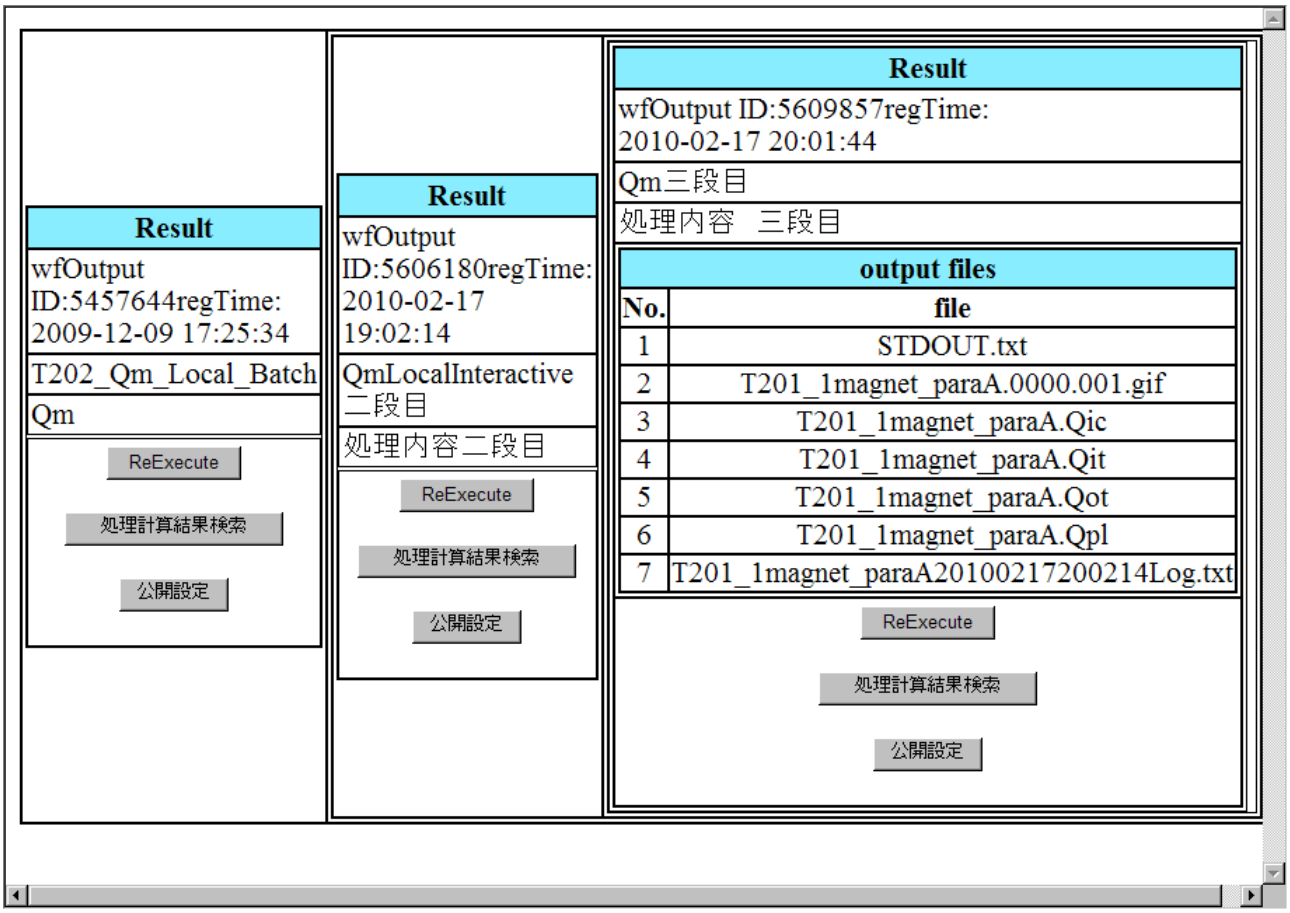



図 17　XSLT によるカスタマイズ表示画面

## ４－４－３　達成状況及び今後の課題

平成21年度は、ASCIIファイルのXML化支援ツールおよびデータ解析ウィザードの拡張を実施した。これに引き続き平成22年度は、Workflow自動生成機構のマクロ機構であるデータ解析ウィザードをより高品位なUIが生成できるようにさらに拡張し、性能を評価する。また、平成21年度は、現在のRCMにおけるWorkflow自動生成機構の変数置き換え機構である簡易UIを拡張し、Workflowのプリミティブを UIから自動生成できる拡張試作を実施した。これに引き続き平成22年度は、この拡張試作の性能評価を行い、改善を行なう。

