管理番号＃２１－０３

平成２３年度
民間基盤技術研究促進制度 成果報告書

PaaS-CAE 基盤技術に関する研究開発

委託先：　㈱キャトルアイ・サイエンス

平成２３年１１月

平成２３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 3

２　研究開発の全体計画
2-1　研究開発課題の概要 4
2-2　研究開発の最終目標 6
2-3　研究開発の年度別計画 9

３　研究開発体制
3-1　研究開発実施体制 10

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 11
4-1-1　研究開発内容 11
4-1-2　実施状況 11
4-1-3　達成状況及び今後の課題 14
4-2　RCM システムの負荷分散、冗長機構の研究開発 14
4-2-1　研究開発内容 14
4-2-2　実施状況 15
4-2-3　達成状況及び今後の課題 18
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 18
4-3-1　研究開発内容 18
4-3-2　実施状況 18
4-3-3　達成状況及び今後の課題 22
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 22
4-4-1　研究開発内容 22
4-4-2　実施状況 23
4-4-3　達成状況及び今後の課題 29
4-5　RCM システム間（WebServer-WebServer）の連携機構の研究開発 29
4-5-1　研究開発内容 29
4-5-2　実施状況 29
4-5-3　達成状況及び今後の課題 32
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 32
4-6-1　研究開発内容 32
4-6-2　実施状況 32
4-6-3　達成状況及び今後の課題 35
4-7　実証システムの構築と運用の研究開発 35
4-7-1　研究開発内容 35

4-7-2　実施状況 36
4-7-3　達成状況及び今後の課題 41
4-8　総括 41

5　参考資料・参考文献
5-1　研究発表・講演等一覧 44

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、その正確性及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及び進捗に関し、様々な人（特に、経営層や研究開発責任者）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。

本開発では、現状方式に比べ通信ルートが複雑となるため単純な実装ではインタラクティブ性の性能劣化が予想される。本開発では、機構開発とともに性能向上も大きな課題になるため、この機構開発に限って、本研究開発の方針を破り、リモートアプリケーションサーバに通信性能向上のための専用モジュールを追加する可能性もある。

(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、任意のデータ及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 及び入力系 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- Control 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

2.1RCM-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。

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 任意のデータ及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、固定的な複数選択肢は、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

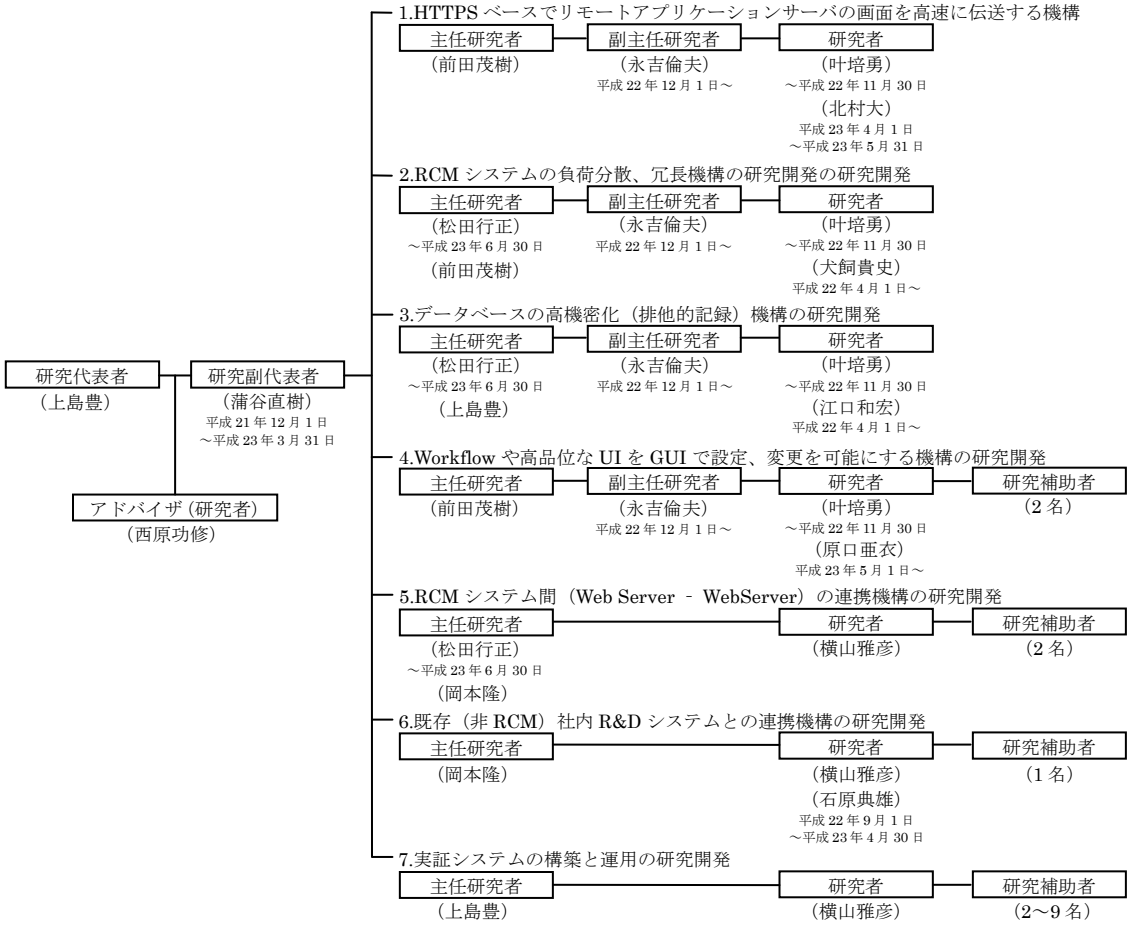
金額は非公表

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 (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	－	－	－	－	

注）１　経費は研究開発項目毎に消費税を含めた額で計上。また、間接経費は直接経費の３０％を上限として計上（消費税を含む。）。
　　２　備考欄に再委託先機関名を記載
　　３　年度の欄は研究開発期間の当初年度から記載。

3　研究開発体制

3－1　研究開発実施体制



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

H21 年度

(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、選定及び導入を行った。

(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する。評価自体は、H22 年度前半も継続調査とした。

H22 年度

(4) 性能評価及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した。

(5) 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）を確保した。

H23 年度

(6) 複数の利用者が本機構を利用しているときの画面伝送性能を安定的かつ実用的な速度で転送できるように、性能の向上と評価を実施した。

４－１－２　実施状況

(1)、(4) に関して、

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開発を行った。

実施内容 1

図 1 に示す通り、アプリケーションサーバ上の GUI を、HTTPS ベースでクライアント PC に伝送する機構を開発した。一般的に、RCM-Web マシンへは、インターネット回線の外部から接続できるが、アプリケーションサーバは、インターネット回線の外部からは直接接続できないように Firewall などでブロックされている。本機構により、クライアント PC にアプリケーションサーバの画面をインターネット回線を通じて転送することをが可能となった。

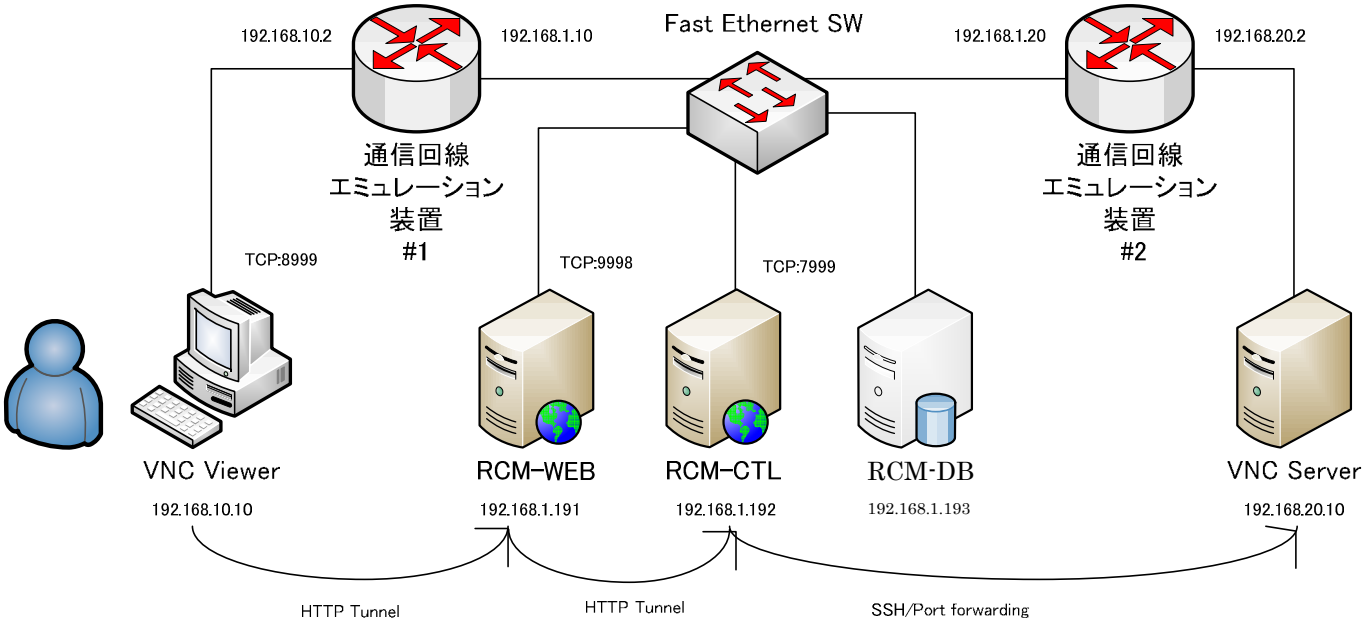


図 1 HTTPS ベースの画面伝送路

具体的な通信のシーケンスを図 2 に示す。

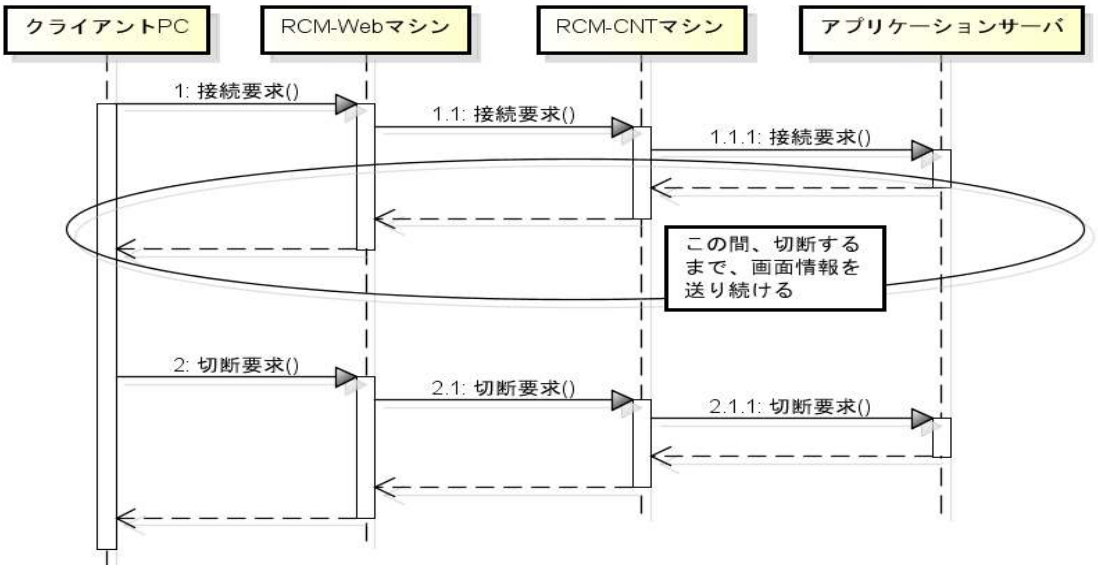


図 2 VNCoverHTTPS のシーケンス

図 2 の通信の流れに示す通り、クライアント PC は RCM-Web マシンのみと通信を行いつつ、Firewall 内部のアプリケーションサーバの画面データを受信することができている。

(3)、(5)に関して、

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）の性能の計測を行い、実測値で 50.3Mbps(RTT=1ms の場合)である事を確認した。また、開発中に計測した本機構の性能は、実測値で 29.87Mbps（RTT=1ms の場合）である事を確認した。BPS と FPS の関係は、1 フレームのデータ通信量を Y bit とすると X bps=X/Y fps と変換することがで

きる。つまり、同じデータ通信量においては、FPS は、BPS に比例する。したがって、開発した機構では、FPS 性能は、29.87/50.3=0.59 倍となっており、当初目標の 0.2 倍（1/5 以内)のインタラクティブ性能（FPS）より高い性能を確保できたことを確認した。

実施内容 1

画面伝送を実用的な応答速度で操作するための性能評価を行い、10Mbps あれば「ほぼ問題なく使用できる」また、5Mpbs あれば、「やや難があるが、操作可能」であることを確認した。これを元に、図 1 に示す RCM 上の通信路において、5Mbps 以上の通信速度を確保するため、性能チューニングを行い、表1 の通信速度を得た。

（クライアント PC ： HTTPS ⇔ HTTPS ⇔ SSH ： アプリケーションサーバ ）

表 1 HTTPS ベースの通信速度

設定 RTT(ms)	PLR 0% (Mbps)	PLR 2% (Mbps)	PLR 4% (Mbps)
1 未満	29.87	10.21	4.62
10	12.90	3.24	2.16
20	8.87	2.27	1.38
50	3.78	1.07	0.77
100	2.17	0.69	0.43
200	1.18	0.33	0.30

