証券オンライントレードシステムの高性能化への取り組み

野村総合研究所
インターネットトレードシステム事業部　上級システムエンジニア

城谷 勝（しろたに まさる）

専門は、証券会社向けオンライントレードシステム開発。入社後、証券会社向けの注文システムの開発部門に配属になり、一貫してミッションクリティカルなシステムの開発に従事。証券注文オンラインシステムのダウンサイジングや､投信口座管理システムの開発に従事し、98年からネット証券向けシステムの企画、開発プロジェクトに参画。現在は、新技術を使ったネット証券向けサービスの企画、開発を行っている。

要旨

手数料自由化以降、日本でも証券オンライントレードが定着し、いまや個人投資家の取引のうちインターネット経由の注文が9割（売買代金ベース）に達しているという。こういった利用者の増加に加えて、アルゴリズミックトレーディングやPTS（私設取引システム）といった高度なサービスへのニーズも高まっており、より高性能な証券オンライントレードシステムが求められている。米国では、証券会社や取引所がキャッシュ技術を活用した高性能システムを実現しており、参考にすべき点は多い。また、NRI（野村総合研究所）でも同様の設計思想にもとづいたリッチクライアントシステムを構築し、PCサーバ上での高性能処理を実現している。

本稿では、米国と当社プロジェクトの事例解説を中心に、最後に、今後実現すべき高性能証券オンライントレードシステムの要素技術についてポイントを述べる。

キーワード：証券オンライントレード、リッチクライアント、Javaフレームワーク、キャッシュデリバリ技術、PTS、ECN、アルゴリズミックトレーディング、EDF、J2EE/JMS/MDB、マルチキャスト

After deregulation of brokerage commissions, stock market online trading has become established in Japan. Now it is said that private investors' order via internet has reached 90 percent, on a trading value basis. In addition to user increase, higher performance stock market online trading system to meet the needs for advanced services such as algorithmic trading or PTS (Proprietary Trading System) is required. In the U.S., Stock Exchange and Securities companies have actualized higher performance system by utilizing cache technologies, where there are a lot of points to refer. NRI, Nomura Research Institute, also has built up rich client system based on a similar design concept, and actualized high performance processing on PC server.
This article focuses on the case description about the projects in our company and in U.S. And in the end, it will refer to the essential points regarding elemental technology on high performance stock market online trading system, which should be actualized in future.

Keywords : Stock market online trading, Rich client, Java framework, Cache Delivery technology, PTS, ECN, Algorithmic trading, EDF, J2EE/JMS/MDB, Multicast

1．はじめに

1998年に本格的なインターネットを使った証券オンライントレードが始まって以来、昨今では、株式委託売買額全体の3割は、個人投資家によるインターネット取引が占め（表1）、口座数が100万を超えるオンライントレード証券会社も出現している。また、日本証券業協会が新聞発表した情報によれば、2006年度の上半期においては、個人投資家が取引した株式売買代金のうち9割はインターネット経由の取引が占めるという。米国においても、2006年の7月以降、個人マネーが相場を押し上げ、インターネットによる株式売買件数が最近になって再び増加傾向にある。景気動向やマーケットの状況に左右されることはあるが、中長期で見るとインターネットチャネルの重要性は今後も変わることがない。本稿では、コモディティ化されつつある証券オンライントレードシステムについての課題や求められている新サービスの動向を踏まえ、先行している米国企業の事例やNRIの取り組みを紹介し、今後実現すべき要素技術について展望する。

2．証券オンライントレードシステムの現状と課題

（1）性能面での課題

①集中化するトランザクション

オンライントレード証券の性能的な特徴としては、取引所開始時の集中的トランザクションがあげられる。寄り付きといわれる取引所での売買が開始される午前9時の前後ではトランザクション量が瞬間的に通常の100倍を超えることもある。利用者が社内に限られるような企業向けのいわゆるエンタープライズ向けシステムでは、秒間トランザクション