拡張試作の性能の評価項目は次のとおりとする。

① XML-Workflow（入力系UI、出力系UIを含む）をGUI画面で設定できること。
② XML-Workflow 設定GUI画面では、XMLのタグ名入力は不要にすること。
③ XML-Workflow 設定GUI画面では、固定的な複数選択しは、ドロップダウンで選べるようにすること。XML-Workflow設定GUI画面では、数値、日付など明らかなフォーマット指定がある場合は、入力時にそのフォーマットを誘導するとともにチェック機構を有すること。
⑤ XML-Workflow 設定GUI画面では、job間の相関性（参照、重複不可等）を意識させながら入力できること。
⑥ 入力系UI、出力系UI設定は、レイアウト設定が簡単なようにHomePage作成ツールのようなレイアウト画面で設定できること。

具体的には、次の研究開発を行う。

(1) ①XML-Workflow（入力系UI、出力系UIを含む）のGUI画面での設定を可能にする。
(2) ②～⑤のXML-Workflow 設定GUI画面では、次の機能を満たすことを可能にする。
  ・XMLのタグ名入力は不要にする。
  ・ドロップダウン選択機構を有する。
  ・フォーマットを誘導、チェック機構を有する。
  ・job間の相関性誘導機構を有する。
(3) ⑥入力系UI、出力系UI設定は、レイアウト設定が簡単なようにHomePage作成ツールのようなレイアウト画面で設定することを可能とする。

