Harmonious Cloud
Hitachi vRAMcloud® Series

高速バッチ処理実現で、“攻め”の事業オペレーションを実現

M/F COBOL オンラインバッチ

ホワイトペーパー

2012 年 4 月 発行
Ver. 1.0
株式会社 日立製作所

＜本書で使用する用語＞

APL ： Application Program
ATM： Automatic Teller Machine
BCP ： Business Continuity Plan
CEP ： Complex Event Processing
CMS ： Cash Management System
CRM： Customer Relationship Management
DDL ： Data Definition Language
DR ： Disaster Recovery
EDI ： Electronic Data Interchange
KPI ： Key Performance Indicators
M/F ： MainFrame
MRP ： Material Requirement Planning
POC ： Proof Of Concept
RDB ： Relational DataBase
RFID： Radio Frequency Identification
SAM ： Sequential Access Method
SCM ： Supply Chain Management
SNS ： Social Networking Service
SQL ： Structured Query Language

＜留意事項＞

本ホワイトペーパーで記載する数値は、日立環境での計測データであり、諸条件により結果は異なります。

＜他社商標注記＞

・GemFire は、VMware,Inc.の米国およびその他の国における商標または登録商標です。日立製作所がソフトウェアを提供し、サポートを実施している製品です。
・その他記載されている会社名、製品名は、それぞれの会社の商標または登録商標です。

本ホワイトペーパーの想定読者

本資料は、(株)日立製作所 情報・通信システム社 金融システム事業部が提供する vRAMcloud®(※)を適用したソリューション「M/F COBOL オンラインバッチ」についてまとめたものです。さまざまな企業、組織において情報システムの企画・構築に携わる方々を読者として想定しています。

(※)クラウド環境でビッグデータの利活用に最適な先端テクノロジーを提供するソリューション群。

概要

企業が扱う情報は、年々増加の一途をたどり、ビッグデータ時代が到来しています。

本ホワイトペーパーは、増加し続けるビックデータを扱うために最適な先端テクノロジーを集結させた vRAMcloud®により、

- 高速バッチ機能による処理時間の短縮化
- キャンペーンや新商品／サービス投入による急激な処理量増減対応
- シミュレーション実行による先手での事業オペレーション実施

が可能なソリューション「M/F COBOL オンラインバッチ」について、その特徴を記述します。

製造業における所要量計算(MRP)、製造・流通業における販売予定・実績を反映させるリアルタイム SCM、金融業における約定データ及びネッティング(相殺処理)に代表されるような、大量データ処理が夜間処理として実施されています。

夜間バッチ処理は、処理時間が長く、結果が出たときにはすでに事業環境の変化が発生しているなど、事業環境の変化をダイナミックに把握し、企業活動に直結させるのが難しいのが現状です。

今後ますますバッチ処理を短時間で実施し、生産ラインへの適切な計画反映や経理情報の最新化で企業活動を円滑に行うことが求められます。

さらに、リアルタイム情報とのバッチ処理結果の比較検討による“次の一手”をうつ事業オペレーションが必要となってきています。

ビッグデータを高速に処理でき、且つ小規模から始め、グローバル・グループ経営までスケーラブルに適用可能となる M/F COBOL オンラインバッチをご紹介します。

目次

はじめに

近年、企業活動を支える情報システムでは、リアルタイム情報を効率的に扱い、ビックデータに対処できることが競争優位に欠かせない要件となりつつあります。たとえば、バリューチェーンの川下にある消費者動向を、川上にある購買物流、製造工程へタイムリーに反映した事業オペレーションがますます必要となります。

BtoBあるいはBtoCの取引において、取引履歴、販売履歴などの履歴情報は蓄積される一方です。さらに、属性情報（性別、年齢、気温、天気など）による細分化や、RFID（Radio Frequency Identification）による個体管理など、データの"色分け"を行うことによる付加価値が発生し、販売ターゲットの選定など事業オペレーションへ直結するようになってきています。

また、一般消費者やマーケッターからのつぶやき、ソーシャルネットワークによる口コミ情報など、商品・サービスの売上を左右する情報が巷にあふれています。

これら大量データ（ビッグデータ）を分析し、商品開発、販売計画、製造計画、購買計画などの企業活動へフィードバックすることが不可欠となっています。

経理・会計分野でもグループ経営の観点から、CMS(Cash Management System)を用いたネッティングによる経営効率化がいわれ、ますます大量バッチ処理が必要となります。

これら日々実施される大量バッチ処理を高速化し、本来実施すべき事業オペレーションの検討に、システム資源を有効に活用する必要が増してきています。さらに、処理された情報を蓄積情報とし、リアルタイム情報と比較検討することで、動向変化を敏感にとらえることが求められています。

本ホワイトペーパーは、ビッグデータ特に、バッチ処理として実施していた大量データに対してバッチ超並列処理技術を用いて短時間化する事で、処理結果を待って対応する“受身の”事業オペレーションではなく、シミュレーションなどにより、選択オプションを評価した上での、“攻めの”事業オペレーションができる高速バッチシステム化への変革をご紹介します。

また、変革に際して必要となるアプリケーションの改修コストを最小限にして移行した結果をまとめます。

本ホワイトペーパーでは、M/F　COBOLオンラインバッチを適用した実機検証の結果をとおして、バッチ超並列処理の実現やスケーラビリティの確保、移行コストの最小限化といった、本ソリューションの優位性をご紹介していきます。

M/F　COBOL オンラインバッチの狙い

vRAMcloud®の狙い

分散処理技術、仮想化技術は、黎明期から成熟期へと移ってきています。

vRAMcloud®は、企業に増加し続けるビックデータを扱うために最適な先端テクノロジーを集結させたソリューション群です。ビッグデータを“個単位”にとらえ、リアルタイム分析から将来の予想や、KPI(Key Performance Indicator)評価までをソリューション対象としています。