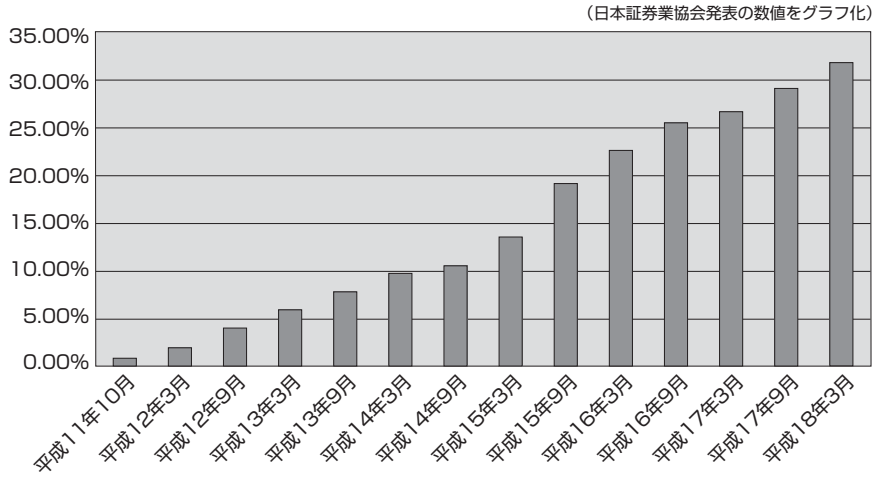

表1　インターネット取引の割合（株式委託代金ベース）

量が1,000件を超えるシステムは、そう多くは見当たらない。しかし、インターネットを使ったコンシューマ向けのシステムの場合は、最大秒間トランザクション量が1,000件～10,000件というのは現実的な性能要件となる。

一方、日本の証券オンライントレードを支えてきた技術を顧みると、本格的に始まった1998年当初は、インターネット上の技術もそれほど確立しておらず、ベースとしては、それまでダウンサイジングプロジェクト等にて構築経験のあるUNIX、OLTP、RDBMSを軸にした、いわゆる枯れた技術を採用するケースが多かった。しかし、それらはエンタープライズ向けシステムのニーズを前提としており、インターネットの本格利用が進むにつれ、性能面での課題が深刻化している。

これまでの経験をベースに、口座数別に証券オンライントレード専業の仮想証券会社を想定し、発生するトランザクション量をまとめた。

②システム構成上の課題

現在、証券オンライントレードシステムの基本的なシステム構成は、3階層からなるケースが多い。つまり投資家へのインターフェイスを担当するプレゼンテーション層（以下、PLサーバ）、業務ロジックを担当するビジネスロジックサーバ、データ管理を担当するデータベースサーバ（以下、DBサーバ）である。こういった3階層システムでの課題は、トラフィックが増大すると、サーバをスケールアウト的に並列で増設できないデータベースサーバへのアクセスが集中してしまい、DBサーバを上位機種にスケールアップさせないと、性能拡張が困難なことである。確かに、昨今のDBサーバの中には、複数台からなるDBサーバを論理的には1つのデータベースとして見せることで、サーバの増設によりリニアに性能向上が可能と謳っているプロダクトも存在している。しかし、経験上、サーバ増加に対してリニアに性能を向上させるには

	口座数	10万規模	50万規模	100万規模
(a) 1日の平均注文件数	実績値から推定	1.5万	7.5万	15万
(b) 1日の最大注文件数（相場活況期）	(a) × 300% >	4.5万	22.5万	45万
(c) 寄り付き時の注文件数（相場活況期）	(b) × 30%	1.8万	6.8万	13.5万
(d) 1日の最大約定件数（相場活況期）	(c) × 50%	0.9万	3.4万	6.8万
(e) 特殊注文件数（相場活況期）	(c) × 10%	1800	6800	13500
(f) 秒間での最大Webトランザクション件数（相場活況期）	投資家からのアクセス	500PV/S	2500PV/S	5000PV/S