## ４－５　RCM システム間（WebServer-WebServer）の連携機構の研究開発

### ４－５－１　研究開発内容

(1) 各システムのデータベースにおいて一意性を保つようにデータベースの拡張開発を行なった。

(2) 支配下サーバ設定の各システムにおいて一意性を保つように支配下サーバ設定方式の拡張開発を行なった。

### ４－５－２　実施状況

(1) 異なるセットの RCM 間で、データの識別番号が重複しない機構を、RCM システムのオプション機能として開発した。

実施内容

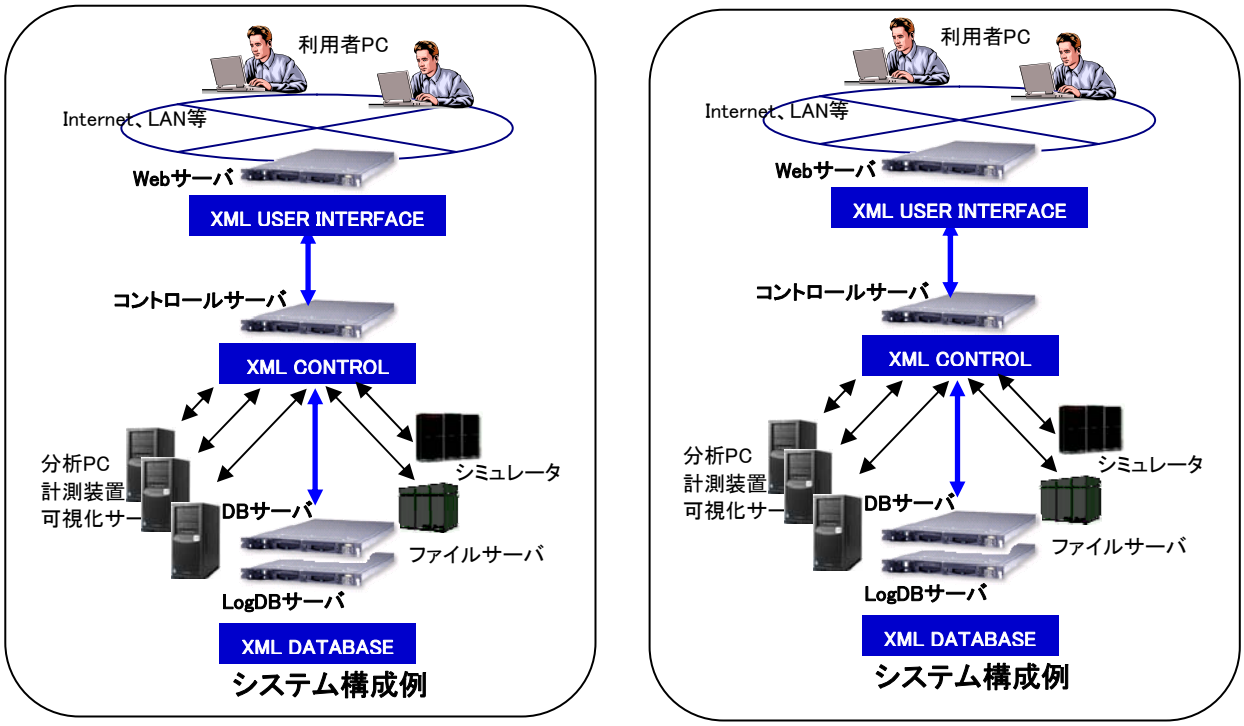



図 18　実施内容図

本課題で開発したオプションを組み込む前は、図 18 のように各 RCM セット内で、XML タグの一意性は、@tagid という属性に、自動的に付与するシリアル番号で管理している。

この@tagid の値は、各 RCM セット内で、独自に採番するため、セットが異なれば、番号が重複する。

| | |
|---|---|
| <data tagid="1234">データ 1</data> | <item tagid="1234">道具 1</item> |

本課題で開発したオプションでは、@tagid の採番に、各 RCM セット共通の採番機構を導入し、複数の RCM 間で @tagid の値がユニークになる機構を実現した。

| | |
|---|---|
| <data tagid="1234">データ 1</data> | <item tagid="1235">データ 2</item> |