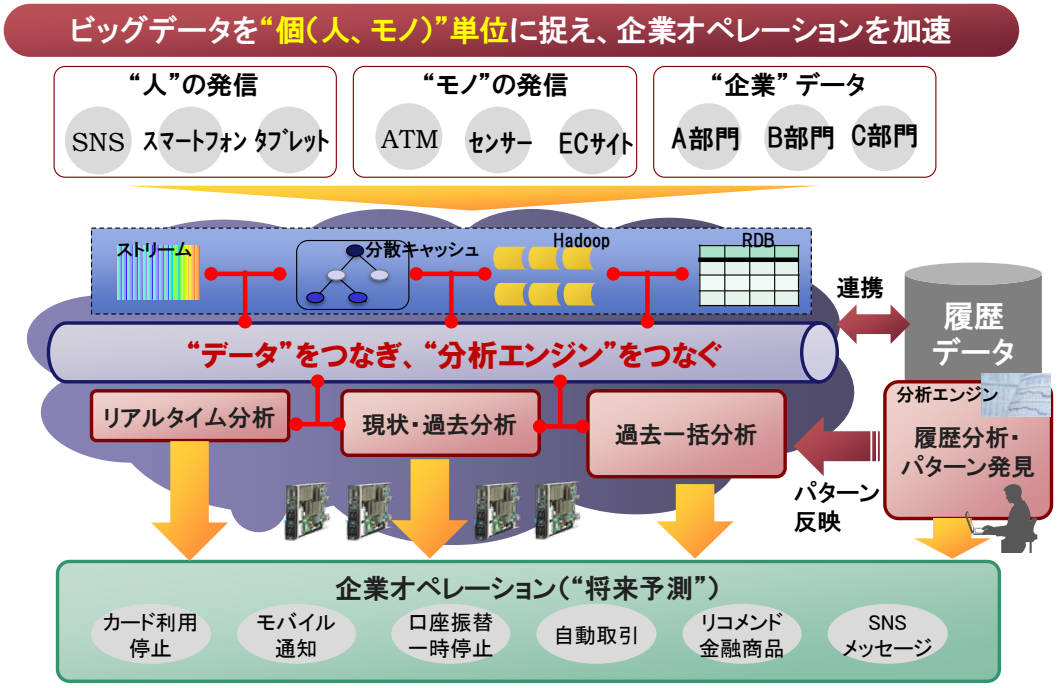

図 1　vRAMcloud コンセプト図

M/F　COBOL オンラインバッチの狙いと実現の考え方

M/F　COBOL オンラインバッチは、小規模なデータ量からビッグデータの利活用までスケーラブルに対応できるソリューションです。事業規模拡大や取扱商品の増加に起因するバッチ処理件数増加に柔軟に対応できます。これは、RDB を用いた従来の処理であれば、キーレンジ分割や、バッチ多重度を上げることになりますが、データの分散に偏りがある場合、一番データ量が多いキー値の処理に完了時間が影響されます。本ソリューションは、データ分散の偏りに影響がないアーキテクチャを採用しています。単なるデータグリッドではなく、仮想化技術、多重並列実行、ジョブスケジューリングが組み合わされることにより、発揮されるソリューションとなります。

M/F　COBOL オンラインバッチの実現には、現行システムの JOB ネットを活用します。JOB ネット構造は、そのままで、バッチ処理のボトルネックとなる JOB をグリッド JOB として登録する事により多重化が可能です。

JOB ネット上クリティカルパスとなっている処理に対して、多重並列実行を行うことによりバッチ処理高速化が可能となります。さらに本ソリューションは、バッチ JOB にはありがちな中間ファイル(SAM ファイル)を処理ステップで保持する必要はなく、インメモリで処理するため中間ファイル作成・読み込み処理時間、ソート・マージ時間も短縮できます。

目指すべき方向性

M/F COBOL オンラインバッチは、24 時間 365 日が当たり前と考えられる今後のビジネス環境に対して、有効に機能します。さらに、蓄積された情報を統計的に判断し、“今”の情報との比較による事業オペレーションの効率化が可能となります。

現行の JOB ネット構造は、システム処理順序、他システムとの送受信連携時間などにより定義されたものです。つまり、業務上リラン単位を定義した JOB ネットは、運用ノウハウの固まりです。本ソリューションは、JOB ネットをそのままに、最小移行コストでバッチ超並列処理を実現できます。

さらに、バッチ時間が短縮できることから、シミュレーションのように Try & Error を前提とした業務オプション評価により、先の見えにくい時代を、“攻め”の事業オペレーションを実行することが可能となります。

スケーラビリティ

M/F COBOL オンラインバッチでは、特定のサーバに負荷が集中、あるいは、障害となった場合でも動的にサーバを追加・削除できる仮想化技術アーキテクチャを採用することにより、スケーラビリティの確保をしています。動的にスケールアウトが可能で高負荷となっている処理を中断することはありません。スケールアウトに伴い、データの再配置が行われ負荷の平準化が可能です。

バッチ JOB の並列度を動的に変更可能であり、変更後のデータリバランスについても自動的に制御します。特異日やキャンペーンのような急に処理対象データが増加した場合でも、動的にマシンの増加が可能です。これはデータ流量を監視しており、一定数を超えた場合は、あらかじめ設定した並列度に仮想マシン構成を動的に変更します。

vRAMcloud®で適用している分散キャッシュ技術は、同一拠点にとどまらず他拠点との分散化が可能となる DR 対応機能が組み込まれており、バックアップサイトへのデータ転送もパラメータを設定するだけで、差分情報のみを転送可能で、ユーザが接続先を意識することなく、BCP 対応は完了します。