参考)RTT(Round Trip Time)：信号を発信してから応答が返って来るまでの時間。
PLR(Packet Loss Rate)：パケットロス率。

国内の一般的なインターネット回線では、RTT <= 20ms, PLR 0%の環境を想定することができるため、表から、8.87 ～ 29.87Mbps の通信速度を HTTPS ベースの画面伝送で確保できることが評価できた。すなわち、HTTPS ベースの画面伝送において、「やや難があるが、操作可能」または「ほぼ問題なく使用できる」伝送速度を確保できた。

(2)に関して、

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の選定・導入を行い、「リモートアプリケーションサーバ画面の伝送」での動作確認を行った。

実施内容 1

評価対象となるアプリケーションを選定し、実証環境へ導入を行い、本機構の動作検証を行った。評価対象アプリケーションを表 2 示す。

表 2 評価対象アプリケーション

評価対象アプリケーション
・GLView
・FieldView
・GAMBIT(TGrid)
・AVS/Express
・MicroAVS
・Paraview
・Gnuplot

(6)に関して、

複数の利用者が本機構を利用しているときの画面伝送性能を安定的かつ実用的な速度で転送できるように、性能の向上と評価を実施した。

実施内容１

複数の利用者が本機構を利用しているときの画面伝送性能を安定的かつ実用的な速度で転送できるためには、画面伝送の比較的トラフィック量の多い通信を効率的に伝送する必要性があり、伝送路の論理多重化による実装として、HTTP（S）をベースとした通信プロトコルを採用し、Firewall 透過性を持たせつつ、動作性能を従来と同等水準にするためには、クライアント-サーバ間、各サーバ間の通信部分において従来方式から変更を加え、複数の中継サーバ経由によって、トランスポートのためのプロトコルオーバーヘッド等を含むことを前提に、実装を行った。

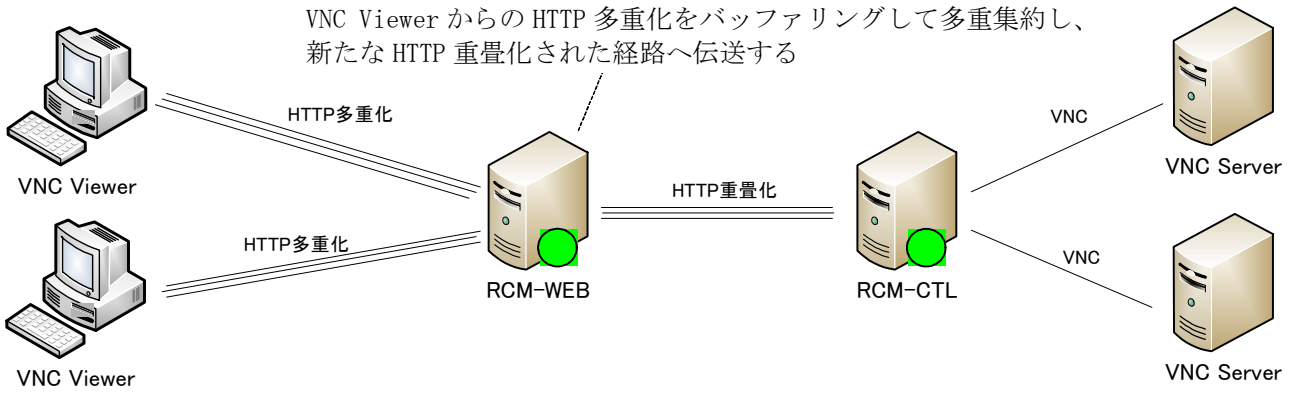


図 3 VNCoverHTTPS の多重化中継方式

４－１－３　達成状況及び今後の課題

平成 23 年度 10 月末までに、当初予定していた開発は終了した。今後の課題として、より快適な操作性を実現する為の性能向上と、インタラクティブアプリケーションとの親和性向上、三次元グラフィックを多用するアプリケーションでの画面伝送効率の向上があげられる。

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

H21 年度

(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。

H22 年度
(5) RCM-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機構を開発し、基礎性能評価を実施した。
(6) RCM-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機構の開発・評価を行った。
(7) ハードウェアで本機構を実現できることを実機を用いて検証し、その基礎性能を評価した。
H23 年度
(8) 複数のファイルサーバに格納される実ファイルに関しての負荷分散、冗長機構の開発を行った。

４－２－２　実施状況

(1)に関して、
アプリケーションサーバについて機能比較を行い、本開発を実施するに当たり、現行の Jakarta Tomcat6 が一番有利であることを明確にした。

実施内容１
以下のアプリケーションサーバにおいて、機能レベルでは、Jakarta Tomcat6 と JBoss EAP5 が RCM の PaaS 化に一番有利であることが分かった。また、性能面及び価格面での比較を含め、最終的には、Jakarta Tomcat6 が一番有利であると判断した。

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

(2)、(3)、(4)に関して、
一時ファイルの冗長化機構を実際に試作し、評価を行った。

実施内容１
以下の３種類の一時ファイルの冗長化機構を検討し、実際に試作し、評価を行い、ソフトウェアレベルの冗長化機構では、試作２が一番性能が高いことが分かった。

(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。

(5)、(6)、(7)に関して、
ハードウェア(ロードバランサ及び FT サーバ)による負荷分散、冗長機構を実装し、実機を用いて検証し、その基礎性能を評価した。

実施内容 1

ハードウェアでの負荷分散機構を組み込むためにRCMシステムに、稼働中の系列を一意に示す機構を開発し、ロードバランサを通じて、負荷を分散できることを評価した。その結果、正常に負荷分散がなされていることを確認し、ロードバランス機能による負荷分散機構の開発を達成した。RCM-DB サーバに関しては、一貫性が重要であるためソフトウェア的に負荷分散機構を開発し、本部分に関しては、ソフトウェアによる負荷分散機構が有効に働くことを確認した。

表 3 評価に使用したロードバランサ

Hardwear	Vender	Spec
IPCOM EX 1200 LB	Fujitsu Co.,Ltd.	FW:E10L51 NF0201 B03

RCM-Web, RCM-CNT は、Workflow を実行する際に、RCM-ParallelDB にトランザクション番号と呼ばれる一連の処理番号を渡すことがあり、系列 1 の(RCM-Web, RCM-CNT)と 系列 2 の(RCM-Web, RCM-CNT)が重複したトランザクション番号を発行しないように、系列ごとにユニークなトランザクション番号を発行する機構を実装した。

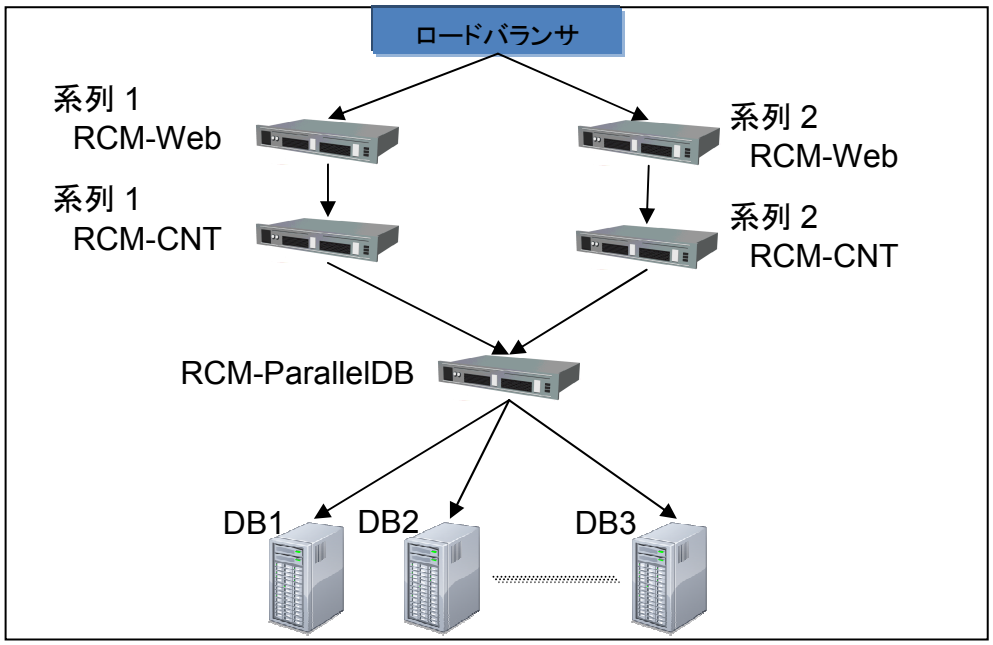


図 4 ロードバランサによる負荷分散

実施内容 2

ハードウェアでの冗長機構を組み込むためにRCMシステムに障害が発生した場合は、ロードバランサが障害を検知できるように、死活監視機構をRCMに開発した。これにより、障害が発生したRCMシステムはロードバランサから、命令を転送されなくなり、他の、稼働しているRCM-Web サーバで処理を継続することを評価した。

RCM システムは、系列ごとに、ロードバランサからの死活監視要求に対して、応答を返す。このとき、RCM-Web または RCM-CNT に障害があれば、死活監視要求に応答しない、または障害通知を返信することで、ロードバランサから切り離される。

(例) 系列 1 の RCM-CNT に障害が発生した場合 RCM-Web は、ロードバランサからの死活監視要求に対して、障害通知を返信する。ロードバランサは、障害通知を受けた系列のRCMには、それ以降、要求を転送せず、系列2の RCMにのみ、要求を通知する。

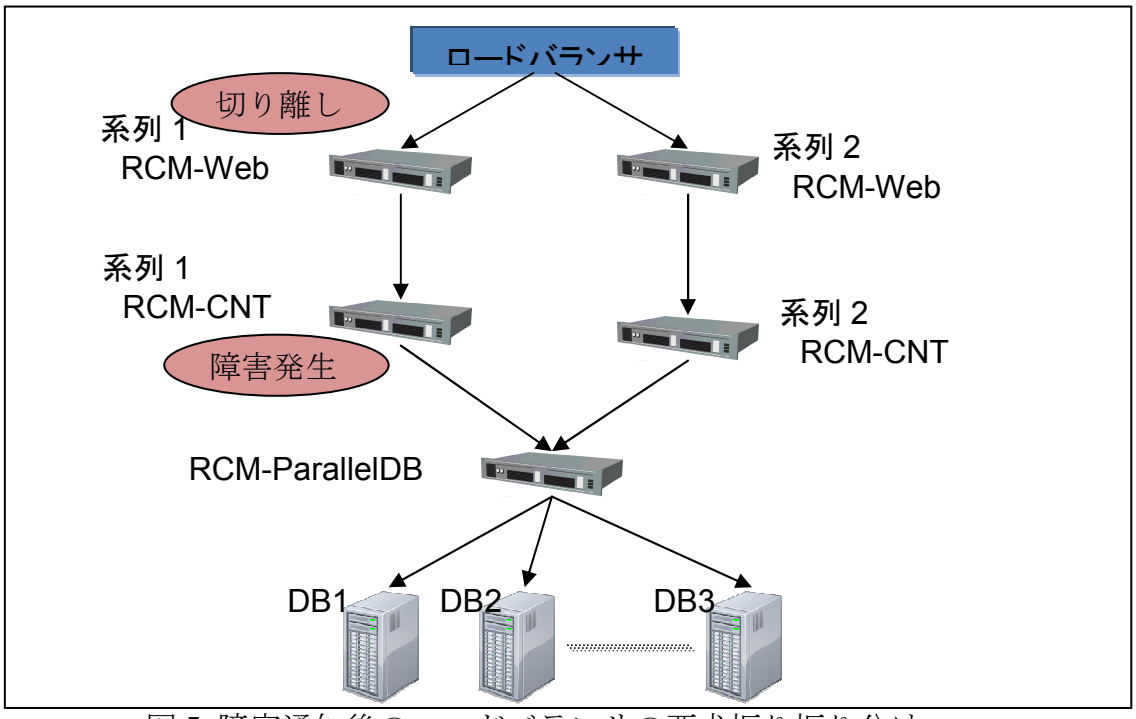


図 5 障害通知後のロードバランサの要求振り振り分け

(8) に関して、

複数のファイルサーバに格納される実ファイルに関しての負荷分散、冗長化機構を実装し、評価を行った。

実施内容 1

複数のファイルサーバに負荷分散的にファイルを格納させる機構を開発することで、ファイルサーバの負荷分散及び書き込みに関しての冗長化機能を実装し、ほぼ性能劣化なしに動作することが確認できた。