（PV/S…1秒あたりのWebページ参照件数）

表2　口座規模別性能要件（仮想証券会社）

制約も多く、現場での現実解とはなっていない。その一方で、DBサーバへのアクセス集中は、基盤ソフトウエアやアプリケーションの処理方式自体にも問題がある。つまり、インターネット経由で投資家から入力されるトランザクションと証券取引所からの約定データは、ともに1つのDBサーバへアクセスするため、特に寄り付きといわれる証券取引所の売買開始時間帯では、それら2方向からのトランザクションが、同一DBサーバ上で同時に処理されることになる。さきほどの仮想証券会社の100万口座の例でもわかるとおり、相場活況期の寄り付き時に発生する注文件数が13.5万件、約定件数が6.8万で、少なくとも合わせて20.3万件のトランザクション処理がDBサーバ上で実行されることになる。2桁のCPU数を持つ大型UNIXサーバでさえ、RDBMS上での処理性能は、実際に稼動しているアプリケーションでは秒間700件強ほどしかなく、すべての処理が終わるまで最速でも5分以上かかる計算となる。秒のレベルでトレーディングを行うアクティブトレーダにとっては致命的な遅延である。

(2) 高機能性の追求

システム構成上の問題を抱えつつも、新しいサービスに対するニーズも高まってきている。それらのニーズは性能面でのインパクトが大きく、既存のシステム構成で実現するには困難なものも多い。

①トレーディングツール

オンライントレードの世界においても、ユーザエクスペリエンスの重要性が今後ますます重要になってくると思われる。たとえば、アクティブトレーダといわれる、頻繁に取引を繰り返す投資家は、変動する株価をモニタリングしながら、数秒の判断で注文を頻繁に行っていく。オンライントレードが始まった当初はWebブラウザによって行うしかなかったため、更新ボタンを押下しないとそのときのリアル時価が参照できなかった。この方法では、アクティブトレーダが求めるスピード感を実現することができない。また、個人投資家の層が厚くなるに従い、大量退職により豊富な資金がある団塊世代とアクティブトレーダのそれぞれのニーズに応じた別々のユーザインターフェイスも求められてくる。これらの要求に応えられるシステムの1つがリッチクライアント技術を活用したトレーディングツールである。トレーディングツールは、PCにアプリケーションの実行環境を搭載するもので、Webブラウザのように画面操作のたびにサーバ側で画面編集処理を行うのではなく、投資家のPC上で編集処理が行われる。この方式により画面上の数値も自動的に変更されていくため、リアル時価や刻々と変わるニュース情報を動画のごとく表示させることができる。また、注文系と情報系の画面も一体化するため、数秒の世界で売買を繰り返すアクティブトレーダには必須のものとなって

いる（図1）。証券会社によっては、2割以上の注文がトレーディングツール経由で入力されるケースもある。

今後、こういったリッチクライアントによる多様なユーザインターフェイスが増えてくると、サーバ側にも高度な性能が求められる。リッチクライアントを通して表示させるべきリアルタイムな情報としては、証券取引所からの気配値、気配数量、基準値段、約定値段といった更新データがあり、秒間1,500件以上のトラフィックで配信される。さらにこれらのデータを各投資家のPCに対して、秒間30件以上の性能で送出する必要がある。これらの処理をサーバ側で実装するには、プッシュ技術とマルチキャストを活用したデータ配信処理やキャッシュ技術を使った高速処理を組み込む必要がある（図2）。情報を顧客にリアルタイムに発信していく基盤（以下、プッシュ基盤）は今後重要となっていくが、サーバ側の基盤をいかに低コストで効率的に高速処理を実現するかが重要なポイントとなってくる。

②特殊注文

オンライントレードの機能比較で、重要な要素の1つに特殊注文がある。通常、証券の売買をする場合に、銘柄の種類、値段、数量を指定して売買する。価格の指定は、成行（なりゆき）というマーケットプライスに委ねる方法と指値（さしね）という値段を指定する方法が基本となる。昨今では、この基本系から派生して、さまざまな執行の条件を指