採番機構の概念図を、下図に示す。

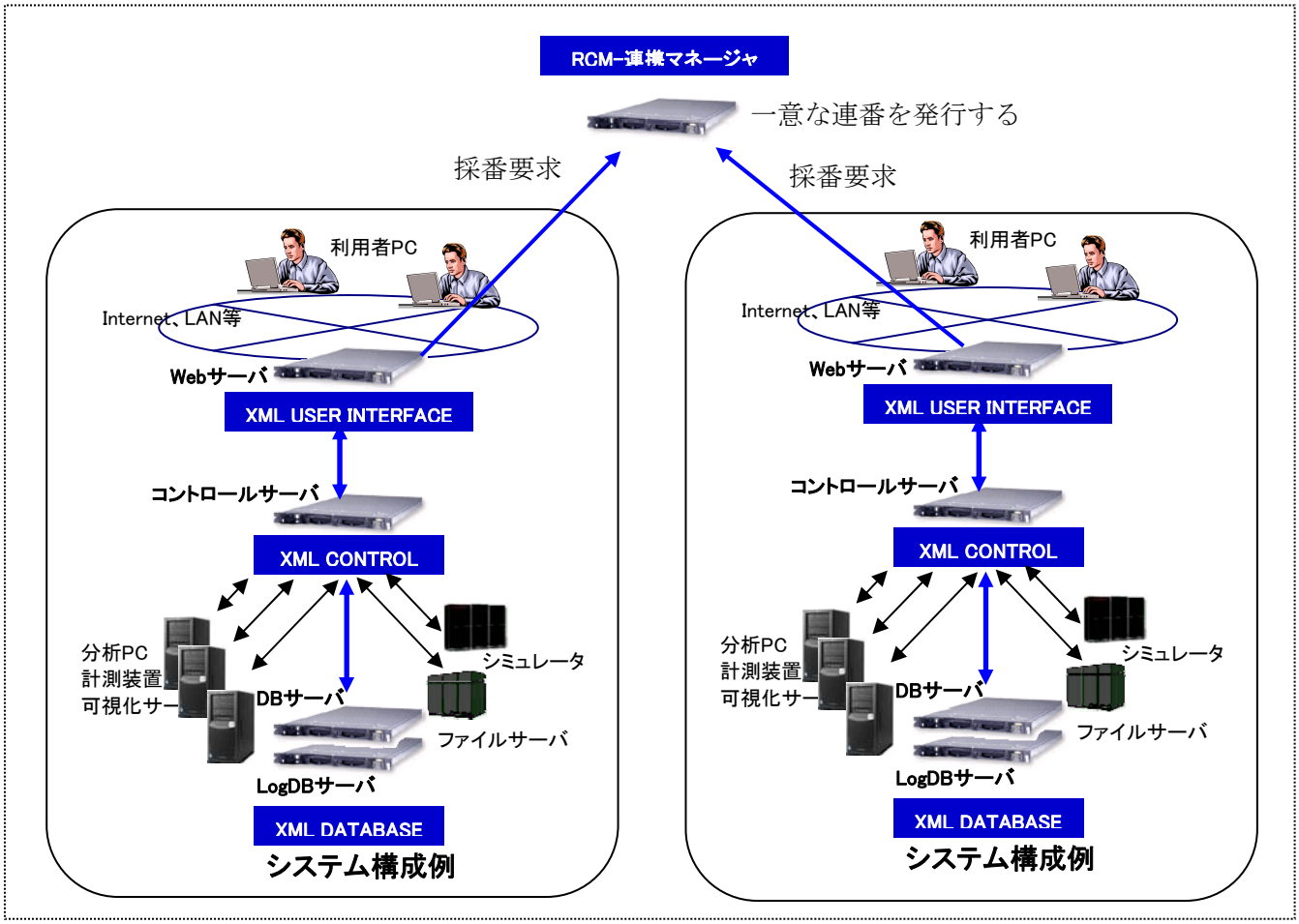



図 19　採番機構の概念図

(2) 異なるセットの RCM 間で、支配下サーバの計算リソースを利用する機構を、RCM システムオプション機能として開発した。

実施内容

下記の機構を開発した。

1. 各 RCM セットは、支配下サーバの計算リソースを指定する際に、<serverType>と<serverName>を指定する。

   本機構導入前は、<serverType>と<serverName>に該当する計算リソースが、当該 RCM セット内に定義されていない場合、計算リソース取得エラーとなっていた。

2. 本機構により、当該 RCM セット内に定義がない計算リソースについては、RCM 連携マネージャに、計算リソースを問い合わせ、<serverType>と<serverName>が一致する計算リソースを、他の RCM セットから探索し、利用する。

3. 協調する RCM セット間で、<serverType>と<serverName>のルールを定義することにより、サイト間をまたいだクラウドシステムとして、さらに計算リソースの共有化を図ることが可能となった。

## ４－５－３　達成状況及び今後の課題

平成 21 年度は、各システムの一意性を保つようにデータベースの拡張開発を実施した。これに引き続き平成 22 年度は、DB 機構の一意性の性能評価をし、改善を行う。また、平成 21 年度は、各システムの Workflow 内での支配下サーバ指定の一意性を保つための拡張開発を実施した。これに引き続き平成 22 年度は、支配下サーバ指定の一意性機構の性能評価をし、改善を行う。

具合的には、次の研究開発を行う。

(1) Workflow 内で異なる RCM システムで実行される job について、他方の RCM システムに処 理を依頼し、その戻値を受け取ること、および job 内で異なる RCM システム支配下のサーバを利用する場合、他方の RCM システムと連携して 1 つの job 機能を果たすことを可能とする。

## ４－６　既存（非 RCM）社内 R&D システムとの連携機構の研究開発

### ４－６－１　研究開発内容

(1) 任意の通信プロトコルを受信し、応答するためのブリッジモジュール機構を開発した。既存システムの通信プロトコルに合わせ専用ブリッジモジュールを開発し、Controller に簡単に追加できる機構を試作開発した。

(2) (1) の試作の機能、性能を評価した。

### ４－６－２　実施状況

(1) 任意の TCP/IP 通信プロトコルを Controller に追加できる機構を試作開発した。また、動作確認のため、必要最低限の管理機構の試作( 実用レベルではない ) を実施した。