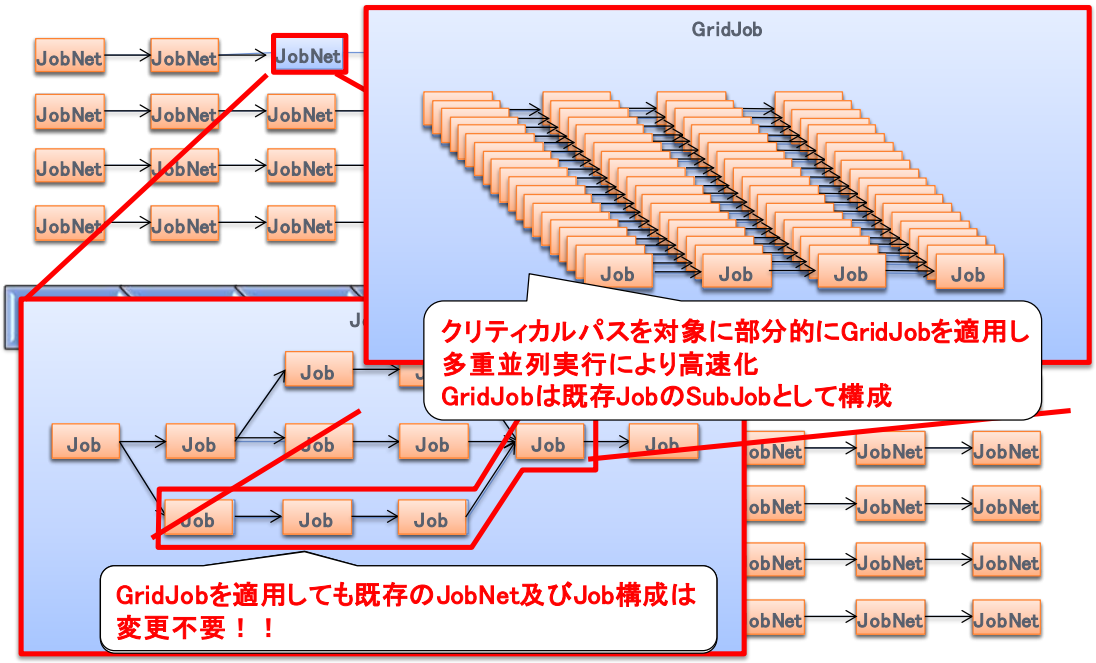

図 2 ジョブネットの並列処理化

現行システムからの移行について

M/F COBOL オンラインバッチの適用により、現行システムの品質を保ちながら、移行することができます。具体的には、JOB ネットやアプリケーションロジックはそのままで、DB I/O、FILE I/O のみを変更します。

バッチ処理システムは、JOB ネットと JOB を構成するプログラムから構成されています。

JOB ネットは、JOB 管理ツールなどにより定義され、時間起動あるいは先行 JOB 終了、待ち合わせなどの起動条件があります。これら、JOB ネットは、他システムとの連携もあるため変更すると運用が変わってしまいます。可能な限り、現状の運用をそのままにして移行することが最善です。

M/F COBOL オンラインバッチは、JOB ネットの変更を最小限として移行することが可能です。

移行の概要

M/F COBOL オンラインバッチを適用してバッチ処理環境を移行するには、以下の 2 点の特性をみて、移行を検討する必要があります。

1) JOB ネット(JOB ステップ)の移行
2) アプリケーションの移行

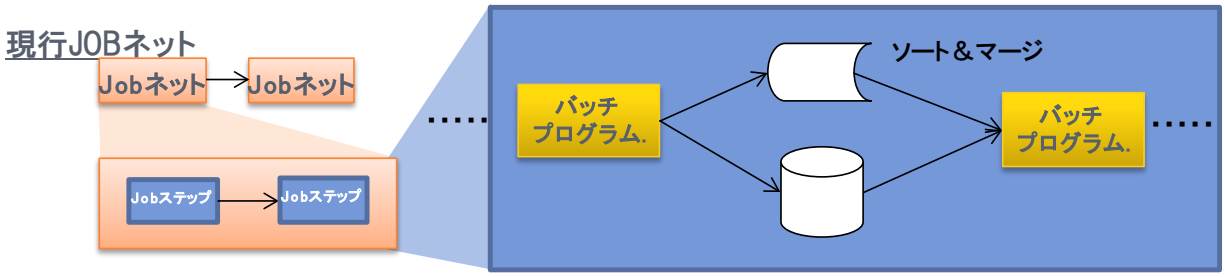

図 3 ジョブネットの構造

JOB ネット移行

JOB ネットと JOB ステップの移行は、OS やミドルウエアの起動・停止などを定義したシステム基盤系 JOB 以外、変更しません。ただし、時間起動されている JOB については、M/F COBOL オンラインバッチにより処理時間が短くなるため調整が必要となることがあります。

JOB の多重化は、処理データ量を監視し動的に変更することが可能です。これは、M/F COBOL オンラインバッチのフレームワークで実現しており、既存 JOB ネットを変更することなく適用できます。運用実績のある JOB ネットを活用しながら、高速バッチ処理化できるメリットがあります。

アプリケーション移行

アプリケーションの移行は、バッチ処理プログラムの DB I/O、FILE I/O を変更するだけで、業務ロジックは変更しません。

また、変更対象箇所についても、現状の SQL 等を解析し、M/F COBOL オンラインバッチソリューションで提供するインタフェースに変更します。変更には、ソース解析ツールを活用することにより、極力人手を介さないことで、現行品質を保ちながら移行することが可能となります。