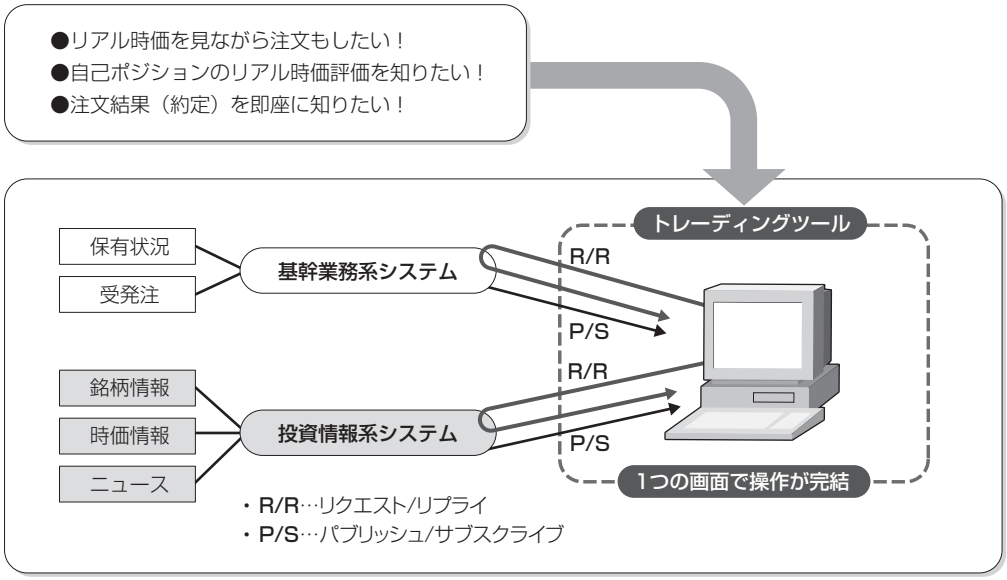

図1　トレーディングツール概念図

定できるオンライントレーディングの仕組みが数多く出現している。

このようにさまざまな執行条件を指定させる方式では、取引所に注文せずに、まずは自社のシステム内で、投資家別の執行条件を管理し、値段条件や約定条件をリアルタイムにチェックしながら、条件とマッチングした場合のみ、即座に注文電文を生成して取引所へ注文回送する必要がある。これを実現するには、サーバ側で高速なイベントドリブン型の

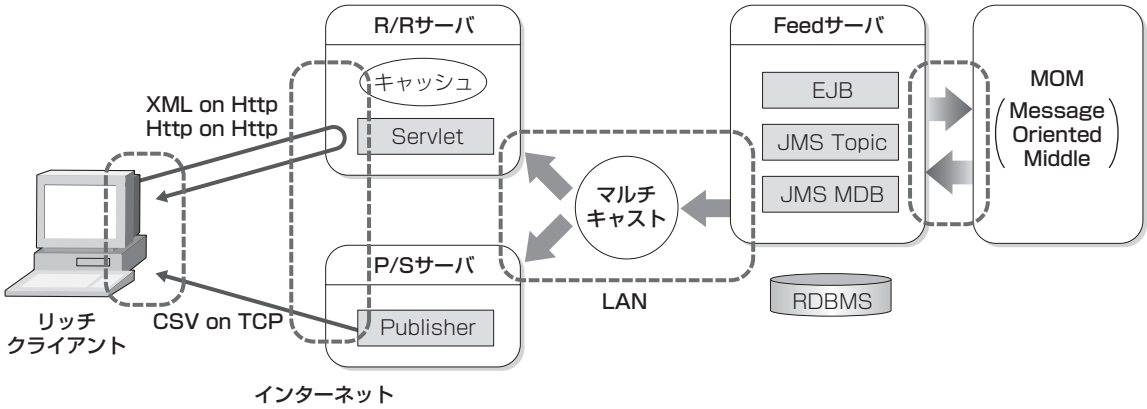

図2　リッチクライアント システム概念図

種類	内容
逆指値注文	通常の指値による売り注文では、売却によって利益が得られる株価を指定して注文を行うが、逆指値では、株価が下降しても傷口が広がらないよう、株価が指定した水準まで下がった場合に、自動的に売却注文を執行する。株価が100円のときに90円の逆指値注文を出した場合は、株価が90円まで下がると自動的に90円での売却注文が執行されるため、損失が10円以上拡大することはない。買い注文の場合も考え方は同様で、値段上昇の流れに乗り遅れないよう、株価が指定した水準まで上がった場合に買い注文を自動執行する。
W指値注文	逆指値注文と指値注文を同時に執行する注文方式。
± 指値注文	前日終値、当日始値、現在値等の値段を選択し、その値段を基準に何円高くなったら（安くなったら）、買い注文（もしくは売り注文）が自動的に発注される注文方式。
連続注文（リレー注文）	2件以上の注文を執行する順番に注文予約登録を行い、最初に予約した注文が約定すると、次に予約した注文が自動的に発注される注文方式。
リバース注文	あらかじめ同一銘柄の注文と反対売買により利益が生まれる注文を登録しておき、最初の注文が約定したら反対売買の注文が自動的に発注される注文方式。
先頭指値注文	発注したい値段自体は指定せず、その時点の気配値より1値段有利な値段（買い注文ならば1値段安く）にて発注する注文方式。
優先指値注文	約定成立の確度をあげるために、発注したい値段自体は指定せず、その時点の現在値より1値段分約定がつきやすい値段（買い注文ならば1値段高く）で発注する注文方式。