実施内容 1

既存システムの通信プロトコルは、規格化されていないものが多く、独自プロトコルで実装されていることが一般的である。

RCMBridge 機構として、RCM に独自プロトコルとの通信口を追加し、

1. 既存のシステムの出力を受け取り、Template を起動して、RCM-DB にデータ登録を行うことが可能であることを確認した。

2. RCM で稼働中の WorkFlow から、RCMBridge の API を起動する仕組みを試作し、WorkFlow から、既存システムに対して、機器の停止などの命令を発行することが可能であることを確認した。

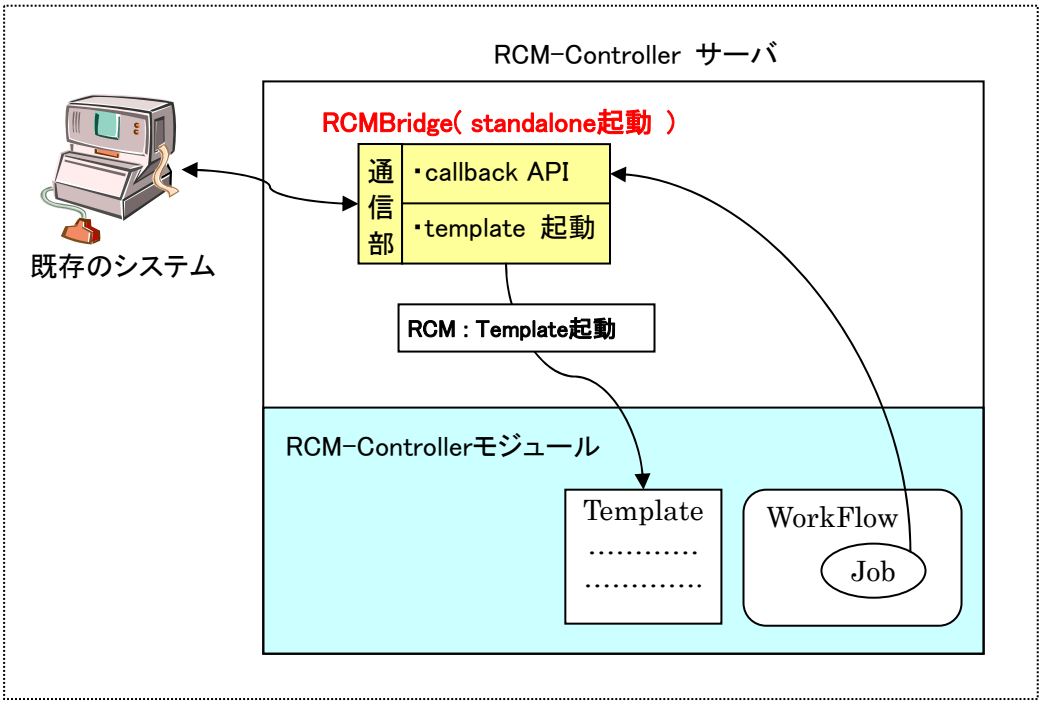



図 20 RCMBridge による既存システムとの通信

実施内容 2

RCMBridge を RCM システム内に複数登録、稼動する仕組みの試作を行った。下図は、試作した RCMBridge 管理画面である。

この管理画面を通じて、複数の RCMBridge が同時に起動し、いずれも干渉を受けずに、独立して稼動し、既存システムおよび WorkFlow と連携できることを確認した。

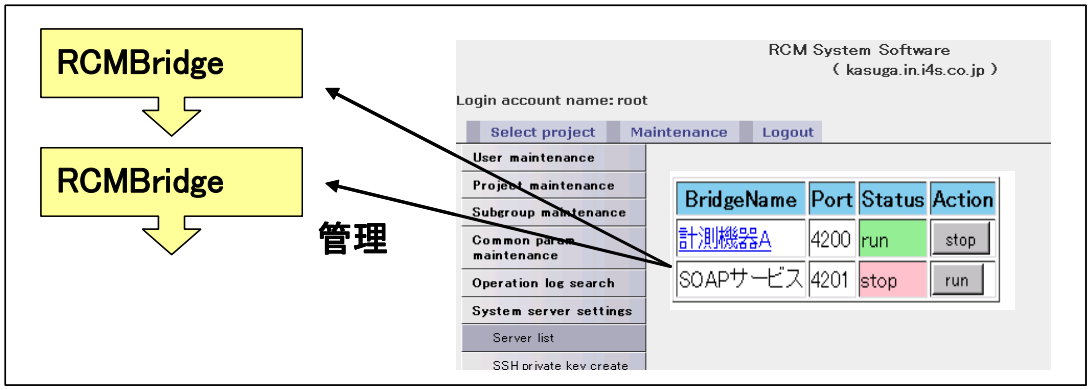



図 21　RCMBridge 管理画面

実施内容 3

RCMBridge の開発工数について考察した。
既存システムとの通信部分は、おのおの独自プロトコルであるため、各自がコーディングを行う必要がある。

他方、RCMBridge から Template を起動する箇所、及び WorkFlow から RCMBridge に命令を発行する箇所は、大半を RCM システムがライブラリとして提供し、この部分のコーディングはわずか 200 行程度で十分な実用性をもつことが確認できた。