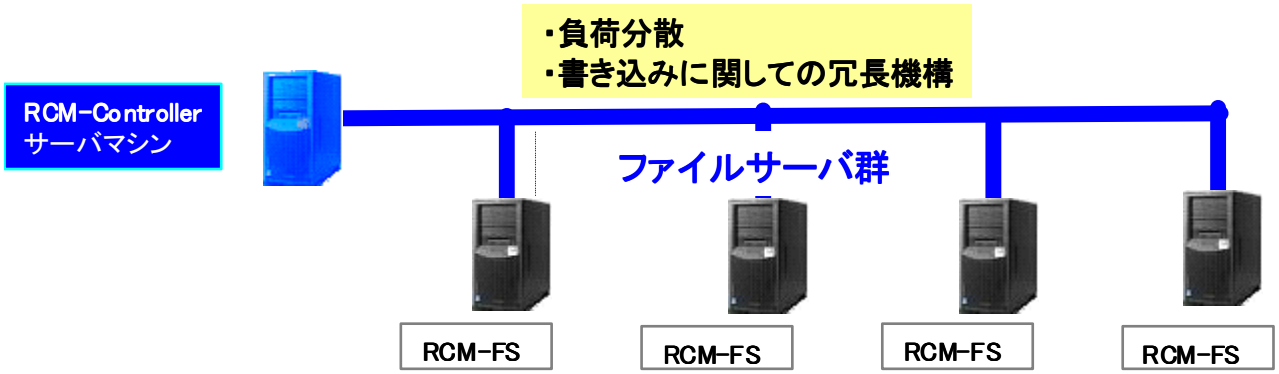


図 6 ファイルサーバの負荷分散と書き込みに関しての冗長化機構

実施内容 2

各ファイルサーバにバックアップサーバを設定、準備することで、ファイルサーバが停止したり、内部のファイルが紛失しても、システムとして停止、エラーすることなくファイルを参照できる冗長機構を実装し、動作を検証し、確認できた。また、本機構において、ファイル破損時には、バックアップサーバからファイルサーバへの自動復旧が行われる機構も実装し、システムの運用負担軽減を図った。

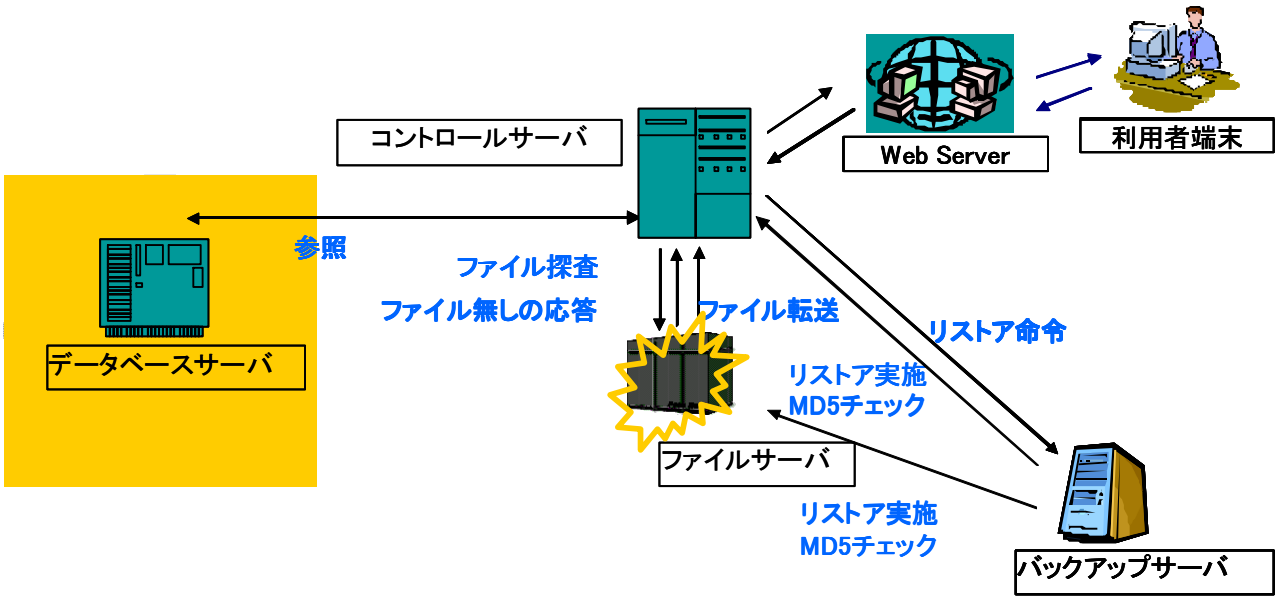


図 7 ファイルサーバの読み込みに関しての冗長化機構及び自動復旧機構

４－２－３　達成状況及び今後の課題

平成 23 年度 10 月末までに、当初予定していた開発は終了した。今後の課題として、動的追加、切り離し時の速度・安定性を向上させ、商用 PaaS サービスとして十分なレベルまでの品質向上を図る。

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

H21 年度

(1) 支配下サーバを動的に追加できる機構を設計し、試作実装する。

(2) 試作機構の基礎性能を評価する。評価自体は、H22 年度前半も継続調査とする。

H22 年度

(3) 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。

(4) 任意のデータ及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。

H23 年度

(5) 初期に設定した機密レベルを一括して変更することができる機構。

４－３－２　実施状況

(1)、(2)、(3)に関して、

支配下サーバを動的に追加できる機構の設計と実装を行った。

実施内容 1

図 8 のように RCM-Controller に支配下サーバを動的に追加できる機構の設計と実装を行い、支配下サーバを動的に追加する事を可能とした。本機構自体は、RCM-Controller の API として実装したので、利用者がこの機構を簡単に操作し利用する為に、操作を容易にするための GUI 開発を行った。

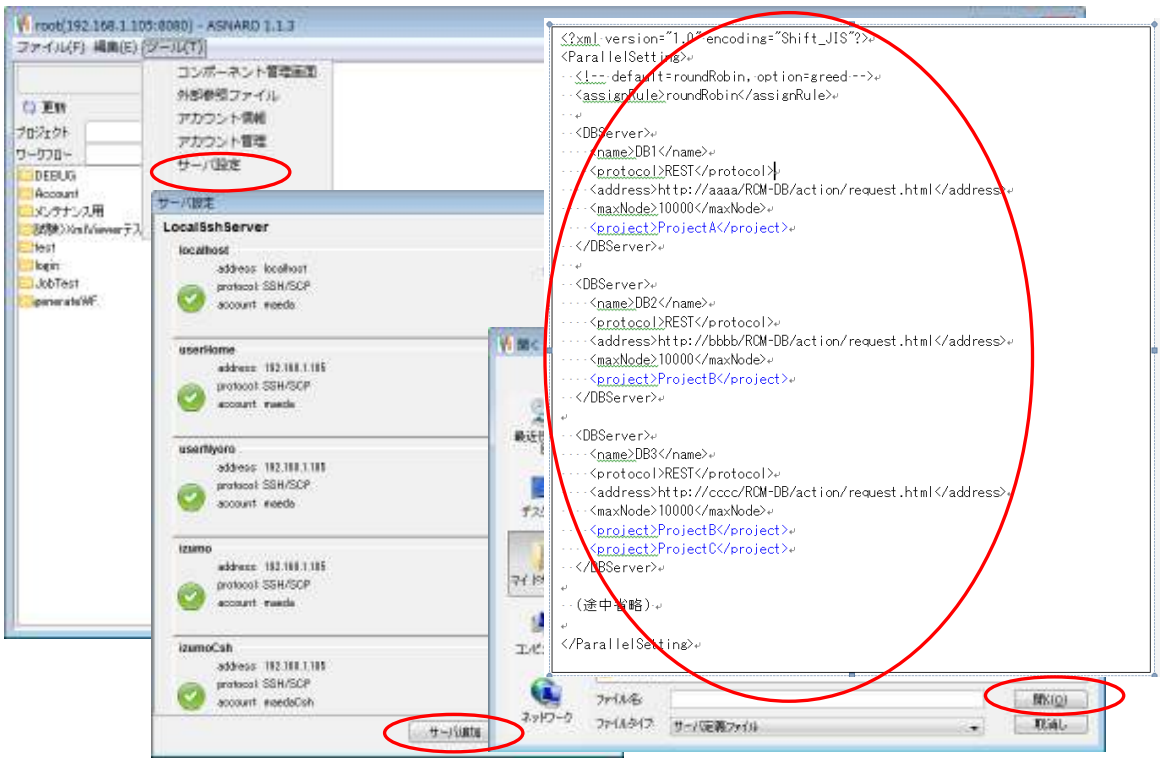


図 8 支配下サーバを動的に追加する操作を容易にするための GUI

(4)に関して、

任意のデータ及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した。

実施内容 1

任意のデータ及びメタデータを格納するハードウェア（データベースマシーン）に関し、図9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。

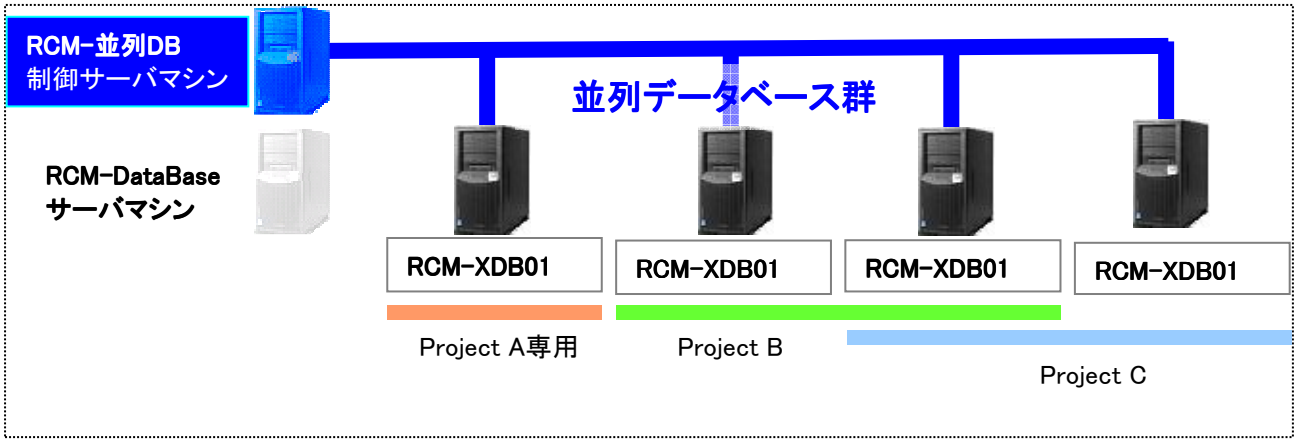


図 9 データベースマシーン構造図

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、図 10 の設定ファイルで管理を行うように拡張した。

```
<?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 10 設定ファイル XML

(5)に関して、
　初期に設定した機密レベルを一括して変更することができる機構を実装した。

実施内容 1
　データベースに登録済みの利用者が所有者になっているタグの 所有者情報や、read/write 権限情報を一括して変更するデータベースリクエスト構文の設計と実装を行った。データベース構文のフォーマットを表 4 に示す。

表 4 所有者情報や、read/write 権限情報を一括して変更するためのリクエスト構文

DBRequest の UPFDATE_AUTH 命令の書式			
No	タグ名	値	意味
1	beforeOwner (必須)	所有者の userid を示す整数	一般ユーザは 自分自身の userid のみ指定できる。 root は、一般ユーザの userid を指定できる。 (注) root の userid は指定できない。
2	afterOwner (必須)	変更後の userid を示す整数	* を指定することもできる。 * は、userid を変更する。
3	beforeAuth (必須)	変更対象になるタグの read,write 権限	read 権限及び write 権限には、a, g, u, subgroup 番号または * を指定する。 * は全ての権限という意味になる。
4	afterAuth (必須)	変更後のタグの read,write 権限	read 権限及び write 権限には、a, g, u, subgroup 番号または * を指定する。 *は、元のタグの read 権限または write 権限を変更しないという意味になる。

本開発で実装したデータベース構文を実際に実行するための、Template を図 11 に示す。

```
<?xml version="1.0" encoding="Shift_JIS"?>
<submitWorkFlow>
  <WorkFlow>
    <status>start</status>
    <dispMode>auto</dispMode>
    <saveJobStdout>false</saveJobStdout>
    <!-- ========================== -->
    <!--   start                    -->
    <!-- ========================== -->
    <job no="0" jobClass="Start">
      <saveJobStdout>false</saveJobStdout>
      <next>update</next>
    </job>
    <!-- ========================== -->
    <!--   UpdateAuth               -->
    <!-- ========================== -->
    <job no="update" jobClass="DBRequest">
      <DBRequest>
        <instruction no="">
          <instkind>UPDATE_AUTH</instkind>
          <instbody inputid="" format="table" cols="2">
            <beforeOwner                                                    inputid=""
caption="beforeOwner">$WORKFLOW_USER_ID</beforeOwner>
            <afterOwner inputid="" caption="afterOwner">*</afterOwner>
            <beforeAuth inputid="" caption="beforeAuth">g,u</beforeAuth>
            <afterAuth inputid="" caption="afterAuth">g,g</afterAuth>
          </instbody>
        </instruction>
      </DBRequest>
      <!-- branch indicator -->
      <next>end</next>
```