表3　特殊注文の種類（例）

処理方式を実装する必要がある。寄り付き時にどれだけの処理性能が必要かを先ほどの仮想証券会社の例（表2）で推測すると、口座数100万規模の証券会社で寄り付き時に13,500件の特殊注文が想定される。値段条件の前提となる取引所からの時価データは、秒間1,500件以上、約定条件の前提となる取引所からの約定データは、68,000件が9時直前まで取引所側に待機しており、午前9時の取引所の売買時間が開始された時点で証券会社のシステムに一斉に送出される。取引所から入力されてくる電文を見ただけでも、秒レベルで考えれば、理論上は秒間69,500件の処理性能が必要となる。秒間69,500件と聞くと、これまでのシステム構築の常識では、数10億円レベルのコストで大型UNIXサーバを何百台並列処理させる必要があるのだろう、と想像する。しかし、後述するように米国ではPCサーバとキャッシュ技術を活用し、はるかに安価に数万～数十万の処理性能を実現している。

③アルゴリズミックトレーディング

昨今、日本でもアルゴリズミックトレーディングへのニーズが個人投資家からも聞かれるようになってきた。米国においては、アルゴリズミックトレーディングに関連する取引が市場の30％に達している。通常の法人向けのアルゴリズミックトレーディングでは、取引コスト軽減のためのタイミングに合わせて注文を細分化して高速に取引所へ注文を回送するため、取引件数の増大と取引の集中が発生する。法人向けと個人向けのアルゴリズミックトレーディングは基本的に異質なものになると考えられるが、取引件数が増大することには変わりはない。アルゴリズミックトレーディングを実装する証券会社側も、これを受ける取引所のシステムもより高速化する必要がある。取引所のシステムを例にとれば、旧来型のシステムを使っている取引所のシステムでは、システムの増設にも限界があり、性能的な拡張性が期待できなくなっている。一方で、後で説明する電子取引市場では、新しい技術により、より低コストで高速な執行能力が実現されている。また、米国のヘッジファンドなどは、証券会社が提供するDMA（Direct Market Access）サービスといった、直接的に注文を取引所に出すシステムを利用して、機動的な売買を行うようになってきた。オンライントレードも、投資家からの注文がコンプライアンス的なチェックを瞬時に行った後、取引所へ電子的に回送されるため、一種のDMAが実現されているともいえよう。日本の証券取引所でも、この数年のオンライントレーディングの増加により取引件数が5倍以上にふくらみ、システムが逼迫するといった問題が起こっており、抜本的な性能対策を実施している。取引所がより高速化されてくるに従って、証券会社側のアルゴリズミックのエンジンもそれにあわせて高速化する必要がある。こうしたシステムの性能要件とし

ては、注文の取引所への回送処理だけを見ても、ミリセコンドの単位でチューニングを行う必要があるほどの高性能さが求められるという。このような高性能な仕組みが必要になるアルゴリズミックトレーディングであるが、昨今は日本の個人投資家の間でも関心度が高まってきている。アルゴリズミックトレーディングやこれに類するシステムによる証券自動売買については、自作のソフトウエアで仮想的に自動売買を行って利益を競うコンテストも開催されており、千人以上の応募者を集めているという。