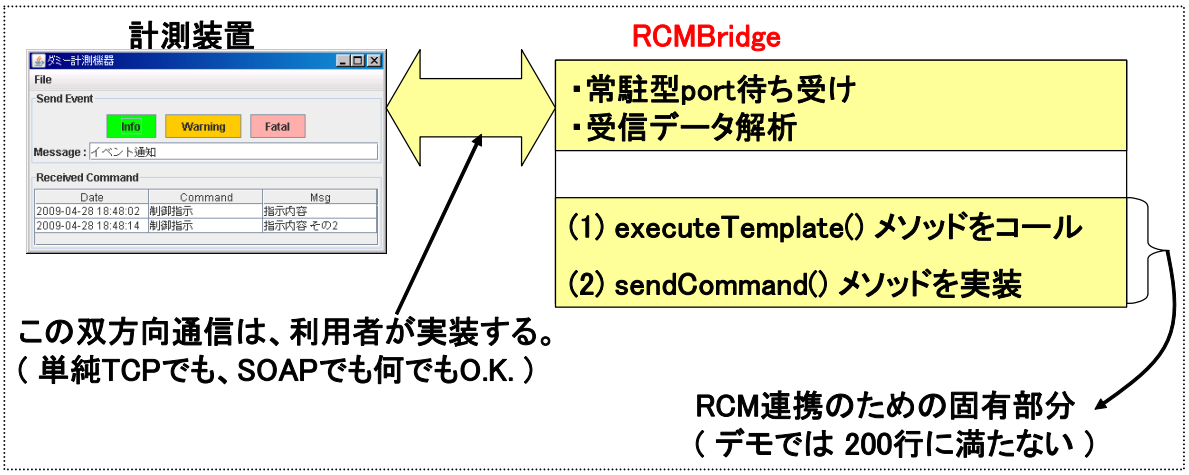



図 22　RCMBridge のコーディング部位

(2) 実際にサンプルにより機能評価を行い、TCP/IP 通信機能および Controller との連携機構の実用性を確認した。性能面については、特に目立った速度遅延などがないことを確認した。

### ４－６－３　達成状況及び今後の課題

(1) 平成 21 年度は、ブリッジモジュール機構の試作開発を実施した。これに引き続き平成 22 年度は、本機構を機能・性能を評価する。具体的には、任意の通信プロトコルを受信し、応答するためのブリッジモジュール機構を開発・評価する。
(2) ブリッジモジュールにおいて SOAP プロトコルを使う場合、ESB を使った通信をサポートできるようにする。
(3) ブリッジモジュールの稼働・停止状態を RCM システムのメンテナンスモードから監視、制御できるようにする。

## ４－７　実証システムの構築と運用の研究開発

### ４－７－１　研究開発内容

(1) 現状の RCM を下記商用 OS、商用 Web ミドルで動作試験を行なった。
(2) 性能、機能面で利用上問題がある部分に関して、ボトルネックポイントを同定し、改善プランを策定した。

### ４－７－２　実施状況

(1) 商用 OS( Red Hat Enterprise Linux および Windows )と商用 Web ミドル( JBoss, WebLogic, WebSphere )の 6 種類の組み合わせ上に構築した RCM システムの評価作業を実施した。

商用 Web ミドルに関しては、現状の RCM そのままでは、動作できないことが判明した。ただし、修正は大規模ではないが、商用 Web ミドルそれぞれ特性があり、統合的に同じ修正で RCM システムを稼働させることが困難であることが分かった。

また、「RCM システムの負荷分散、冗長機構の研究開発」において、非商用 Web ミドルの Tomcat は、機能面で遜色がなく、性能面でも大きな低減が見当たらなかった。

したがって、CAE クラウドを実際の商用展開する場合には、商用 Web ミドルを使う可能性は排除しないが、本研究開発では、一番中立的な、非商用 Web ミドルの Tomcat をベースに実証システムを構築する方針とした。

また、実際に実証システムで利用可能にするアプリケーションの選定、協力交渉を進めた。現時点では、以下のアプリケーションが候補であり、現時点での交渉状況を以下の表に記載しておく。