```
    </job>
    <!-- ========================== -->
    <!--   end                      -->
    <!-- ========================== -->
    <job no="end" jobClass="End"/>
  </WorkFlow>
</submitWorkFlow>
```

図 11 所有者情報や、read/write 権限情報を一括して変更するための Template

４－３－３　達成状況及び今後の課題

平成 23 年度 10 月末までに、当初予定していた開発は終了した。今後の課題として、支配下サーバの追加だけでなく、変更・削除の機能や、それを実現するために支配下サーバの状態遷移にメンテナンスモードの状態を追加する必要がある。

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

H21 年度

(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う。

(2) (1)の試作の性能を評価する。

(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う。

(4) (3)の試作の性能を評価する。評価自体は、H22 年度前半も継続調査とする。

H22 年度

(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う。

(6) (5)の試作の性能を評価する。

(7) XML-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。

(8) 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

(9) 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。

H23 年度

(10) データ解析ウィザード GUI の自動試験機構を構築する。

4－4－2 実施状況

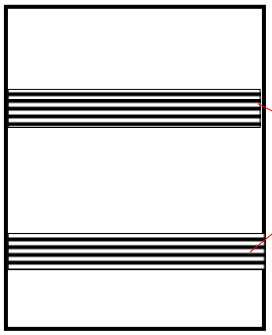
(1)、(2)に関して、

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った。

実施内容1

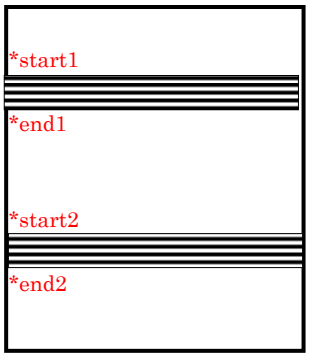
大きなファイル(例えば 1Gbyte のテキストファイル)を Load すると、本試作以前の RCM-Picker は メモリ不足のため、マウス操作に対するレスポンスが極端に遅くなることがあった。本試作では、図 12 のように Load するテキストファイルのうち、処理したいエリアを定義し、関心領域のみをメモリにロードして、表示する拡張を行った。

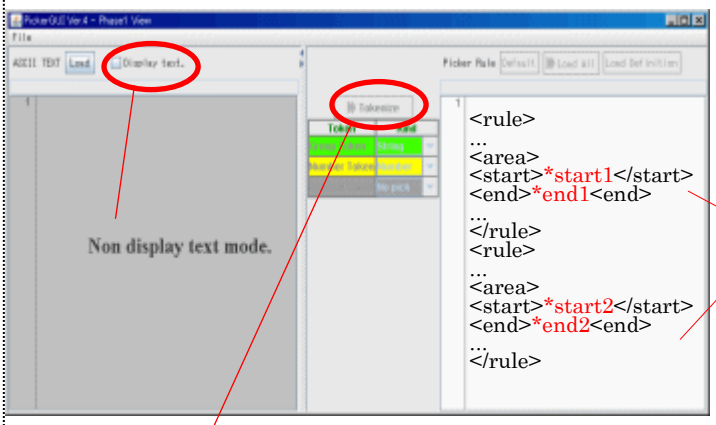
巨大なテキストファイル



ピックしたい雛形 ASCII テキストファイルが巨大で、すべて読み込んで表示するにはメモリに入りきらないが、ピックしたいのは、ごく一部であるときに利用する。

あらかじめ、その範囲を判定できるキーワードで、ピックしたい範囲を特定できるようにしておく。右図では最初の範囲を*start1 から*end1 までのキーワードで、次の範囲を*start2 から*end2 までのキーワードで範囲指定できるものとしている。





Picker の非表示モードで ASCII ファイルのパスを指定する。(このときまだファイルは読み込まれていない。)
次に、範囲ごとに<rule>を作成

「Tokenize」ボタンを押すと Picker はそれぞれの<rule>の<area>で指定された範囲のみ読み込み、それによってピック候補を Phase2 へリストアップする。

次ページへ続く

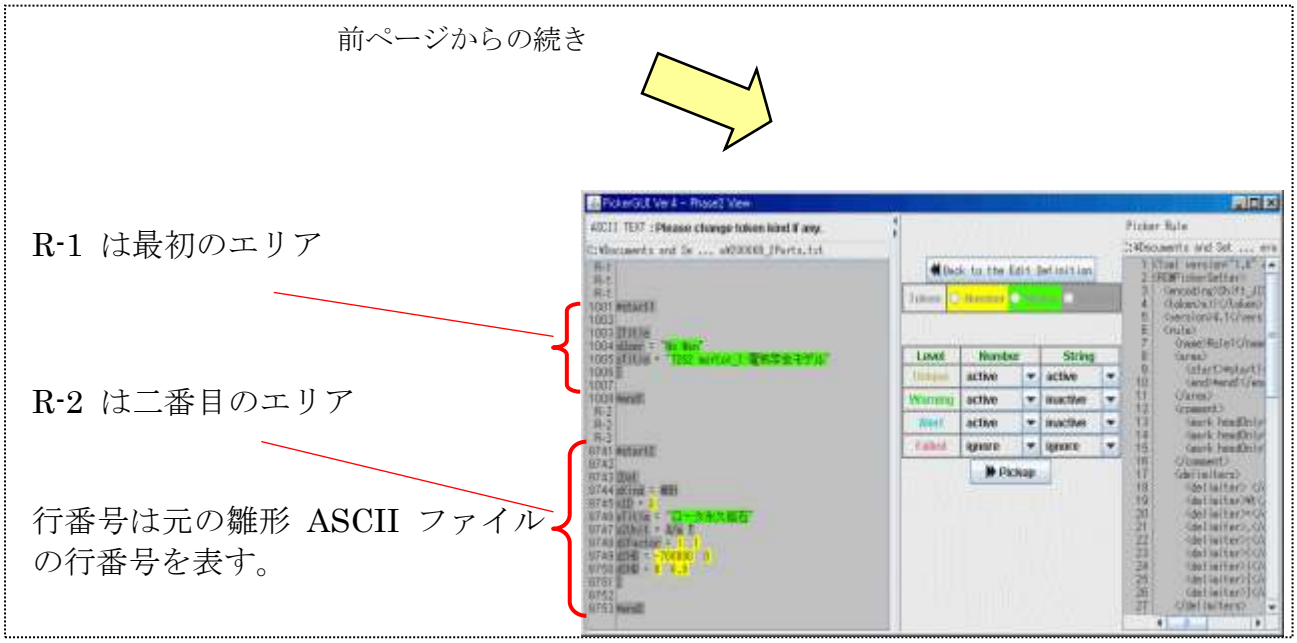


図 12 拡張 RCM-Picker の画面遷移について

(3)、(4)、(5)、(6)、(7)、(8)、(9)に関して、

現在の RCM におけ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の作成、、Workflow のプリミティブを UI の自動生成、ワークフローと、その入力 UI，出力 UI を 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した。これらの機能は、すべて GUI で操作できるようにし、誰でも扱いやすい操作性を実現した。

実施内容 1

データ解析ウィザードのマクロ関連系、スーパーマクロ作成機能を拡張し、図 13、図 14 で示す GUI 画面で新規作成、編集と実行の操作をできるようにした。