④ATS/ECN/PTS

オンライントレードのシステムにインパクトを与える要因として、ATS（Alternative Trading System）があげられる。これは証券取引所の外で行われる取引所と同等の機能を有する取引の場である。ATSは時代とともに進化しており、なかでも価格提供や売りと買いの突合せを電子的に行えるECN（Electric Communication Network）と呼ばれるシステムが有名である。日本でも1998年の取引所集中義務の撤廃にともない設立が可能となり、PTS（Proprietary Trading System）という名前で知名度を得ている。2006年、大手オンライントレード会社が夜間PTSを開始したのは記憶に新しい。米国では日本より先行してATSが設立されていたが、2005年にSECがマーケットルールを変更したことにより、その重要性が増してきている。

米国では、SECが主体となり、1934年の証券取引法をベースとして技術の進展や時代の流れに応じて、より公平で自由なマーケットのルールを構築してきた。基本的な思想は、米国内の証券に関するトレーディング基盤および情報開示基盤をNMS（National Market System）と位置づけ、時価等の公正な開示と、最適な執行値段での取引機会の提供を市場参加者に求めている。しかし実際には、技術や制度的な制約の中で課題も多かった。たとえば、1970年代のNMSに関するルールでは、全米の主要マーケットを電子的に接続し、最良執行価格がわかるといわれていたITS（Intermarket Trading System）というシステムの利用を義務づけていたが、スペシャリスト等のマーケットメーカといわれるブローカの手動による値付けとシステムによる自動的な値付けが混在しているといった状況もあり、必ずしも純粋な需要と供給で値が決まるものではなかった。しかし、2005年のルール改正により状況は変りつつある。つまり、昨今のアルゴリズミックトレーディングによる注文の小口化/大量化や、電子化されたPTSの広がりによる執行時間の短縮化といった状況によりITSを前提としたルールが陳腐化し、現実とそぐわない状況となった。そこで、SECは、2005年にそれまでのNMSに対するルールを改めた。これは、「レギュレーションNMS」という名で知られ、それまでのITSの

利用を前提としたルールから、ITS以外のシステムも利用ができるルールへと変更された。新しいルールのポイントとしては、以下4点があげられる。

1. Order Protection Rule

これまではスペシャリスト等のマーケットメーカの介在により、投資家の注文は必ずしもベストな価格で約定されていなかった。新しいルールでは、人手を介さずに、システムにより自動化された値付けにより公平に行われる必要性を謳っている。(フロアーベース市場からシステムベース市場へ)

2. Access Rule

ITSのように限られた手段で、限られた人にしか開放されなかった株価を、さまざまな接続ベンダーを通して、廉価(1株当りでみた情報コスト上限を明記している)に提供する必要性を謳っている。

3. Sub Penny Rule

株価が1ドル以上の場合は、呼び値を1セント未満にしてはいけないというルール。これは、ヘッジファンドや極端なアクティブトレーダが、約定が成立するように、他人の指値より1セントでも優位な条件を指定するといった経済的には意味のない指値の応酬による非効率性を避ける目的がある。

4. Market Data Rule

マーケットの価格データや取引データを販売して得られる収入は、その送出元である自己規制機関(取引所等)にも分配されるべきであり、また、取引所やブローカディーラはこれらの情報を自由に公開できる権利を有する必要があるというルール。これにより、価格の透明性を高め、不正売買を防ぐことが目的である。

この「レギュレーションNMS」により、各市場参加者は、最新のテクノロジーを取り入れ、これまでにない高性能の証券取引システムや市場システムの構築を実施している。一部のPTSは、最新のテクノロジーによりスピーディに大量の値付けを実現できるシステムを構築することにより、これまでの脇役的存在から、技術的には既存の取引所から主役の座を奪った形になっている。結果的には、主要な取引所もそれらのテクノロジーを手に入れるべく、NYSE(ニューヨーク証券取引所)のアーキペラーゴ買収や、NASDAQによるINET ECNの買収が行われた。同様の動きは、欧州でもMiFID(Markets in Financial Instruments Directive)という名でルール化が進められている。

証券会社側も、高速化と自動化が進む市場やPTSにあわせ、投資家に最良な執行条件を提供すべく高速な執行エンジンの構築を行ってきている。特にPTSに関しては、自社の

PTSで取引を成立させることで証券取引所から徴収される取引参加料金を軽減させる効果もあるため関心が高い。日本の証券取引所では取引参加料金が取引所の営業収益の4割を占めているという。こういった流れは、日本でも進行しており、大手証券会社による米国電子取引専門会社の買収や、証券オンライントレード会社および外資系証券会社によるPTS設立へとつながっている。