下図は、バッチ処理が COBOL で実装された場合の事例です。DB I/O（SQL 変数定義部、SQL 実行部）、FILE I/O 変数定義部、FILE アクセス部を抽出し、C/C++モジュール、Java インタフェースと変換することにより、現行アプリケーションの業務ロジックを変更せず移行が可能です。

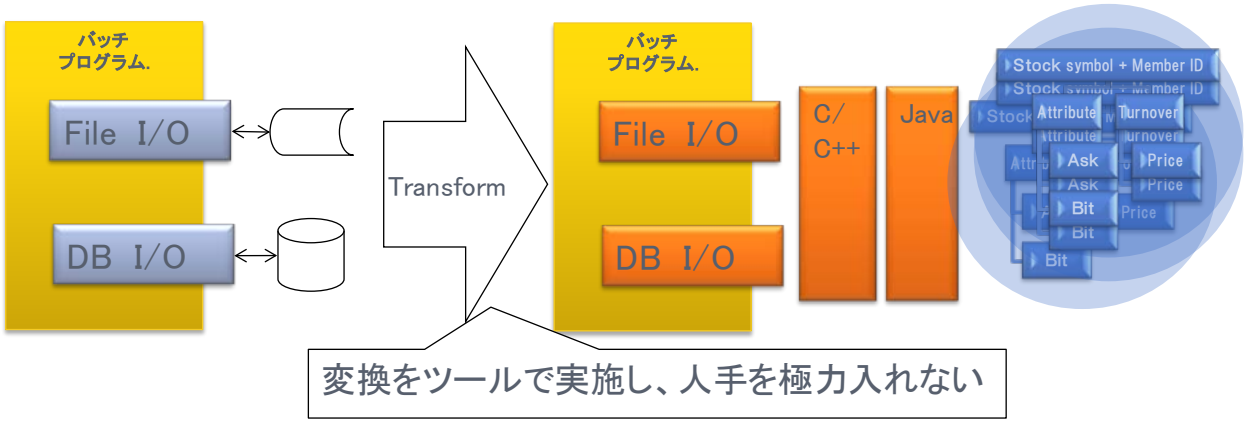

図 4 現行プログラムをツールにより vRAMcloud での実行環境へ変換

関連システムとの連携

M/F COBOL オンラインバッチソリューションの適用により、当該システムのバッチ処理時間が短くなっても、バッチ処理全体に必要な時間は他システムとの連携があるため、短くなるとは限りません。

バッチ JOB では、システム間連携を実施するために、ファイルのやり取りを行っています。これら関連システムからのファイル授受は、連携時間や連携手順について取り決めがあります。M/F COBOL オンラインバッチソリューションによりバッチ処理時間が短縮されたことによる他システムへの影響を検討する必要があります。

実機検証結果

検証概要

某企業のネッティング処理を例に、M/F　COBOL オンラインバッチの実機検証を実施しました。実機検証用に用いたプログラムは、実システムをモデルに作成し、COBOL プログラムからツールによるDB I/O 移行、FILE I/O 移行を実施し移行性の評価も実施しました。

検証環境

本検証における各マシンの役割及びサーバ環境を以下に示します。

分散処理を行うサーバを 10 台利用し、処理のスケールアップ、スケールダウンを動的に検証しました。具体的には、グリッドバッチを制御する運用サーバを 1 台設定し、残り 8 台で並列処理を行いました。処理データ量を監視し、処理件数が多い場合に動的スケールアップを行い、並列度を上げて処理しました。

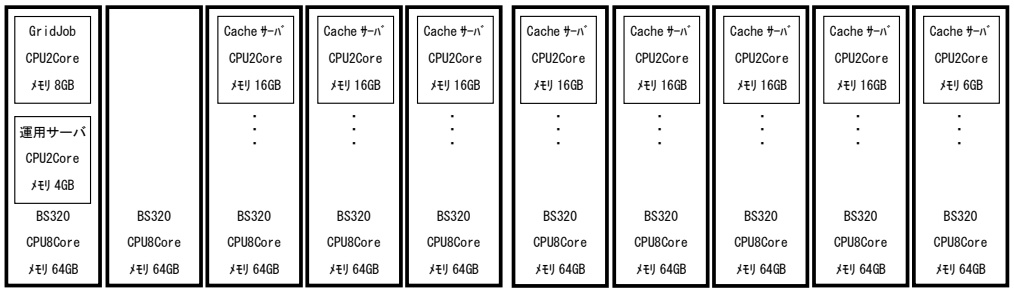

図 5　ハードウエア構成図

ハードウエア構成、ソフトウェア構成は以下の通りです。

表 1 ハードウエア一覧

#	項目	内容	員数
1	サーバ	BladeSymphony320	10
2	CPU	Xeon L5630(2.13GHz,4Core,12MB)×2	
3	メモリ	DDR3 16GB×4	
4	ディスク	147GB(SAS10,000rpm)×2	
5	ネットワーク	1GbpsLANSW	1
6	共有ディスク	BR20 146GB(SAS 10,000rpm)×12 (=1TB)	1