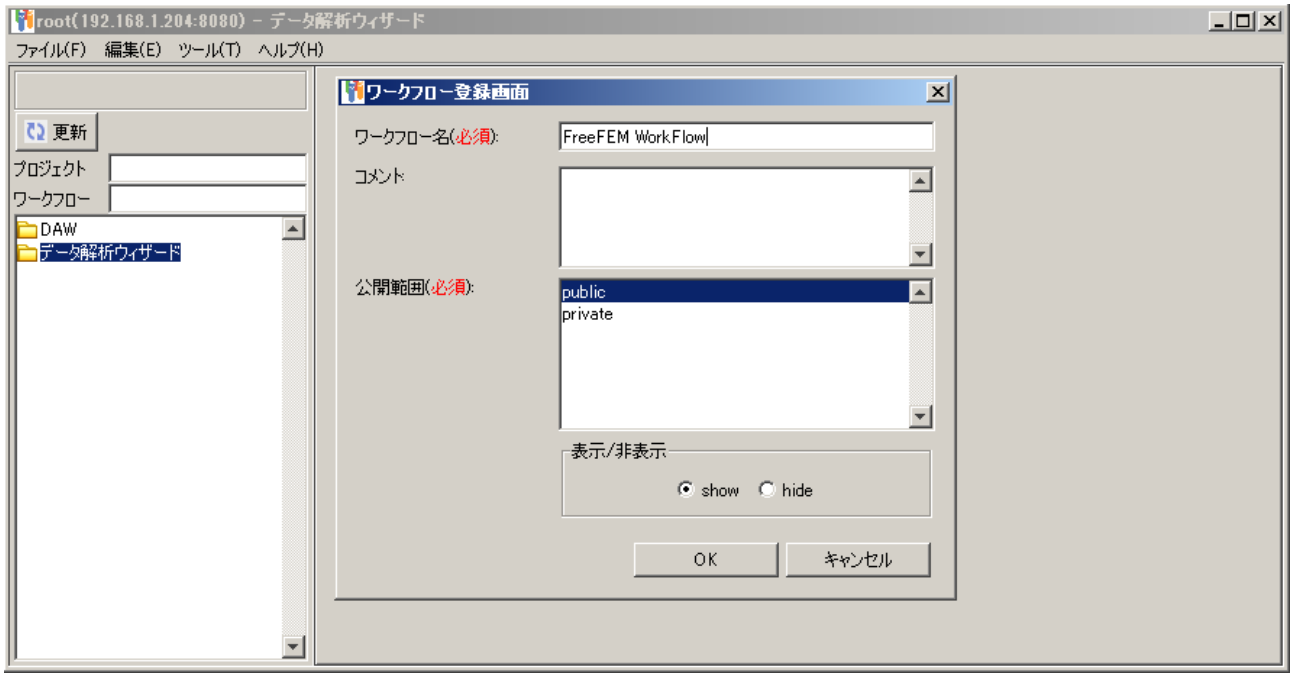


図 13 新規ワークフローの作成

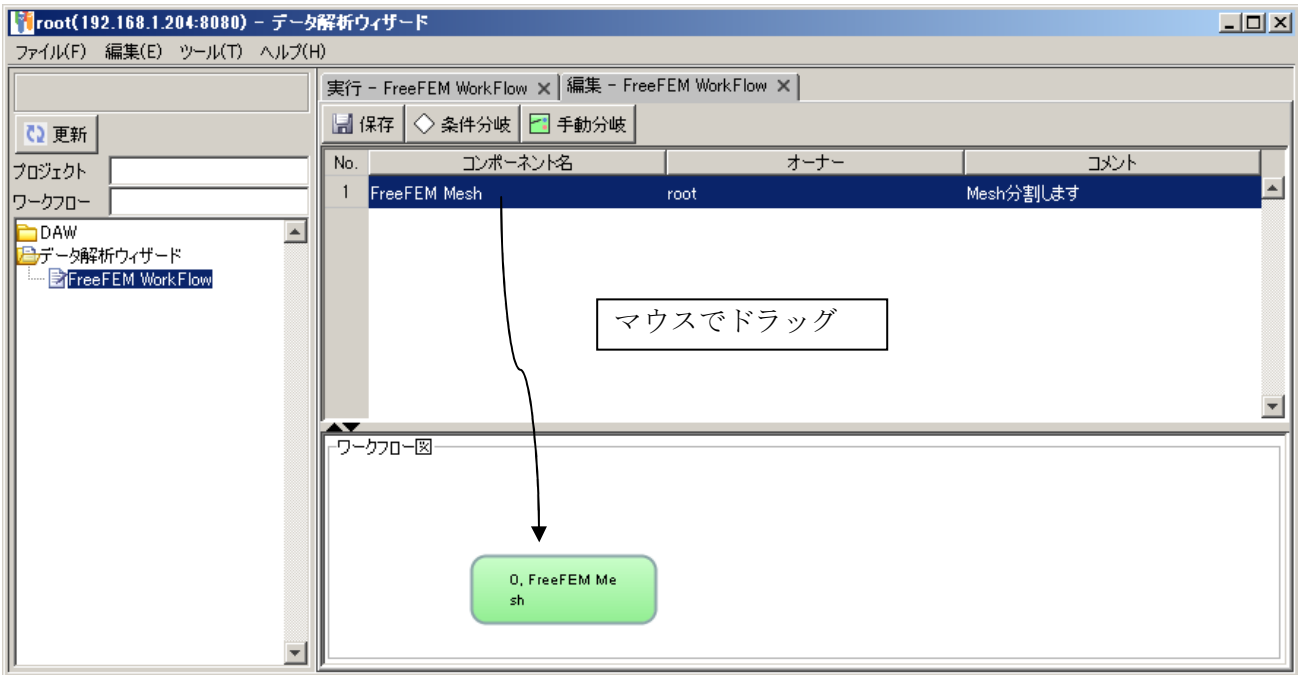


図 14 Workflow をマウス操作で作成する

実施内容 2

GUI 画面におけるパラメータの入力欄を自動的に生成し、XML-Workflow の入力系 UI では、図 15、図 16 の通り、コンポーネント(処理単位)ごとに、XML-Workflow の実行に必要なパラメータを UI 上から入力できるようにした。

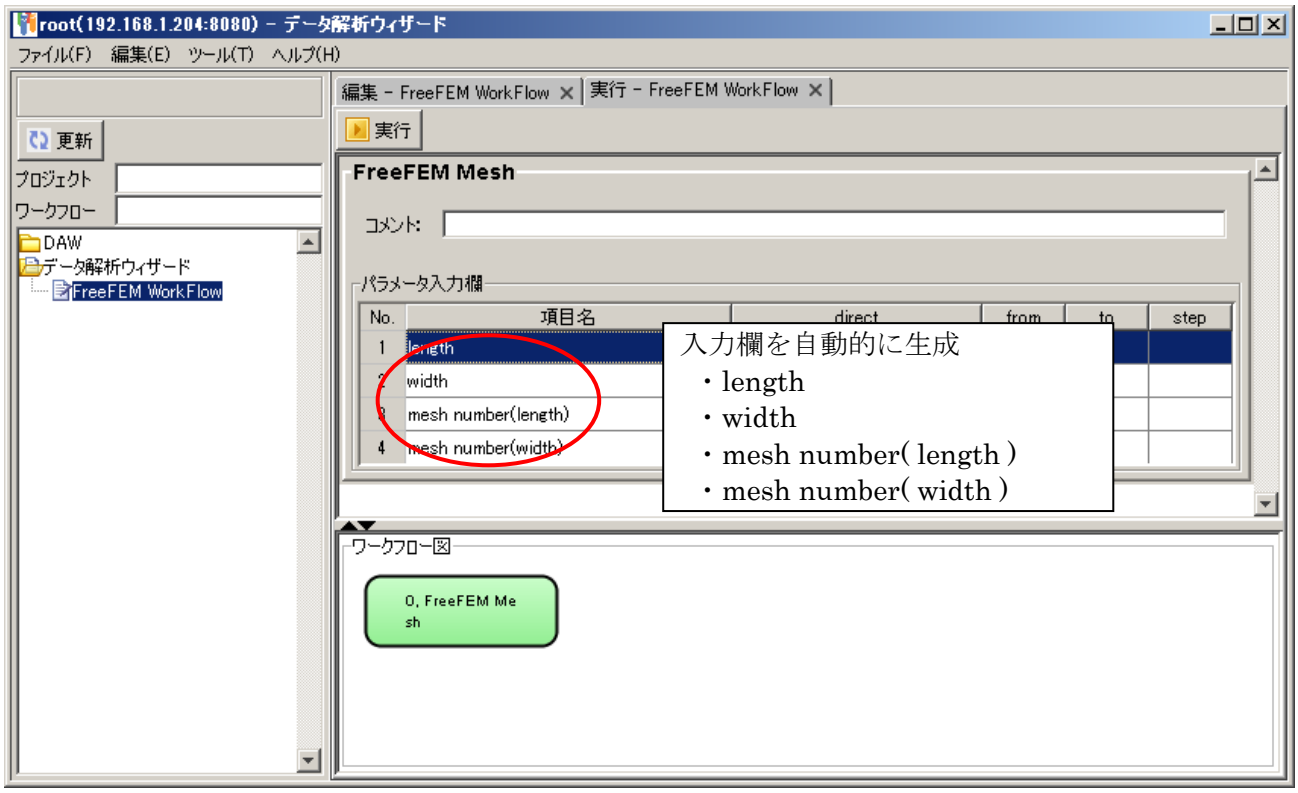


図 15 Workflow の実行画面

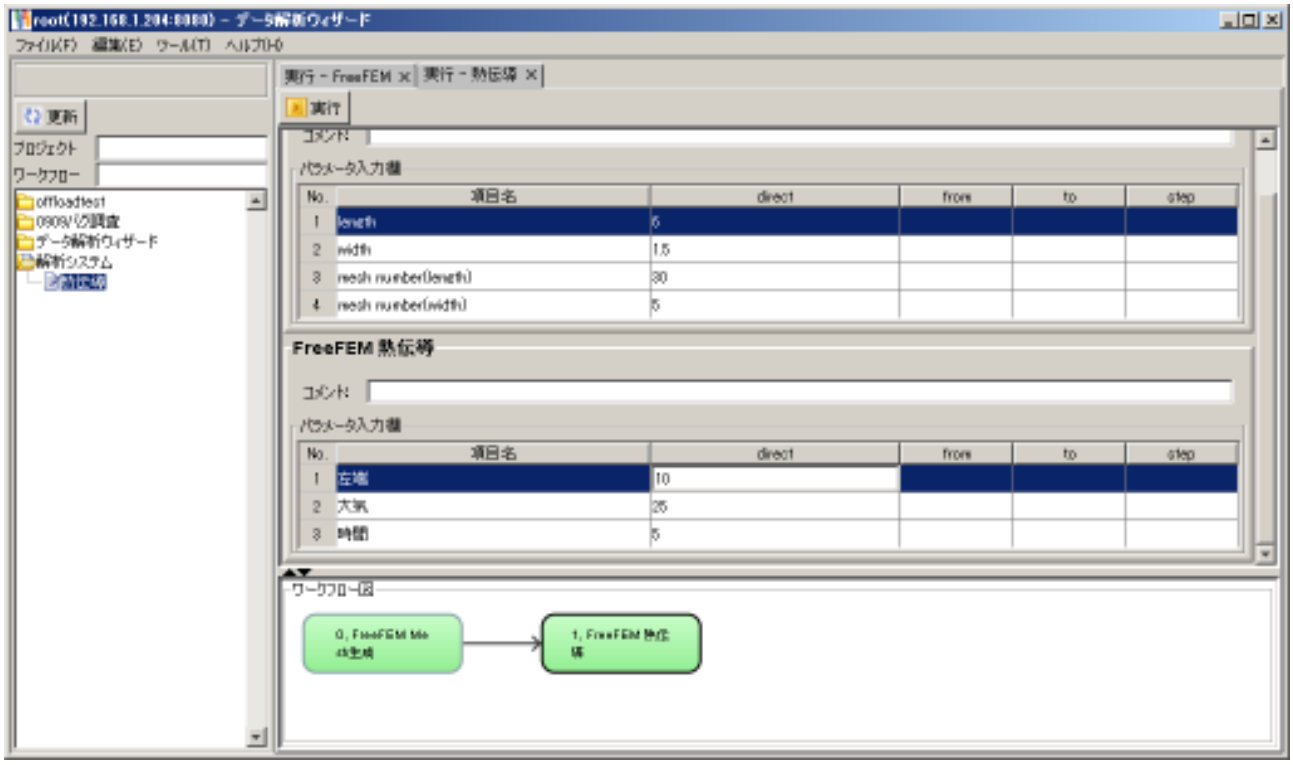


図 16 データ解析ウィザード GUI の入力系 UI での設定画面

実施内容 3

出力系 UI では、登録したデータの表示にあたり、表示する項目を選択し、表示状態をカスタマイズできる機構の開発を行った。

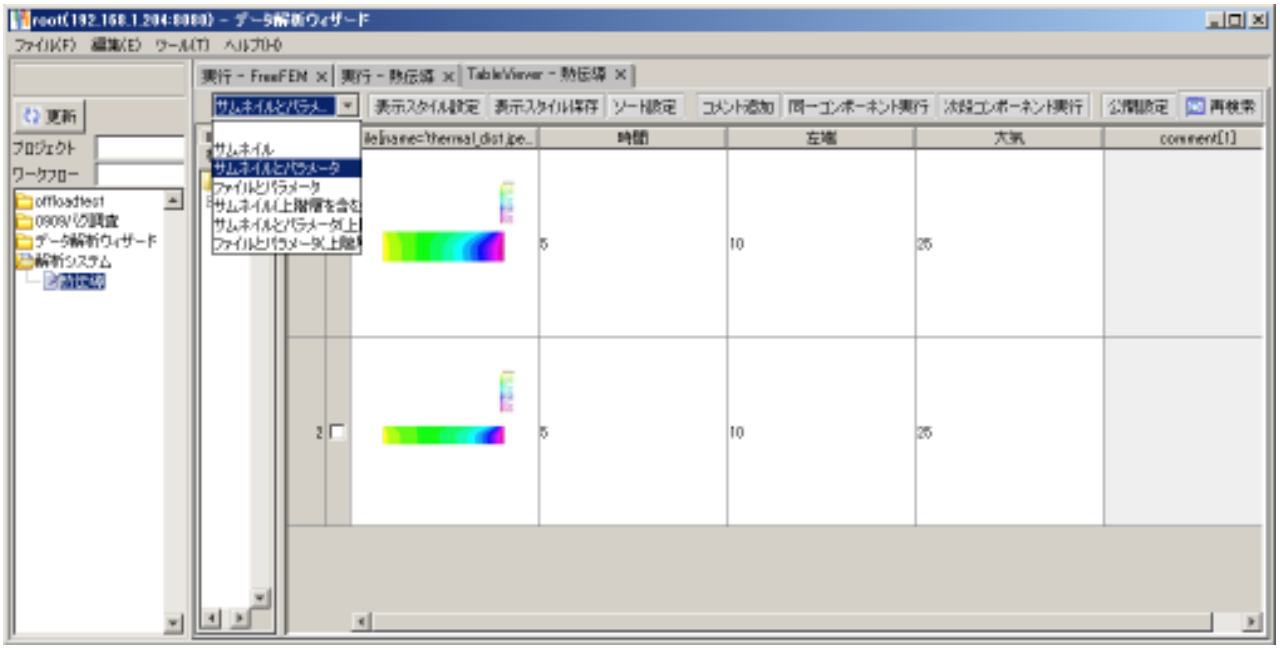


図 17 データ解析ウィザード GUI の出力系 UI での設定画面

(10)に関して、

データ解析ウィザード GUI の動作試験を行う為の自動試験機構を開発し、実際に開発のテストに使用した。

実施内容 1

開発したデータ解析ウィザードGUIの機能の全てを動作試験するためには、多くの操作パターンで実際に操作を行い、正しく動作するかを確認する必要がある。これを全て人手で実施するには膨大な工数が必要となるため、研究開発費を節約する為に、動作試験の自動化を行った。

自動試験システムのベースには、汎用テストツールとして知られているFEST-Swing をベースシステムとして使用し、この上にデータ解析ウィザード GUI のテストシステムの構築を行った。このシステムの試験結果は自動的にWebサーバにアップロードされるため、試験結果は図 18 のようにウエブブラウザで確認することができる。

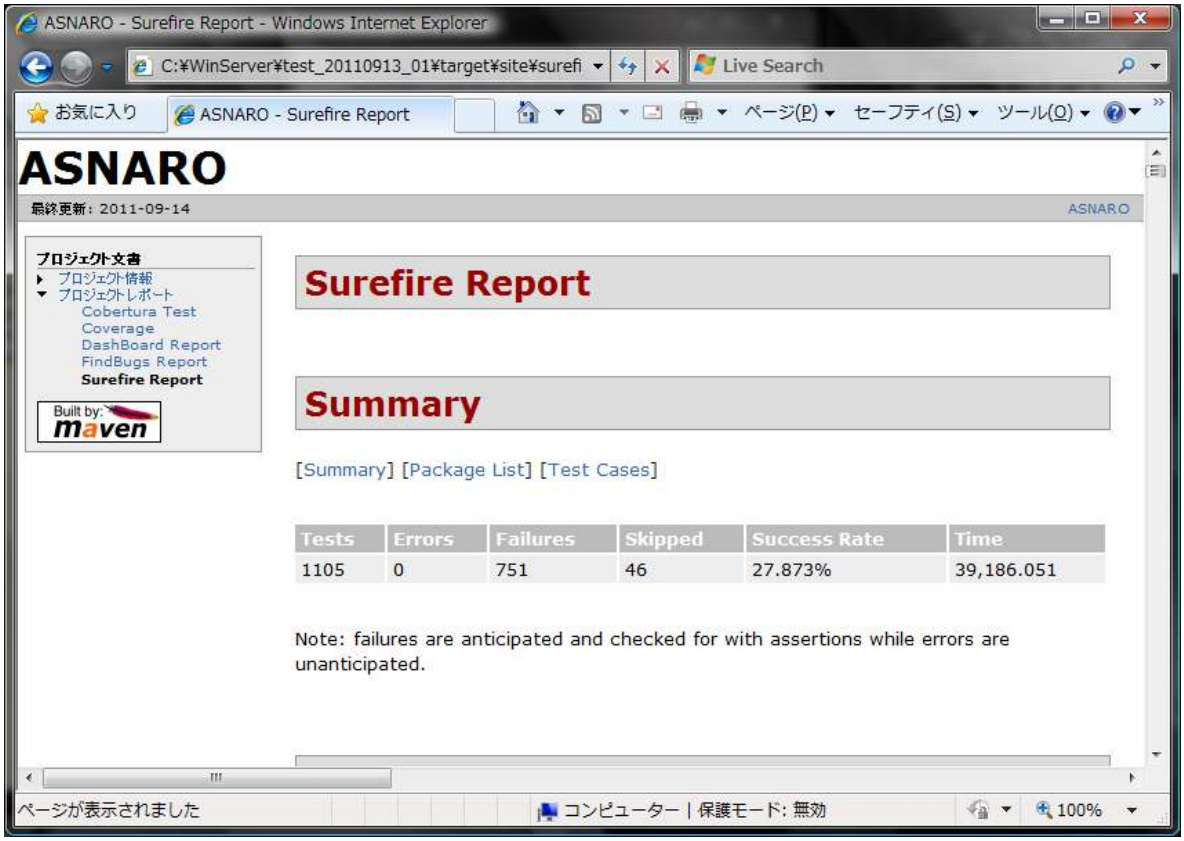


図 18 テストパターン結果表の見方

このシステムにおける試験結果の例を表5に示す。この例は、試験結果に不合格が含まれるものを例示しているが、実際の開発では全ての試験が合格となっている。

表 5 自動試験システムによる試験結果

	Class	Tests	Errors	Failures	Skipped	Success Rate	Time
	AF9Test	6	0	6	0	0%	377.215
	AF5_10Test	8	0	6	0	25%	177.468
	AF1_2Test	6	0	0	0	100%	129.81
	AF4_1Test	15	0	1	0	93.333%	230.751
	AF2_1Test	5	0	2	0	60%	11.24
	AF3_4Test	136	0	58	0	57.353%	7,982.968
	AF7_1Test	14	0	2	0	85.714%	229.352
	AF7_7Test	20	0	15	0	25%	272.3
	AF12_1Test	38	0	0	0	100%	401.482
	AF5_8Test	8	0	8	0	0%	46.846
	AF5_5Test	30	0	19	0	36.667%	1,570.667
	AF10Test	23	0	19	0	17.391%	869.701
	AF7_8Test	7	0	7	0	0%	45.256
	AF2_2Test	5	0	0	5	0%	120.421
	AF11_1Test	21	0	0	0	100%	127.163
	AF5_1Test	30	0	0	1	96.667%	467.696
	AF7_5Test	23	0	23	0	0%	64.66
	AF1_1Test	8	0	0	8	0%	1,282.219
	AF2_3Test	15	0	3	0	80%	220.551
	AF7_10Test	2	0	0	0	100%	82.27
	AF2_4Test	9	0	0	9	0%	694.529
	AF5_3Test	49	0	44	0	10.204%	3,215.665
	AF7_2Test	22	0	2	0	90.909%	129.852
	AF7_6Test	11	0	11	0	0%	34.971
	AF3_1Test	23	0	0	23	0%	437.132
	AF2_5Test	24	0	0	0	100%	626.841
	AF5_4Test	52	0	52	0	0%	3,369.454
	AF3_2Test	28	0	0	0	100%	166.059
	AF6Test	28	0	9	0	67.857%	183.396
	AF2_6Test	8	0	8	0	0%	14.979