3．米国での事例

（1）業界/業種を超えた技術共有

以上見てきたように、日本で発生しているインターネット経由での大量アクセスの問題や、今後広まっていくと予想されるアルゴリズミックトレーディングやPTSに対して、米国では数年前から技術的な方向性が固まってきている。その背景の1つとしては、米国インターネットの技術の進歩があげられる。Google、eBayといった検索サイトやオークションサイト、1億人が参加する音楽系SNSサイト、さらには、インターネット上で何百万人もが参加できるオンラインゲーム等々、利用者がグローバルレベルに達している大規模サイトが数多く存在している。これらの大量アクセス処理を実践している企業で使われている要素技術が、ベンチャー企業等を介してプロダクト化され、さまざまな業種のシステムへも広まってきていることがあげられよう。今回訪問したのは証券系の企業だが、先方の担当者からは、有名なオークション企業や検索サービス企業といった異業種の企業名が参考事例としてあげられた。つまり、インターネットの世界では、通常のエンタープライズ向けシステムより、100倍から1,000倍のトランザクション処理能力を安価に実現する必要があり、それを達成するためには、業界や業種を超えて、インターネット技術という大きな枠組みで技術の共有化が行われている様がうかがえる。以下では、米国企業が実際にどのような技術で目標を達成しているのかを見ていく。

①アルゴリズミックトレーディング　A社

A社は、世界で10本の指に入るブローカレッジカンパニー。アルゴリズミックトレーディングのシステムを構築し、グローバルマーケットを対象にした取引を行っている。アルゴリズミックトレーディングでは、自社の注文によるマーケットインパクトを押さえる必要があり、平均約定単価や出来高加重平均単価をリアルタイムに計算しながら、大量の注文をタイミングよく、瞬時に取引所へ発注する必要がある。発注の待ち時間は、ミリセコンドのレベルを目指す必要があり、1秒以上かかることは、他社との競争での負けを意味する。また、発注に際しても、全市場から最良な価格で執行が可能な市場へと注文をルーティングする必要もある。サーバはPCサーバを使用しており、サーバ内のリアルタイ

ム処理はキャッシュDBを使っており、これにより1台のPCサーバでも秒間25,000件のトランザクション処理性能を実現している。(図3)

エンタープライズデータファブリック

サーバは、マルチサーバ構成で高速並行処理をしており、各サーバ間は、EDF(エンタープライズデータファブリック)といわれるキャッシュのデリバリシステムを採用し、口座情報等の全サーバが必要な情報は、瞬時に各サーバにデリバリーされ、格納される。これにより、システム全体では、秒間80,000件のトランザクション処理性能まで実現可能という。

また、キャッシュの利用方法も同期処理と非同期処理を適材適所に使い分けており、"レイジーアクセス"ポリシーを採用している。

JAVAフレームワーク

ソフトウエアはJavaで開発されており、フレームワークはスプリングを採用している。スプリングは、機能的に軽量なフレームワークとなっており、性能負荷がかかるような処理は自社で独自にチューニングしたものをアドオンすることが可能である。このよう