ソフトウェア構成

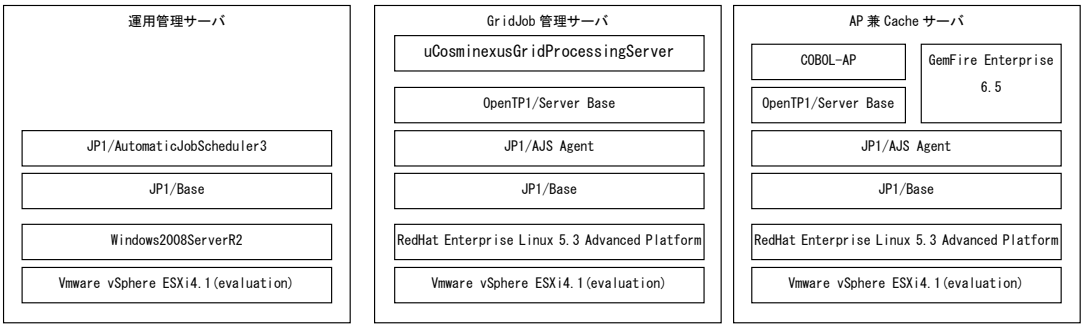

図 6　ソフトウエア構成図

検証方法

実証実験では、実システムをモデルとしたバッチ処理を対象とし、超並列バッチ処理化した際の処理速度の向上効果を測定しました。また、COBOL プログラムの DB I/O 移行、FILE I/O 移行を行った際の改修ステップ数を測定し、移行性を評価しました。

検証に際しては、バッチ JOB の並列度を上げてスケールアップを行い、スケールアップ時の性能向上効果を評価しました。

検証結果

本検証では、モデルとしたシステムとして某企業のネッティング処理（データ格納→集計→帳票印刷）を対象としました。複数の業務処理を対象に検証を実施しており、以下に各業務処理の結果を示します。

データチェック・格納処理

入力となるファイルをデータベースへ格納する前にデータチェックする処理と、ファイルをデータベースへ格納する処理を行なうJOBを実行しました。データチェック処理では、データの整合性や英数字チェックを行ないました。

検証の際は、処理対象データを800万件とし、現行多重度1で実行されている処理を、多重度12、16、32多重と変化させて実行しました。図 7 に実行時の処理時間とスループットを示します。

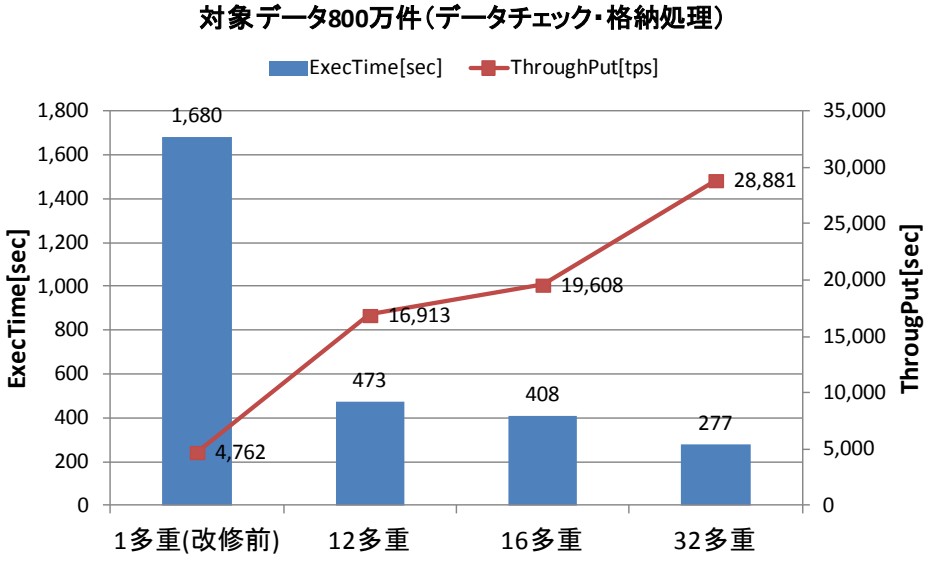

図 7　データチェック・格納処理の並列バッチ処理

1 多重（改修前）と比較すると、32 多重で並列バッチ処理を実施した場合、約 6 倍の性能が出る事が確認できました。また、16 多重から 32 多重へスケールアウトさせた場合、約 1.5 倍の性能向上が確認できました。

データ集計処理

入力ファイルに複数存在する会社コード別にデータの集計を行い、合算値を計算する処理を実行しました。

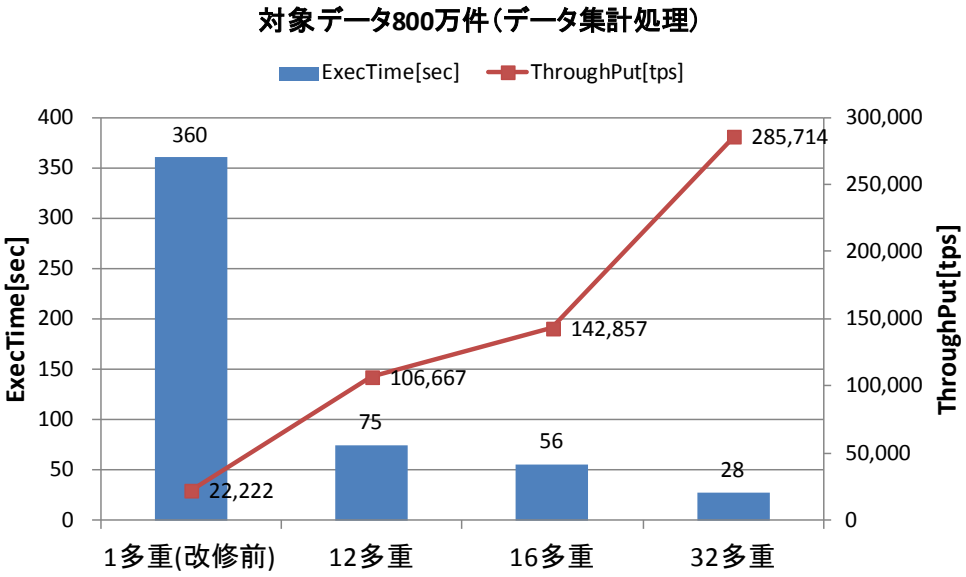

図 8　データ集計処理の並列バッチ処理

1多重（改修前）と比較すると、32多重で並列バッチ処理を実施した場合、約13倍の性能が出る事が確認できました。また、16多重から32多重へスケールアウトさせた場合、2倍の性能向上が確認できました。