	AF8_1Test	12	0	12	0	0%	37.192
	AF3_3Test	74	0	6	0	91.892%	1,047.34
	AF5_2Test	7	0	1	0	85.714%	4,480.589
	AF8_2Test	12	0	9	0	25%	499.077
	AF5_9Test	143	0	97	0	32.168%	36,340.312
	AF12_2Test	45	0	4	0	91.111%	975.473
	AF11_2Test	35	0	35	0	0%	79.251
	AF7_9Test	22	0	22	0	0%	609.718
	AF7_4Test	16	0	11	0	31.25%	100.656
	AF7_3Test	73	0	0	0	100%	447.147
	AF12_3Test	11	0	11	0	0%	711.649

また、各テストケース毎に、図 19 のように試験結果の詳細表示を行うことも可能であり、これにより試験状況を確認することもできる。

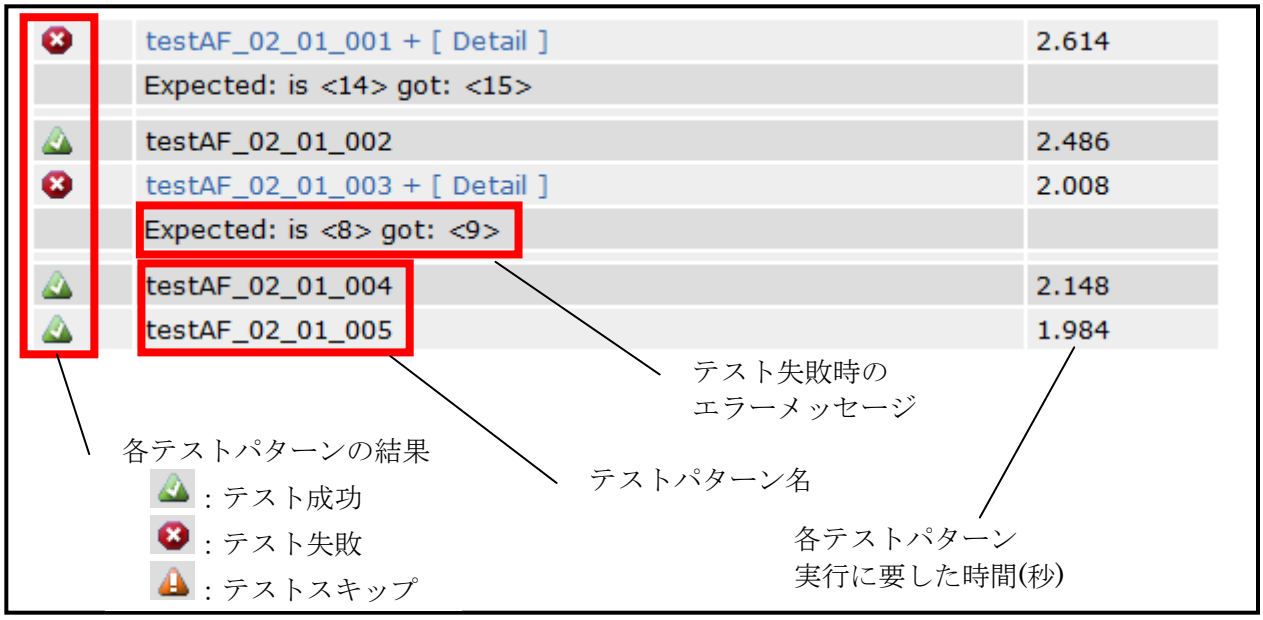


図 19 自動試験システムによる試験結果の詳細表示

４－４－３　達成状況及び今後の課題

平成23年度10月末までに、当初予定していた開発は終了した。今後の課題として、外国人ユーザ向けのメニューやメッセージの多言語対応や、クライアントを閉じてもワークフローを実行し続ける機能、データ管理機能の強化などがあげられる。

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

H21年度

(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。

H22年度

(3) 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、及び、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機能の開発・評価を行った。

H23年度

(4) データ解析ウィザードGUI上でRCMシステム間の連携を実現するための機構拡張の開発と評価を行った。

４－５－２　実施状況

(1)に関して、

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1

本課題で開発したオプションでは、図20のように@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。

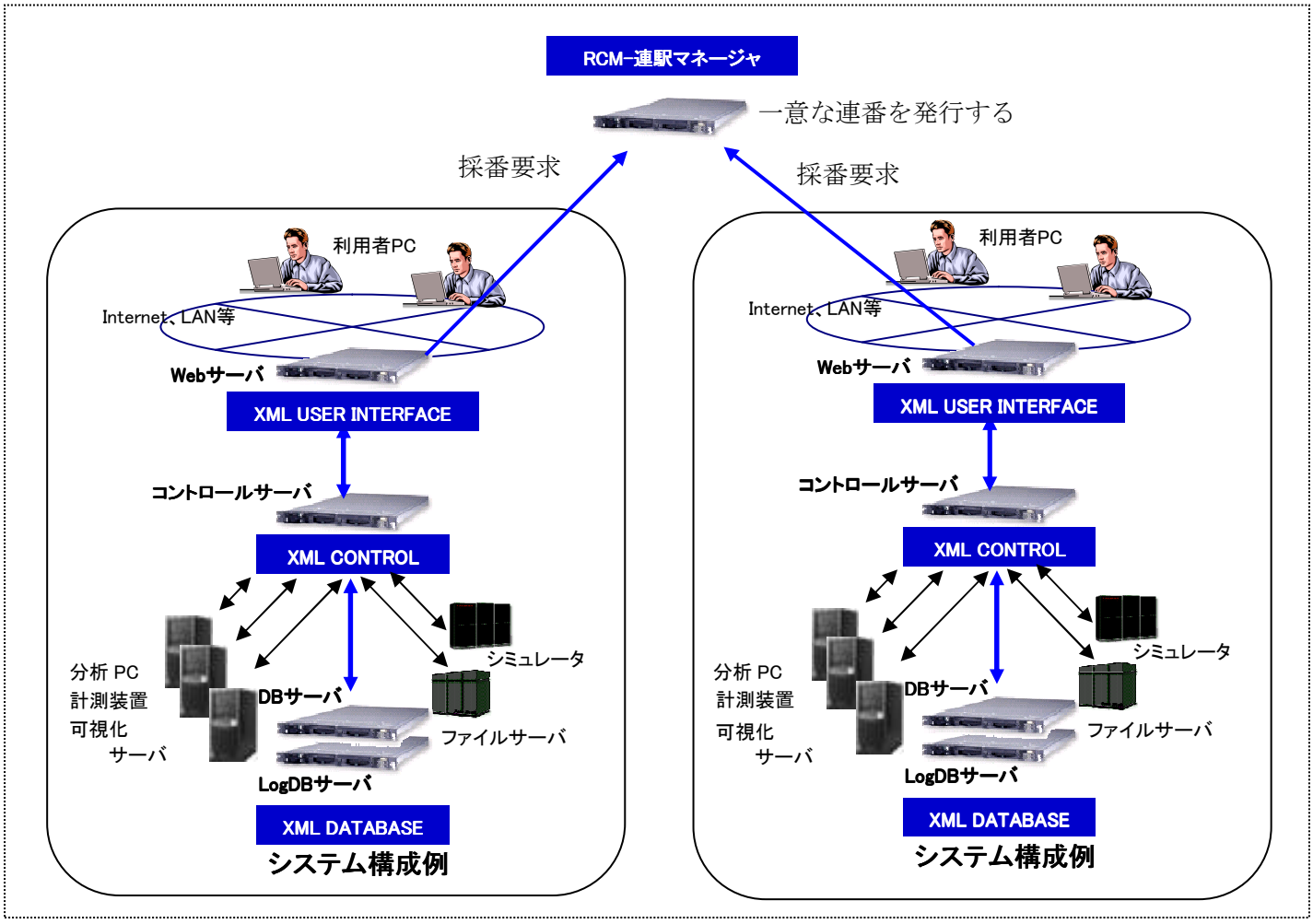


図 20 採番機構の概念図

(2)に関して、

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 1

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。

(3)に関して、

他方の RCM システムに処理を依頼し、その戻値を受け取る機構の開発・評価を行った。また、他方の RCM システムと連携して 1 つの job 機能を果たす機構の開発・評価を行った。

実施内容１

一方の RCM システムから、他方の RCM システムに処理を依頼するための job を新規に定義し、他方の RCM システムと通信を行い、処理結果を受領する機構の開発・評価を行った。その結果、連携した RCM システムが１つの job 機能として動作することを確認し、他方の RCM システムと連携して１つの job 機能を果たす機構の開発を達成した。

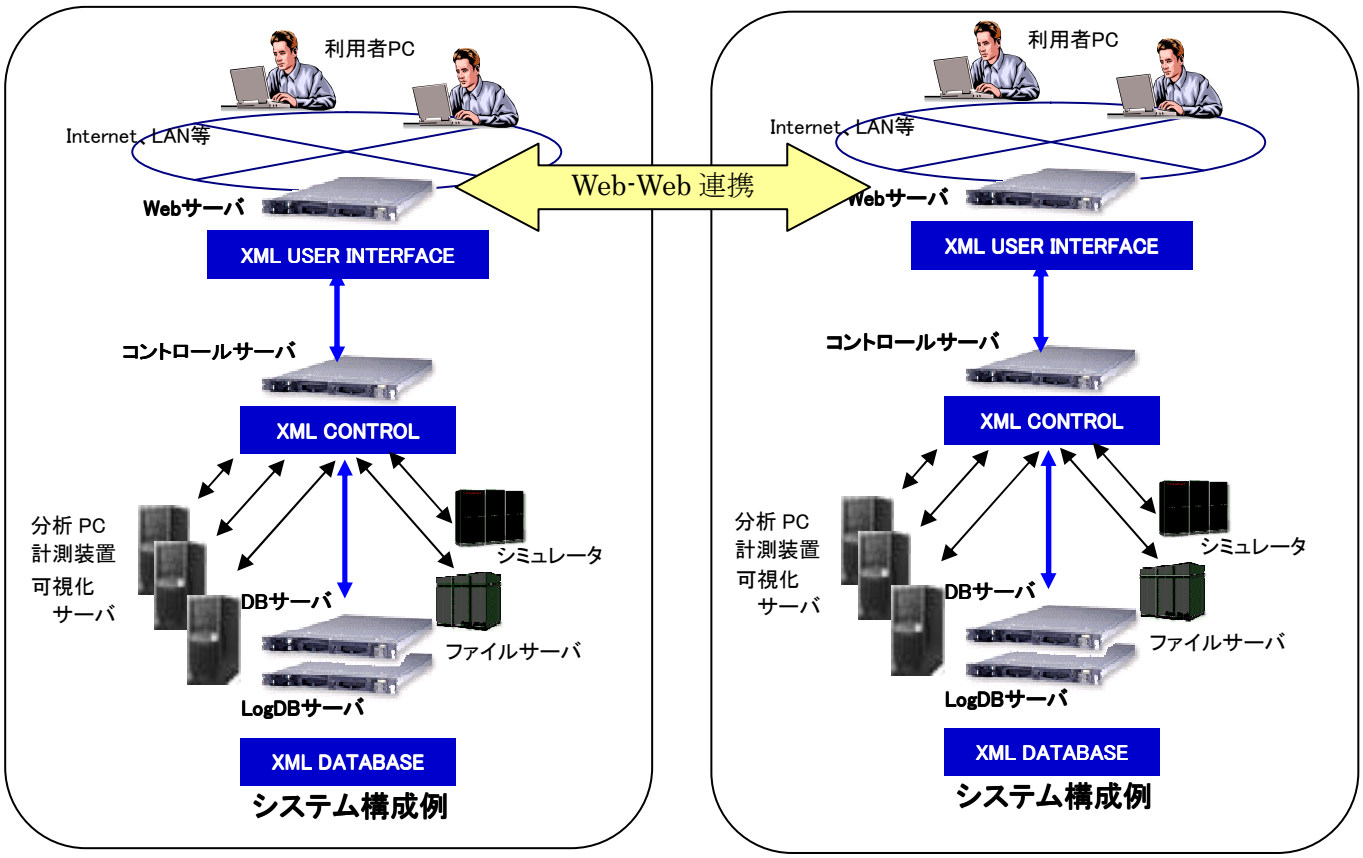


図 21 他方の RCM システムに処理を依頼し、その戻値を受け取る機構

(4)に関して、

データ解析ウィザード GUI 上で RCM システム間の連携を実現するための機構拡張の開発と評価を行った。

実施内容１

開発した GetRemoteWorkFlowFile Job ,GetRemoteWorkFlowList, GetRemoteWorkFlowXML Job を組み合わせ、データ解析ウィザード GUI 上で、RCM システム間連携が可能であることを評価した。

具体的には、呼び出し元の RCM(以下メイン RCM)上のテンプレートからリモート RCM 上の論理サーバを利用するテンプレート「example_useRemoteRCMservers」を作成し、メイン RCM からリモート RCM が管理しているリモートサーバが実行可能であることを確認した。

図 22 は、メイン RCM からリモートサーバへ linux の一般的なコマンドである ls コマンドをコマンド投入した結果であり、正常に ls コマンドの出力結果が表示されていることから、この拡張機能が正常動作することを確認した。