表 15　実証システムに搭載を計画しているアプリケーション一覧

| アプリケーション | 動作環境 | 許可依頼先 | 状況 |
|---|---|---|---|
| Qm（動磁場解析コード） | Windows XP | 株式会社シフトロック | 許可済み |
| AVS/Express | Windows XP | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| AVS/Express | Redhat5.3 | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| Ensight | Redhat5.3 | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| RealIntage | Windows XP | 株式会社ケイ・ジー・ティ | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| HexaGrid | Redhat5.3 | 株式会社計算力学研究センター<br>(独)宇宙航空研究開発機構 | 内諾済み<br>2010 年 6 月以降に正式依頼予定 |
| ADVC | Redhat5.3 | 株式会社アライドエンジニアリング | これからアプローチ |
| 依頼先が販売可能なソフトウェア全般 | ----- | 株式会社アドバンスソフト | 依頼済み。回答待ち。 |
| 依頼先が販売可能なソフトウェア全般 | ----- | 株式会社計算力学研究センター | 依頼済み。回答待ち。 |
| 依頼先が販売可能なソフトウェア全般 | ----- | 株式会社ソフトウェアクレイドル | まだ、アプローチしていない。 |
| ANSYS | Redhat5.3 | サイバーネットシステム株式会社 | ライセンスの問題で最終ユーザしか購入ができないので、実証試験には利用できない可能性が高い。ただし、1ヶ月のデモライセンスで対応可能か協議中。 |
| Gaussian | Redhat5.3 | 株式会社ヒューリンクス | ライセンスの問題で最終ユーザしか購入ができないので、実証試験には利用できない可能性が高い。 |
| ProteinDF System 1.4.2 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| ABINIT-MP Ver.4.1 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| PHASE Ver.7.01 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| ABCAP | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| FXZTX ver.1.0 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| ASCOT ver.3.01 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| CAMUS-FSIS ver.2.01 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| CIAO ver.2.16 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| MonCaFe ver.1.20 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| MEAMDB ver.1.00 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| REVOCAP_Magnetic Ver.1.0 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| REVOCAP_Coupler ver1.0 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| FrontFlow/red-Ver.3.0 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| FrontFlow/blue-Ver.5.2 | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |
| FRONT-STR | | 東京大学 | 国家プロジェクトの RSS21 プロジェクトの成果であり、申請すれば、利用可能 |

(2) (1) の機能評価時に、必要となった修正を RCM システムに追加実装した。

Windows に関しては、動作をさせるために以下の修正を行った。

1) Windows OS を RCM-Controller マシンとするために、必要なプログラムを追加する。

SSH 秘密鍵を作成する ssh-keygen コマンドは、Windows OS には備わっていないため、Cygwin というツール群の openssl パッケージを追加でインストールする。
→ openssl パッケージ内の、ssh-keygen.exe がインストールする。

RCM-Conroller-Conf/RCMConfig.xml 内の ssh-keygen の参照箇所を上記の ssh-keygen.exe のフルパス( 例 c:¥cygwin¥bin¥ssh-keygen.exe ) に書き

換える。

2) RCM-Controller-Conf 内の AnalyzeServer.xml, SimulationServer.xml に記述している <address>localhost</address> は、別の Linux マシンの IP アドレスに設定しなおす。

3) RCMFileServer.xml, RCMBackupServer.xml, RCMLogFileServer.xml RCMLogBackupServer.xml の <workDir> は C:¥RCM¥FileServer など、Windows のフルパスに書き換える。

（ 該当するディレクトリも作成しておく。）

2), 3)の変更後は、tomcat を再起動する。

4) Firewall は 8080 ないし、80 を空ける。

### ４－７－３　達成状況及び今後の課題

(1) 平成 21 年度は、商用 OS、商用 Web ミドル（WebSphere など）で動作試験を行った。これに引き続き平成 22 年度は、2 つ以上の RCM システムを別ネットワークで構築、運用する。具体的には、2 つ以上の RCM システムを別ネットワークで構築、運用し、サブテーマ４－１～４－6 の機能が正しく動作するかを 1 ヶ月以上検証する。

(2) サブテーマ４－１～４－6 の機能の性能を評価し、性能、機能面で利用上問題がある部分に関してボトルネックポイントを同定し、改善プランを策定する。

(3) (1) の検証時に運用モデルを明確化し、運用マニュアル、運用における留意点をドキュメントにまとめる。また、そのマニュアルに従って、運用を実際に行って、問題点がないかを確認する。

## ４－８　総括

本研究開発は、R&D 系業務をシステム化した現行の RCM システムを拡張し、PaaS 化するものである。研究開発期間は平成 21 年 11 月から平成 23 年 10 月までの 2 年間で、初年度の今回は平成 21 年度の 5 ヶ月間を対象とする成果報告である。

平成21年度は、次のサブテーマの研究開発を実施した。

ア　HTTPS ベースでリモートアプリケーションサーバの画面を高速に伝送する機構の研究開発

(ｱ) 現状の SSH トンネリングベースの「リモートアプリケーションサーバ画面の伝送」機構を HTTPS ベースによる伝送機構に切り替える試作開発を行なった。

(ｲ) 試作開発と並行して、リモートアプリケーションサーバ上で動作する評価対象アプリケーションが評価版などで無償貸与をしてもらえるか、リースが必要か、購入が必要かなどを調査し、選定および導入を行なった。

(ｳ) 試作開発をベースに Firewall 透過性や通信性能の基礎性能を評価し､SSH で VNC を転送する現行の RCM の方式に対して､FireWall 透過性を付与するために､HTTPS 通信路で VNC を転送する本課題については、実現可能性を示すことができた。