帳票データ作成処理

処理対象データから帳票作成のために明細を作成する処理を実行しました。

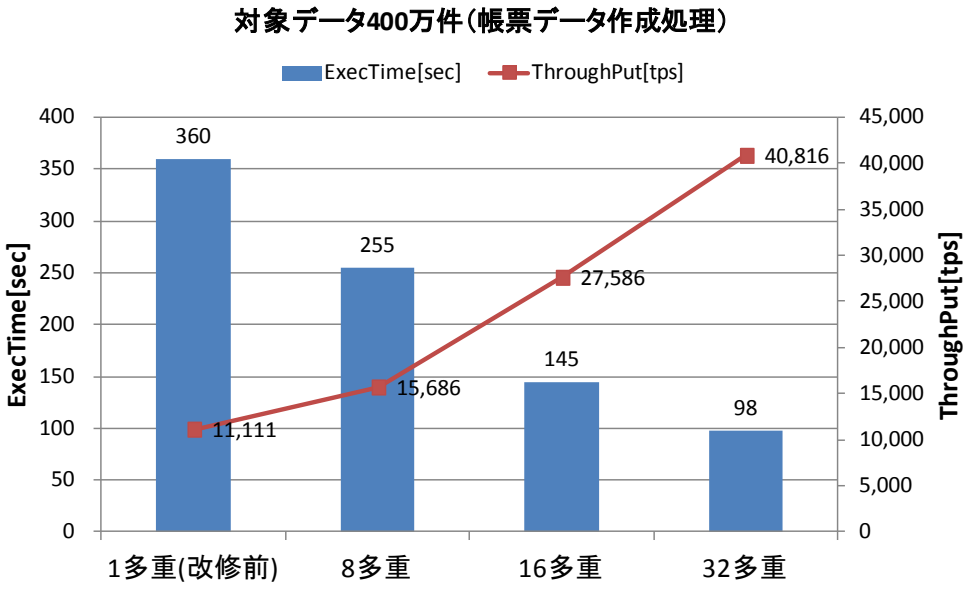

図 9　帳票データ作成処理の並列バッチ処理

1 多重（改修前）と比較すると、32 多重で並列バッチ処理を実施した場合、約 4 倍の性能が出る事が確認できました。また、16 多重から 32 多重へスケールアウトさせた場合、約 1.5 倍の性能向上が確認できました。

運用設計について

本検証では、モデルとした実システムの JOB ネットにおいて、グリッドバッチ化による性能向上効果が見込める JOB を対象にグリッドバッチ化しました。そのため既存の JOB ネットには、ほとんど変更を加えずにグリッドバッチ化する事ができました。これにより、既存の JOB ネットの設計を保持しつつ、グリッドバッチによる高速化が行なえました。なお、既存の JOB ネットは 7 つの JOB から構成されていましたが、このうち並列実行可能な 5 つの JOB をグリッドバッチ化しました（※）。

※並列実行可能な JOB とは、キー分割により業務処理が平行して実施できるもの

大量パラレル処理による対策

大量パラレル処理（グリッドバッチ処理）を実施するにあたり、実行ノードへデータを分散配置させる方法を検討する必要があります。本ソリューションでは、データを分散配置させる際に処理対象データのキー情報に従って分散配置させています。分散配置の例を図 10 に示します。