```
合計 12
drwxr-xr-x  2 root root 4096 10 月  2 18:15 .
drwxr-x--- 31 root root 4096 10 月  2 18:15 ..
```

図 22 SimulationServer「sim1」を利用した場合の結果

４－５－３　達成状況及び今後の課題

平成 23 年度 10 月末までに、当初予定していた開発は終了した。今後の課題としては、3 つ以上の RCM がネットワーク上で、スター型や、ツリー型に接続されることを考慮したユーザインタフェースの開発と、安定性の確保がある。

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

H21 年度

(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。

H22 年度

(3) 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の開発・評価を行った。

(4) (3) 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機構の開発・評価を行った。

(5) (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、制御できる機構の開発・評価を行った。

H23 年度

(6) システム連携の実証を行った。具体的には、LDAP 等の実際に利用されている既存システムとの連携を実地で検証した。

４－６－２ 実施状況

(1)に関して、

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。図 23 のように RCMBridge 機構として、RCM に独自プロトコルとの通信口を追加し、既存システム（任意のプロトコル）との連携ができる以下ような仕組みを試作、動作確認をした。

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。

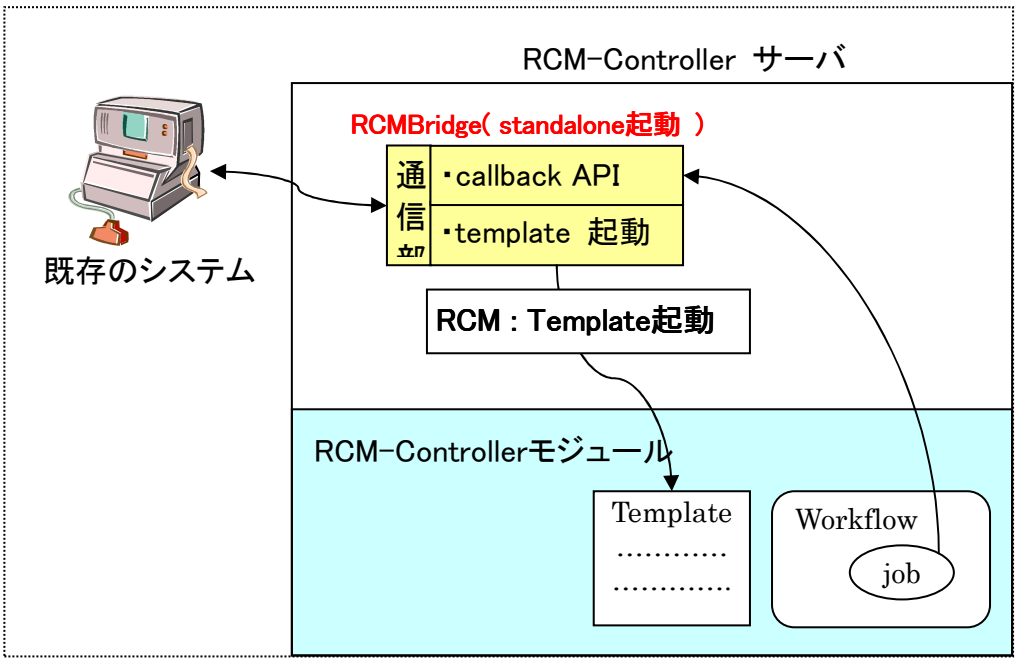


図 23 RCMBridge の API を起動する仕組み

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。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、既存システム及び Workflow と連携できることを確認した。

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、各々が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、図 24 のように 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。

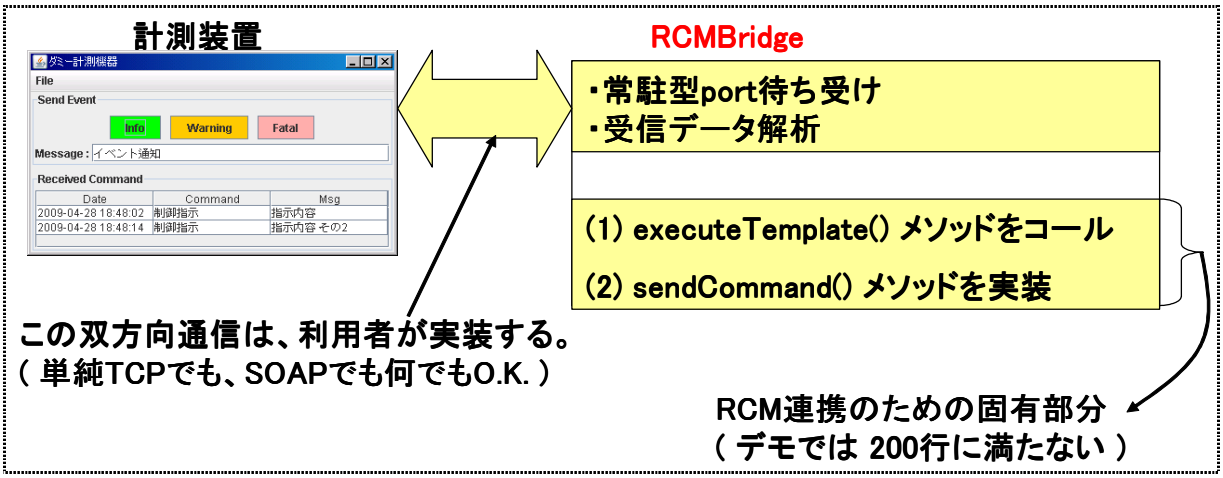


図 24 RCMBridge のコーディング部位

(2)に関して、

実際にサンプルにより機能評価を行い、TCP/IP 通信機能及び 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。

(3)に関して、
　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の開発・評価を行った。

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。このため、RCM では、任意の TCP/IP 通信を待ち受けし、通信を受け付けると RCM の Workflow を起動して、既存システムと RCM 間での通信方法を確立した。

(4)に関して、
　通信プロトコルの中でも、規格化されている SOAP プロトコルに対応し、ESB を使った通信をサポートできる機構の開発と評価を行った。

実施内容 1
　RCM システム内に、SOAP サービスを行うブリッジモジュールを登録し、RCM ブリッジ管理画面から起動した。SOAP サービスが起動されていることは、wsdl 情報を表示して確認することができた。

(5)に関して、
　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、制御できる機構の開発・評価を行った。

実施内容 1
　図 25 のようにブリッジモジュールを RCM 上の管理画面から監視、制御する機構を開発し、RCM 上から、ブリッジモジュールの起動、稼働状態の監視、停止を GUI 画面で操作、表示できることを確認した。

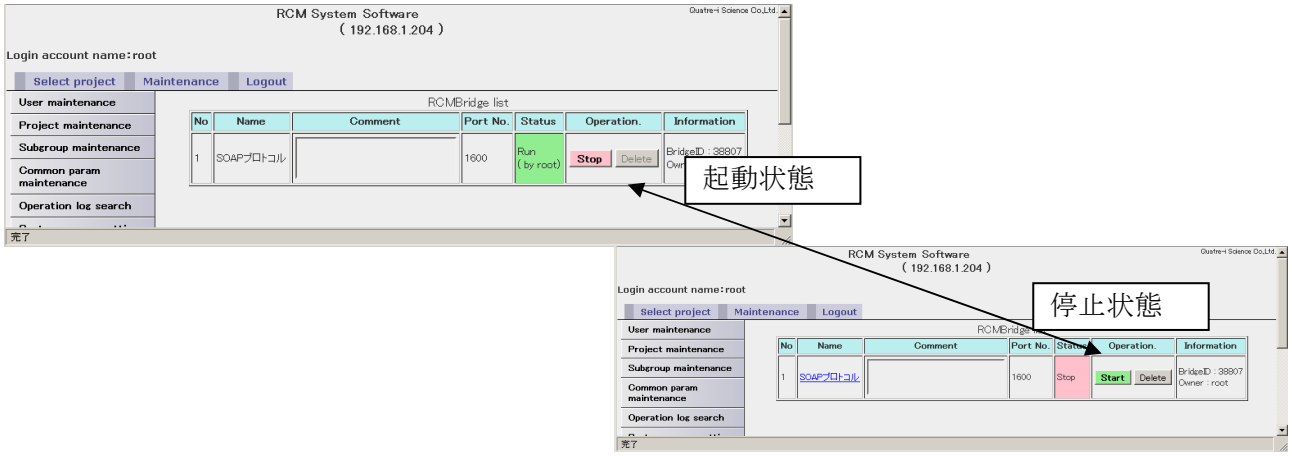


図 25 RCM ブリッジモジュールの管理画面

(6)に関して、

システム連携の実証を行った。具体的には、認証システムである LDAP 既存システムとの連携を実地で検証した。具体的には図 26 のようなシステムを構成し確認を行った。

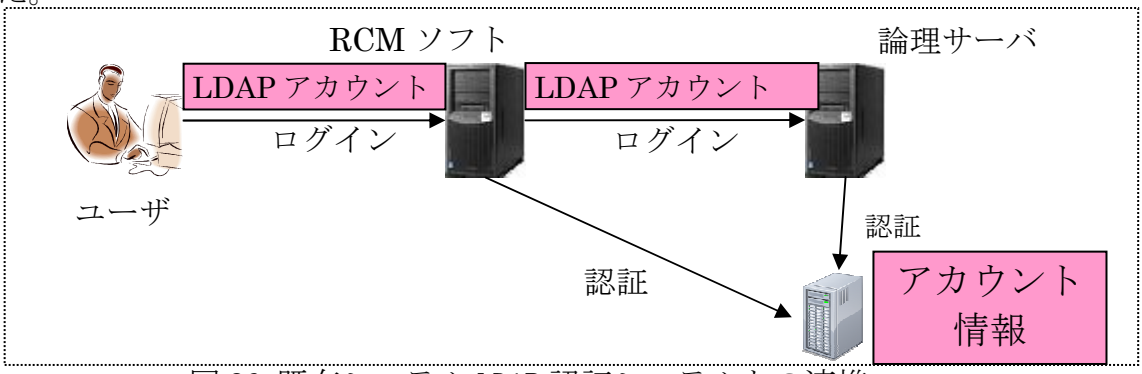


図 26 既存システム LDAP 認証システムとの連携

LDAP の暗号通信方式として、TLS,StartTLS に対応した。また、LDAP ユーザ認証の際に、RCM-Web と RCM-Controller 間で、ユーザが入力したパスワードを平文のまま通信されないように https を用いた暗号化通信を採用した。

４－６－３　達成状況及び今後の課題

平成 23 年度 10 月末までに、当初予定していた開発は終了した。今後の課題として、高速の計測機器から送信されるデータの取りこぼしの抑制や再取得方法の確立がある。

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

H21 年度

(1) RCM を商用 OS、商用 Web ミドルで動作試験を行なった。

(2) 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。

H22 年度

(3) 2 つ以上の RCM システムを別ネットワークで構築、運用し、4 頁記載の(1)～(6)の機能が正しく動作するかを 1 ヶ月以上検証した。

(4) 4 頁記載の(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、改善プランを策定した。

(5) 上記(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た。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した。

H23 年度

(6) 昨年度の実証システムの導入で明らかになったシステム導入マニュアル、システム運用マニュアルの問題点、改善が必要な点を加味したマニュアル改訂を行った。

(7) 作成したシステム導入マニュアル及びシステム運用マニュアルに従い、H23 年度も新規に実証サイトを構築することでマニュアルの評価を行った。

(8) 実証サイトの導入、運用を支援するツールを開発し、より効率的な構築、運用を実現した。

4－7－2 実施状況

(1)、(2)に関して、

商用 OS(Red Hat Enterprise Linux 及び 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 システムの評価を行った。Windows での動作は、可能にしたが、大規模システムへの展開の性能及びコスト面において困難さが明確になったため、実証システムにおける検証は行わないこととした。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、OS は、Red Hat Enterprise Linux 及び既存動作環境の CentOS とし、Web ミドルは、一番中立的な、非商用の Tomcat をベースに実証システムを構築する方針とした。

実施内容 1

RCM を Windows で動作をさせるために修正を行い、動作手順を明確にした。

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 アドレスに設定し直す。

3) RCMFileServer.xml, RCMBackupServer.xml, RCMLogFileServer.xml, RCMLogBackupServer.xml の <workDir> は C:¥RCM¥FileServer などを 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、tomcat を再起動する。

4) Firewall は 8080 ないし、80 を空ける。

実施内容 2

Tomcat と商用 Web ミドル(JBoss, WebLogic, WebSphere)の機能面の比較をした。表 6 は、機能比較の一例であるが、Tomcat と JBoss が機能的に優秀であることがわかる。ただし、JBoss にて RCM を稼働させる為には RCM に若干の修正を加える必要があり、機能的に差が無いのであれば、Tomcat は、RCM をベースとした PaaS 基盤システムのベースを開発するためには、最も優秀であると判断した。

表 6 アプリケーションサーバ機能比較

	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。

(3)、(4)に関して、

実証システムを実際に構築し、長期間の運用を実施し、逐次見つかった問題点を改善していった。

実施内容 1

実証システムで利用可能にするアプリケーションを表 7 のように選定し、ソルバ、メッシャー、可視化などの用途別、商用、非商用別の様々な種類のアプリケーションを準備した。

表 7 実証システムに搭載したアプリケーション一覧

アプリケーション名	種別	商用/非商用
ANSYS	構造系ソルバ	商用
ABAQUS	構造系ソルバ	商用
Marc	構造系ソルバ	商用
NX-Nastran	構造系ソルバ	商用
MSC.Nastran	構造系ソルバ	商用
LS-DYNA	構造系ソルバ	商用
ADVENTURE Cluster	構造系ソルバ	商用
FrontISTR	構造系ソルバ	非商用
FreeFEM++	構造系ソルバ	非商用
ANSYS-CFX5	流体系ソルバ	商用
STAR-CD	流体系ソルバ	商用
STAR-CCM+	流体系ソルバ	商用
FLUENT	流体系ソルバ	商用
FlontFlowBlue	流体系ソルバ	非商用
自作コード	流体系ソルバ	非商用
Patran	メッシャー	商用
FEMAP	メッシャー	商用
HexaGRID	メッシャー	商用
RevoCAP	メッシャー	非商用
FreeFEM++	メッシャー	非商用
GAMBIT	可視化	商用
TGrid	可視化	商用
MicroAVS	可視化	商用
AVS/Express	可視化	商用
LS-PrePost	可視化	非商用
Gnuplot	可視化	非商用
ParaView	可視化	非商用

実施内容 2

合計 6 つの実証サイトを立ち上げ、RCM システムを長期間の連続運用を行った。