イ　RCM システムの負荷分散、冗長機構の研究開発

(ｱ) 最新のアプリケーションサーバ製品で負荷分散、冗長機構がどこまでどのような方式で実装されているのかを実際の動作テスト、性能測定を含め調査した。

(ｲ) 既存アプリケーションサーバ製品で負荷分散、冗長機構において実装されていないことが確実な一時ファイルに関する取り扱い機構の案を複数検討し、将来性、性能面を考慮して、開発機構の優先順位を評価した。最も優先順位の高い方式に致命的問題が出た場合に迅速に次点案に切り替えられるように開発手順を明確化した。

(ｳ) (ｲ)の一時ファイルに関する取り扱い機構を試作した。

(ｴ) 試作機構の基礎性能を評価し、rsync 方式が優れていることを見出した。

ウ　データベースの高機密化（排他的記録）機構の研究開発

(ｱ) 支配下サーバを動的に追加できる機構を設計し、試作実装した。

(ｲ) 試作機構の基礎性能を評価し、動的追加機構が機能的に十分動作できていることを確かめた。

エ　Workflow や高品位な UI を GUI で設定、変更を可能にする機構の研究開発

(ｱ) 現在の RCM における Workflow 自動生成機構内の ASCII ファイルの XML 化支援ツールである RCM-Picker を拡張し、より大きなファイルや複雑なパターンに対して、Workflow 自動生成機構が有効となるように拡張試作を行なった。

(ｲ) (ｱ)の試作の性能を評価し、従来対応ができなかった GB オーダーのファイルや空白行を含む文字列などに対応できていることを確認した。

(ｳ) 現在の RCM における Workflow 自動生成機構のマクロ機構であるデータ解析ウィザードを拡張し、マクロ間連携、複数マクロを束ねたスーパーマクロを作成できるように拡張試作を行なった。

(ｴ) (ｳ)の試作の性能を評価し、マクロ間連携、複数マクロを束ねたスーパーマクロ機構が動作でいることを確認した。評価自体は、H22 年度前半も継続調査とした。

(オ) 現在の RCM における Workflow 自動生成機構の変数置き換え機構である簡易 UI を拡張し、Workflow のプリミティブを UI から自動生成できるように拡張試作を行なった。
(カ) (オ)の試作の性能を評価し、Workflow のプリミティブを UI から自動生成できることを確認した。

オ　RCM システム間（WebServer-WebServer）の連携機構の研究開発
(ア) 各システムのデータベースにおいて一意性を保つようにデータベースの拡張開発を行なった。
(イ) 支配下サーバ設定の各システムにおいて一意性を保つように支配下サーバ設定方式の拡張開発を行なった。

カ　既存（非 RCM）社内 R&D システムとの連携機構の研究開発
(ア) 任意の通信プロトコルを受信し、応答するためのブリッジモジュール機構を開発し、既存システムの通信プロトコルに合わせ専用ブリッジモジュールを開発し、Controller に簡単に追加できる機構を試作開発した。
(イ) (ア)の試作の機能や性能を評価し、複数の RCMBridge が同時に起動し、いずれも干渉を受けずに、独立して稼動し、既存システムおよび WorkFlow と連携できることを確認した。

キ　実証システムの構築と運用の研究開発
(ア) 現状の RCM を上記商用 OS、商用 Web ミドル（WebSphere など）で動作試験を行なった。また、機能テストだけではなく、性能面の評価も行なった。その結果、商用 Web ミドルに関しては、現状の RCM そのままでは、動作できないことが判明した。ただし、修正は大規模ではないが、商用 Web ミドルそれぞれ特性があり、統合的に同じ修正で RCM システムを稼働させることが困難であることが分かった。
　また、「RCM システムの負荷分散、冗長機構の研究開発」において、非商用 Web ミドルの Tomcat は、機能面で遜色がなく、性能面でも大きな低減が見当たらなかった。

(イ) 性能、機能面で利用上問題がある部分に関して、ボトルネックポイントを同定し、改善プランを策定した。

今後は平成 22 年度に残りの全ての機構の開発を終了し、平成 23 年度には実証システムの運用を始めるための基盤を整備する予定である。

## ５　参考資料

### ５－１　研究発表・講演等一覧

＜一般口頭発表＞

上島豊: 「R&D クラウド基盤ミドルウエア事業」、平成 21 年度情報通信ベンチャービジネスプラン発表会、平成 22 年 1 月 22 日

### ５－２　産業財産権

該当なし