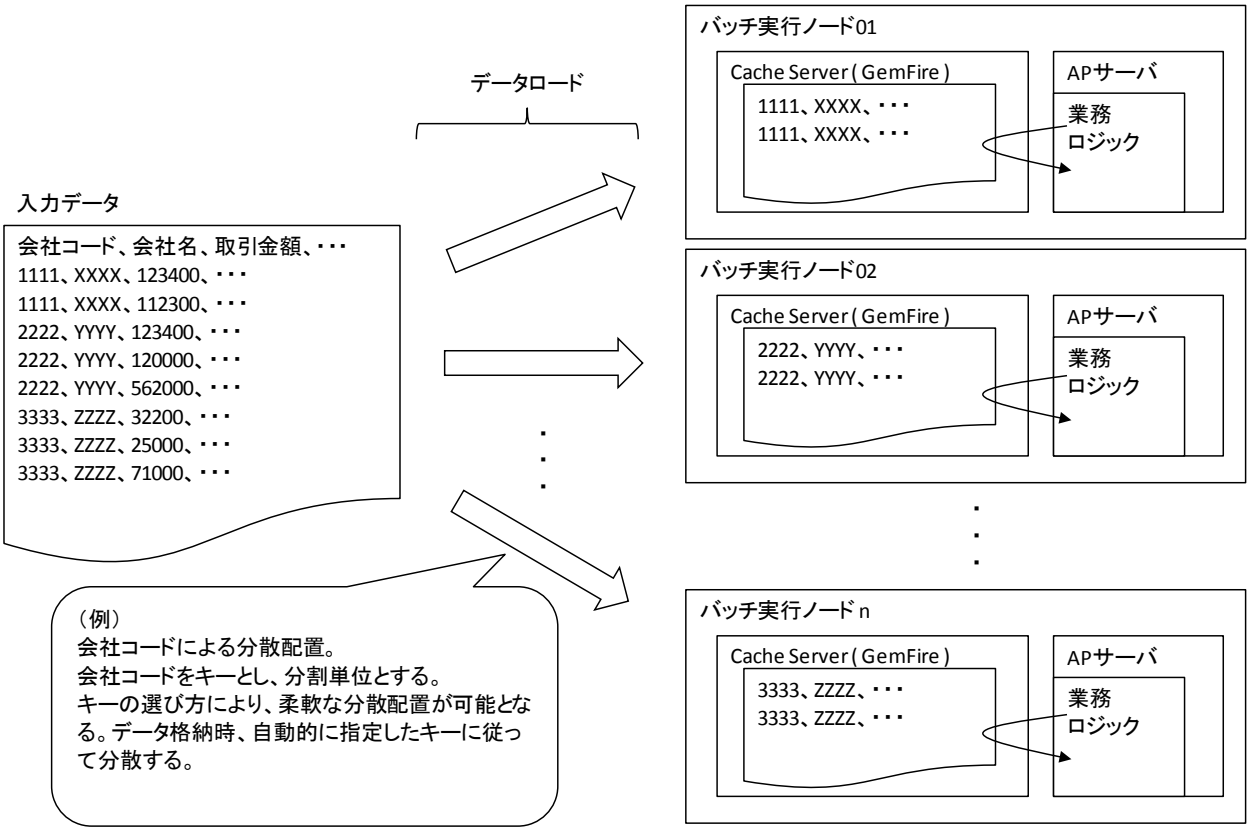

図 10　分散配置の例

上記の例では、会社コードを分散配置のためのキー情報としています。データは、会社コードごとにバッチ実行ノードにデータを配置され、各実行ノードはノード上のデータのみを対象に処理を行ないます。

上記の例のほかにも、以下のようなキーを指定する事でデータの柔軟な分散配置が可能です。バッチ処理の特性に合わせて分散配置のためのキーを指定し、グリッドバッチの効果を高められます。

- 証券業務　…　株取引で「銘柄コード」をキーとし、株ごとのバッチ処理を実行する、など
- 銀行業務　…　「支店コード」をキーとし、営業店単位でのバッチ処理を実行する、など
- 流通業務　…　「商品コード」をキーとし、商品別に効率的なバッチ処理を実行する、など

分散配置データの細分化の検討

各サーバに分散配置されるデータは、分散キャッシュ（GemFire）によって自動的に配置先のサーバが決定されます。そのため、場合により特定のサーバにデータが偏って配置される場合があります。特定の分割単位にのみデータが集中すると、そのサーバの処理時間は長くなり、グリッドバッチ全体の処理時間も長くなってしまいます。

今回の検証では、データの分割を会社コード単位で行ないました。会社コード単位の分割では、実際の取引において、ある特定の会社コードに取引が集中すれば、その会社コードのバッチ処理に長時間掛かってしまいます。そこで会社コード単位の分割だけでなく、「会社コード＋取引者コード」によるデータの分割を検討しました。

しかしながら、今回の検証では「会社コード＋取引者コード」による分割を行なうと業務アプリケーションでエラーが発生してしまい、データ分割を行う事ができませんでした。データ分割できないケースとして、表 2 に示します。

表 2　データ分割を行なえないケース

#	観点	説明
1	レコードの整合性チェックを実施しているケース	通番が振られている項目において、「1から連続した通番であること」といったデータチェック処理がある場合、分割すると業務アプリケーションのチェック処理でエラーになる可能性がある。
2	シーケンシャルな処理を実施しているケース	連続する複数のレコードが必要となる処理がある場合、分割すると業務アプリケーションでエラーまたは結果不正となる可能性がある。