実施内容 3

先述の実証サイトにおいて、各種商用アプリケーションと RCM との連携評価を行った。

実施内容 4

先述の立ち上げた個々の実証サイトのうち、複数個の実証サイトを連携してさせ、CAE アプリの統合クラウド化の運用評価を行った。

(5)、(6)、(7)、(8)に関して、

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た。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し、資料の改善を行った。さらに、実際の導入講習会を実施することを想定し、運用管理者用ガイダンス資料、利用者ガイダンス資料及びそれらに必要なサンプルデータを作成した。その資料をベースに導入講習会を実施し、ガイダンス資料に改善を加えていった。

実施内容 1

図 27 のようにインフラヒアリングシートでは、RCM サーバを設置する実証サイトのネットワーク構成などの情報で、必要な情報を事前にヒアリングする。

▪2.1 ネットワーク通信の制限事項に関する確認事項

ネットワーク構成図に記述したネットワークセグメントについて、セグメント間の通信制限についてご記入ください。

表 1 ネットワーク通信制限情報記述例（ネットワークセグメント間）

ネットワークアドレス		L4 プロトコル	ポート番号	L7 プロトコル	通信可否	特記事項
FROM	TO					
192.168.1.0/24	172.146.3.0/24	TCP	80	HTTP	可	
		TCP	443	HTTPS	可	
		TCP	8080	HTTP	否	
		TCP	6000-6999	X11	可	
		TCP	22	SSH	可	
		TCP	21	FTP	可	
		TCP	5900-5999	RDP(VNC)	可	
		TCP	5500-5599	RDP(VNC)	可	
192.168.1.0/24	202.171.156.200/28	全て許可				
192.168.1.0/24	172.10.5.0/24	TCP	587	STMP	可	
		上記以外全て拒否				

図 27 インフラヒアリングシートの例

RCM設置ヒアリングシートは、図28のようにRCMサーバの物理的なマシン情報などを実証サイトのRCM管理者に記入してもらうシートである。RCMの設置に、何か障害がある場合、本シートの情報を参照することで、問題の切り分けが容易になる。

■3　RCMサーバ構成情報

RCMサーバ設置のためのサーバH/W情報と設定内容について、下表にご記入をお願いいたします。

■3.1　RCM-Frontサーバ

表 1 RCM-Frontサーバ基本情報

ホスト名	
MACアドレス(ライセンス発行用)	
IPアドレス	
ネットマスク	
デフォルトGW	
DNS	
NTP	
HTTP Proxy（もしあれば）	
コンピュータ管理者の部門	
コンピュータ管理者の名前	
コンピュータ管理者のE-mail	
製品名	
サイズ	
電源容量（定格）	
重量	

図 28 RCM設置ヒアリングシートの例

RCMサーバ設定・動作確認シートは図29のように、RCMサーバをインストールし、実際に動作しているかをチェックするためのものである。実証サイトのRCM管理者は、本シートの動作テストを一通り実施することで、RCMを正しくインストールできているかを確認することができる。

■表 23 IPアドレス変更後の接続試験状況

論理サーバタイプ	論理サーバ名	正常時接続結果	結果(合格/不合格)	確認日(確認者)	補足(表示されたメッセージなど)
RCMDBServer	DB1	OK			
RCMFileServer	file1	OK			
RCMBackupServer	backup1	OK			
RCMLogDBServer		N/A			
RCMLogFileServer		N/A			
RCM-LogBackupServer		N/A			
DataServer	dataIn	OK			
SimulationServer	sim1	OK			
AnalyzeServer	ana1	OK			
XXXServer	test1	NG			

図 29 RCMサーバ設定・動作確認シートの例

導入講習会を実施することを想定し、図 30 及び図 31 のように管理者用トレーニング資料、利用者トレーニング資料及びそれらに必要なサンプルデータを作成した。その資料をベースに、図 32 のような導入講習会を実施し、ガイダンス資料に改善を加えていった。

図 30 講習会資料の例（管理者向けトレーニング資料）

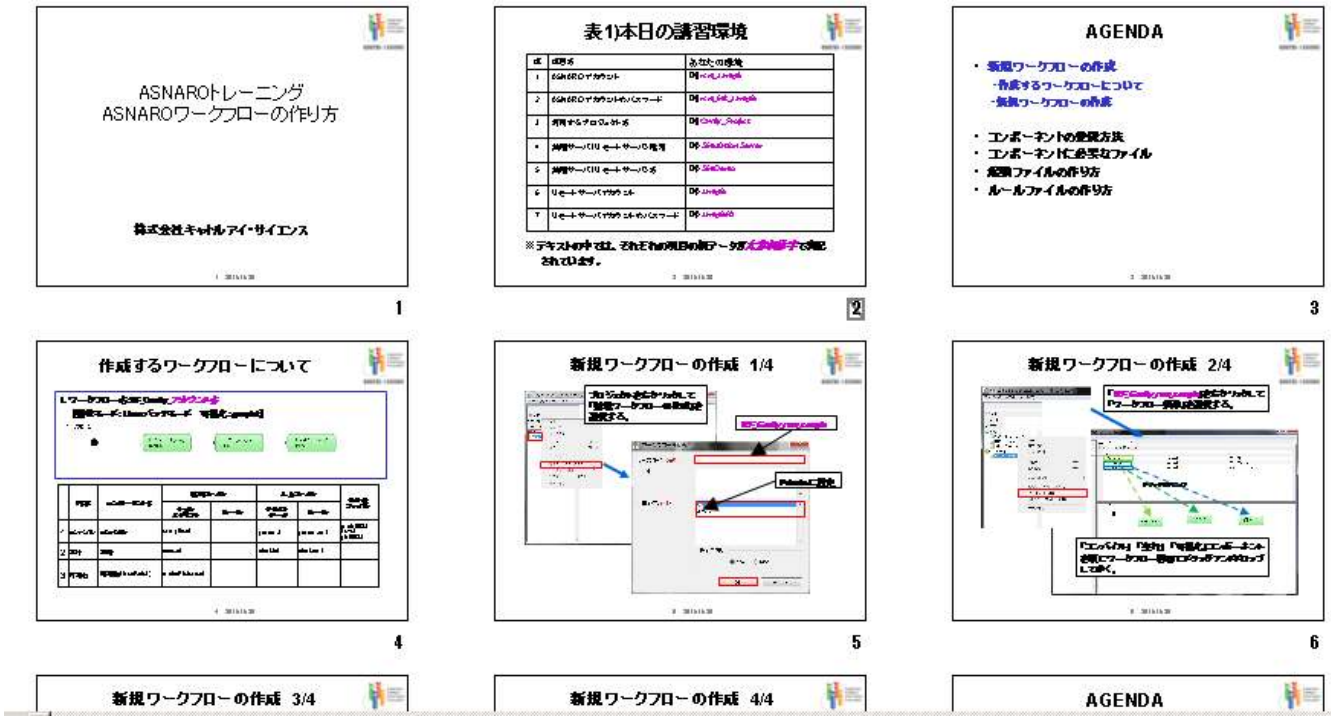


図 31 講習会資料の例（利用者者向けトレーニング資料）

図 32 講習会の様子

４－７－３　達成状況及び今後の課題

平成 23 年度 10 月末までに、当初予定していた開発は終了した。今後の課題として、より具体的なアプリケーションの使い方に沿った講習会内容の検討や、資料などの英語化等の課題があげられる。

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 年間の成果報告である。以下、サブテーマごとに開発成果を総括する。

(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- Control 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機構を開発した。

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）を確保した。

(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機構を開発した。

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機構を開発した。

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機構を開発した。

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機構を開発した。

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機構を開発した。

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機構を開発した。

(3) データベースの高機密化（排他的記録）機構

3.1 任意のデータ及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になる機構を開発した。

(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 画面で設定できる機構を開発した。

4.2 XML-Workflow 設定 GUI 画面では、XML のタグ名入力は不要にする機構を開発した。

4.3 XML-Workflow 設定 GUI 画面では、固定的な複数選択肢は、ドロップダウンで選ぶことができる機構を開発した。

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機構を開発した。

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機構を開発した。

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機構を開発した。

(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機構を開発した。

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 機能を果たす機構を開発した。

(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）追加できる機構を開発した。

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機構を開発した。

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、制御できる機構を開発した。

(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、4 頁記載の(1)～(6)の機能が正しく動作するかを 1 ヶ月以上（実績 13 ヶ月：延べ 61 ヶ月）検証した。

7.2 4 頁記載の(1)～(6)の機能の性能評価を行い、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、改善プランを継続的に策定し、開発へフィードバックを実施した。

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（通常運用マニュアルの他に、インフラヒアリングシート、RCM 設置ヒアリングシート、RCM サーバ設定・動作確認シート、管理者用トレーニング資料、利用者トレーニング資料及びそれらに必要なサンプルデータ）にまとめた。また、そのマニュアルに従って、運用、講習会を実際に行って、問題点を改善しつつ、実用上利用可能な資料に仕上げた。

委託研究期間は、2011 年 10 月末で終了し、当初目標は達成した。今後は、当初計画通り、商品パッケージとして、販売、サービスを行える状態にするために、引き続き、各サブテーマに上げた今後の課題を中心に研究開発を進めていくとともに、営業、関連企業との業務提携などのビジネス環境を整えていく。

５　参考資料

５－１　研究発表・講演等一覧

ア　収録論文

・発表方法　：講演会
・発表雑誌名　：計算工学講演会論文集 Vol.16（2011 年 5 月）
・講演会名　：第 16 回計算工学講演会
・学会名　：日本計算工学会
・発表者　：上島豊、前田茂樹、犬飼貴史、江口和宏、石原典雄
（株式会社キャトルアイ・サイエンス所属）
・発表タイトル　：PaaS-CAE 基盤技術に関する研究開発
・発表日　：平成 23 年 5 月 25 日

イ　一般口頭発表

・発表方法　：その他
・発表会名　：NICT 情報通信ベンチャービジネスプラン発表会
・発表者　：上島豊（株式会社キャトルアイ・サイエンス所属）
・発表タイトル　：NextGeneration クラウド基盤ミドルウェア
・発表日　：平成 22 年 1 月 22 日

ウ　その他資料

・発表方法　：成果展示
・発表会名　：NICT 民間基盤技術研究促進制度/ベンチャー支援制度
成果発表会
・発表者　：上島豊（株式会社キャトルアイ・サイエンス所属）
・発表タイトル　：CAE クラウド基盤技術
・発表日　：平成 22 年 12 月 9 日

・発表方法　：成果展示
・発表会名　：CEATEC JAPAN 2011 ICT Suite　情報通信研究機構ブース内
・発表者　：上島豊（株式会社キャトルアイ・サイエンス所属）
・発表タイトル　：PaaS-CAE 基盤技術に関する研究開発
CAE・シミュレーションクラウド基盤
・発表日　：平成 23 年 10 月 4-7 日