今回の検証でモデルとした業務アプリケーションでは、表 2の項番1に該当してしまったため、会社コード単位で分割したデータをさらに分割する事ができませんでした。

さらなる高速化を検討する場合、アプリケーションの改修が必要になる場合がありますが、上記方針で実施できれば並列実行が可能な処理とできます。

改修ステップ数による移行性

移行性を評価するに当たり、本検証における移行方式の概要を図 11 に示します。

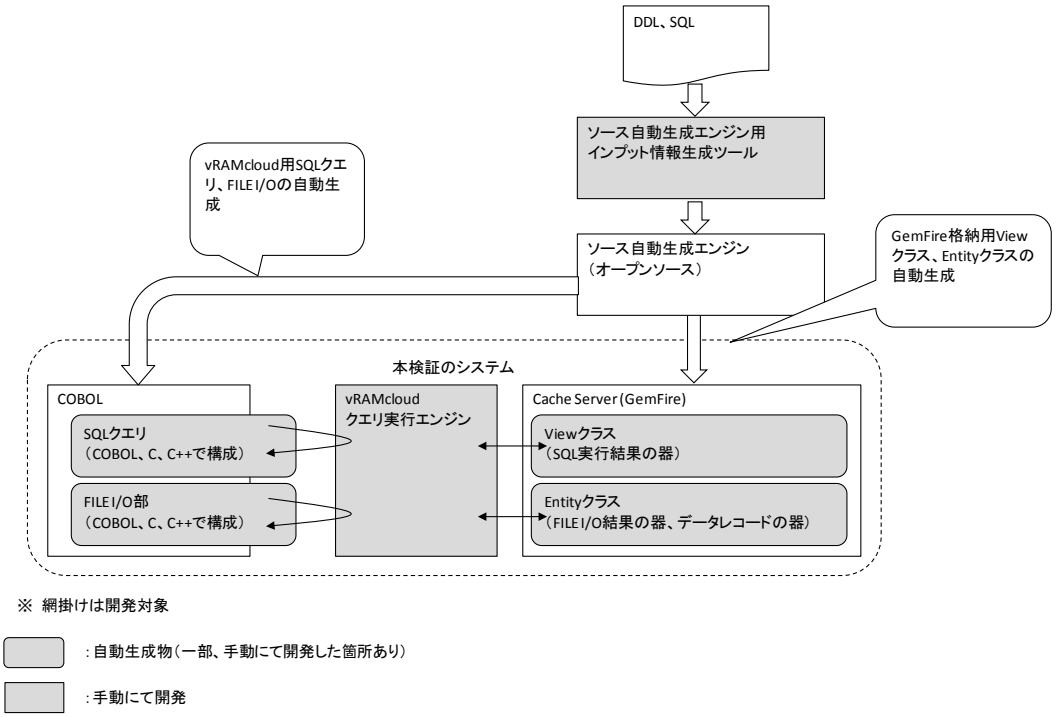

図 11　本検証の移行方式

本検証では開発生産性を向上させるため、ソースコード自動生成ツールを使用しました。一部、複雑なクエリは人手による実装を行ないました。

今回の検証で開発したプログラムと、そのステップ数を表 3、表 4 に示します。

表 3　COBOL アプリケーションの修正ステップ数

#	種別	ステップ数[steps]	備考
1	既存のアプリケーション（POC対象のみ）	14,034	ー
2	修正ステップ数	1,913	既存APLの13.6%

表 3 COBOL アプリケーションの修正ステップ数が示す修正ステップ数の修正内容は、全て SQL クエリ部分のみの修正です。業務ロジックは既存のアプリケーションをそのまま保持し、移行を実現しております。

表 4　フレームワーク上のアプリケーション開発ステップ数

#	開発対象	開発方法	ステップ数[steps]	備考
1	クエリ実行エンジン改修 （既存フレームワークのエンジンのエンハンス）	コーディング	3,089	
2	インプット情報生成ツール (DDLとSQLから自動生成に必要な情報を抽出するツールの開発)	コーディング	1,086	ー
3	vRAMcloud用クエリ、FILE I/O部、 Viewクラス、Entityクラス	コーディング	2,957	自動生成率　84.7%
		自動生成	16,316	
4	合計	コーディング	7,132	ー
		自動生成	16,316	

表 4 の項番1、2については、実システムを開発する際に再利用可能なため、実システム開発時のステップ数には含まれません。クエリやデータ格納用クラス(View クラス、Entity クラス)などの開発は、自動生成率 80%以上で開発が行なえました。

表 5 に本検証での工数を示します。

表 5　本検証での開発工数（設計～単体テスト）

#	開発対象	工数[人月]	備考
1	クエリ実行エンジン改修 （既存フレームワークのエンジンのエンハンス）	1.6	
2	インプット情報生成ツール （DDLとSQLから自動生成に必要な情報を抽出するツールの開発）	0.5	―
3	COBOL修正、vRAMcloud用クエリ、FILE I/O部、 Viewクラス、Entityクラス	1.5	自動生成率　84.7%（COBOL修正は含まず）
4	合計	3.6	―

項番 3 の工数については、自動生成部分は含みません。自動生成を効果的に利用することで、高い生産性で開発する事ができました。

また項番 1 の工数については、本検証での成果を活用するため、実システムの開発では盛り込む必要がありません。

以上により、COBOL プログラムの修正は最小限に抑え、COBOL プログラム以外のアプリケーションについても自動生成により非常に高い生産性によって開発・移行を行う事ができました。

想定サービス提供形態

M/F　COBOL オンラインバッチは、プライベートクラウドとしてご提供するソリューションです。お客様の既存 JOB ネットの変更を最小限としながら、バッチ処理自体を高速化します。構築・導入期間を短縮できるフレームワーク、及び移行をDB　I/O、File　I/O 移行支援するツールを用意しております。また、安価なPC サーバを使ったスケールアウト構成が可能なため、ハードウエアの投資コストを最小限に抑えられます。

グリッドバッチ提供モデル

M/F　COBOL オンラインバッチソリューションは、お客様のバッチ処理高速化エンジンとしてご提供いたします。さらに、既存アプリケーションをM/F　COBOL オンラインバッチを適用した環境上で実行可能とするための移行ソリューションを提供します。

ソリューションメニュー

日立から御提供させていただくソリューションメニューは以下のようになります。

表 6　ソリューションメニュー一覧

#	メニュー	概要	費用	備考
1	導入診断 コンサルテーション	ソリューション導入可否、及び費用対効果の算定	個別見積	
2	導入コンサルテーション	M/F　COBOL オンラインバッチ導入支援（要件定義、カスタマイズ実施）	個別見積	
3	システムインテグレーション	M/F　COBOL オンラインバッチと連携が必要な既存業務システム連携を含め、高品質システム開発を支援	個別見積	
4	既存システム移行マイグレーション	既存システムを M/F　COBOL オンラインバッチ化するマイグレーション支援	個別見積	

まとめ

M/F　COBOL オンラインバッチは、バッチ処理を超並列処理することにより、高速化が可能です。さらに、処理件数が一定の閾値を越えると、動的に仮想マシンを追加し並列度を増して処理できます。

既存の JOB ネットはそのまま生かし、アプリケーションの DB　I/O、FILE I/O を変更することにより移行可能です。バッチ処理のリランについても、処理時間が従来と比較して短くなることから、運用が容易になります。

高速にバッチ処理が可能となることで、通常業務で実施していた業務処理のみではなく、シミュレーションなどによる販売予測・目標設定や製造計画見直しなど高速に実行できることで、さらなる付加価値が実現できます。今までできなかった最新状況を加味しながら、高速バッチ処理により過去の傾向分析、シミュレーションを実現できることから、次の手を“攻め”の事業オペレーションを実現します。

以上