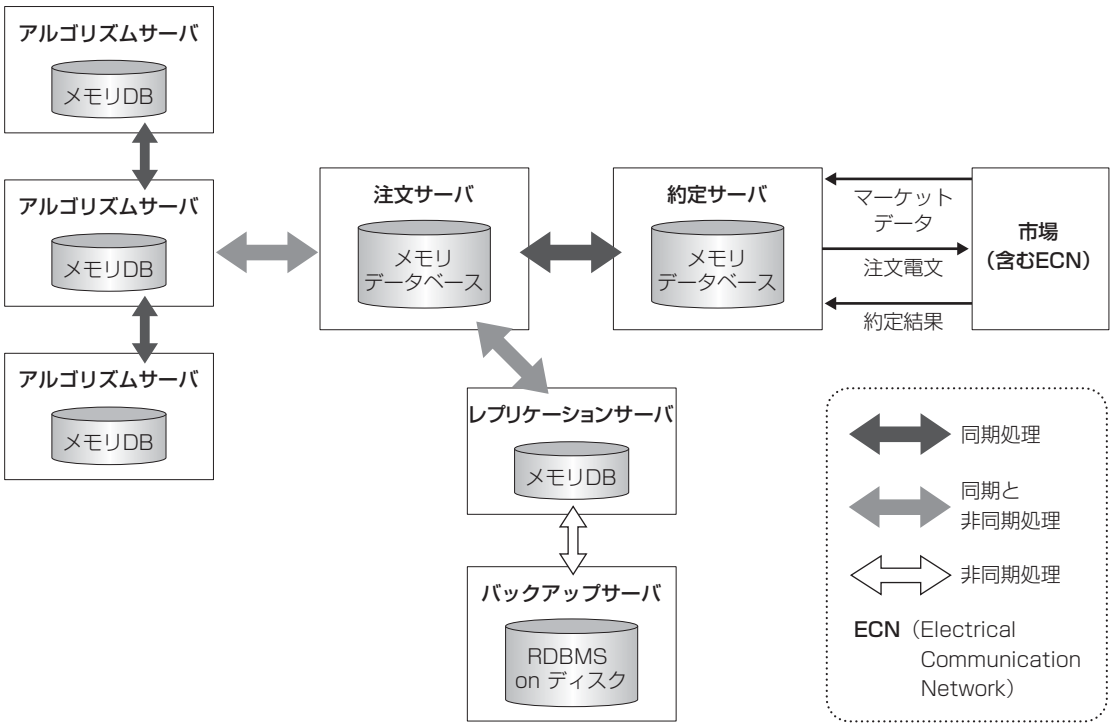

図3　アルゴリズミックトレーディング　概念図(A社)

な性能チューニングの自由度が高いことがスプリングを採用した理由になっている。他のフレームワークだと、実装されている機能も多く、チューニングする余地も小さいため究極的な性能向上が望めない。米国の大手オークションサイトもスプリングフレームワークを使い、1日10億トランザクションの処理を行っているとのことである。

小数精鋭部隊

開発体制は、20名ほどしかおらず、そのうち10名はアプリケーション開発者という。しかもこの10名は、各位が数学、アルゴリズミックトレーディング、Javaコーディングのプロフェッショナルとのことで、業務を理解したうえで自らコーディングまで行っている。他社にない差別化された仕組みを構築するためには、小数精鋭で集中的に構築することが有効という。さらに、こういった技術的な対応以外にも、サーバの設置場所を取引所に近い場所にすることでミリセコンドレベルでの性能向上が実現されていた。

②オンライントレード　B社

存続するメインフレーム

B社は、米国では老舗の大手証券オンライントレード会社である。業界でも早くからオンライントレードへ参入した経緯から、メインフレーム主体のシステム構成となっている。今回ヒアリングしたのは、その中でも公社債やミューチャルファンドにかかわるアセットマネジメントのシステムで、数万以上ある銘柄情報や公社債投信の利回り計算を行い、社内外のチャネルを通して情報提供している。近年特にインターネット経由でのアクセスが増加する中、メインフレームをベースとしたシステム構成のため、効率的なスケールアウト構成をとることができず、大きな問題となっていた。この問題に対して、A社と同様に、キャッシュメカニズムを活用することで問題を解決させようとしていた。メインフレームと、インターネットからのアクセスを受けるWebアプリケーションサーバと、メインフレームにキャッシュデリバリーシステムを導入した（図4）。

キャッシュデリバリーフロー

処理フローとしては、まず、既存の更新処理の結果、メインフレーム上の既存データベースに更新が発生する。メインフレーム上に導入したキャッシュデリバリーのエンジンが、既存データベースの更新情報をイベントとして検知し、メインフレーム上のキャッシュデータベースに複製する。同時に、XMLベースのイベントとして瞬時にフロントのWebアプリケーションサーバ上のキャッシュにメッセージング機構を通して更新情報が伝播される。つまり、メインフレーム上の既存処理はこれまでと同様の処理方式で行うが、キャッシュデリバリーの仕組みを導入したこ

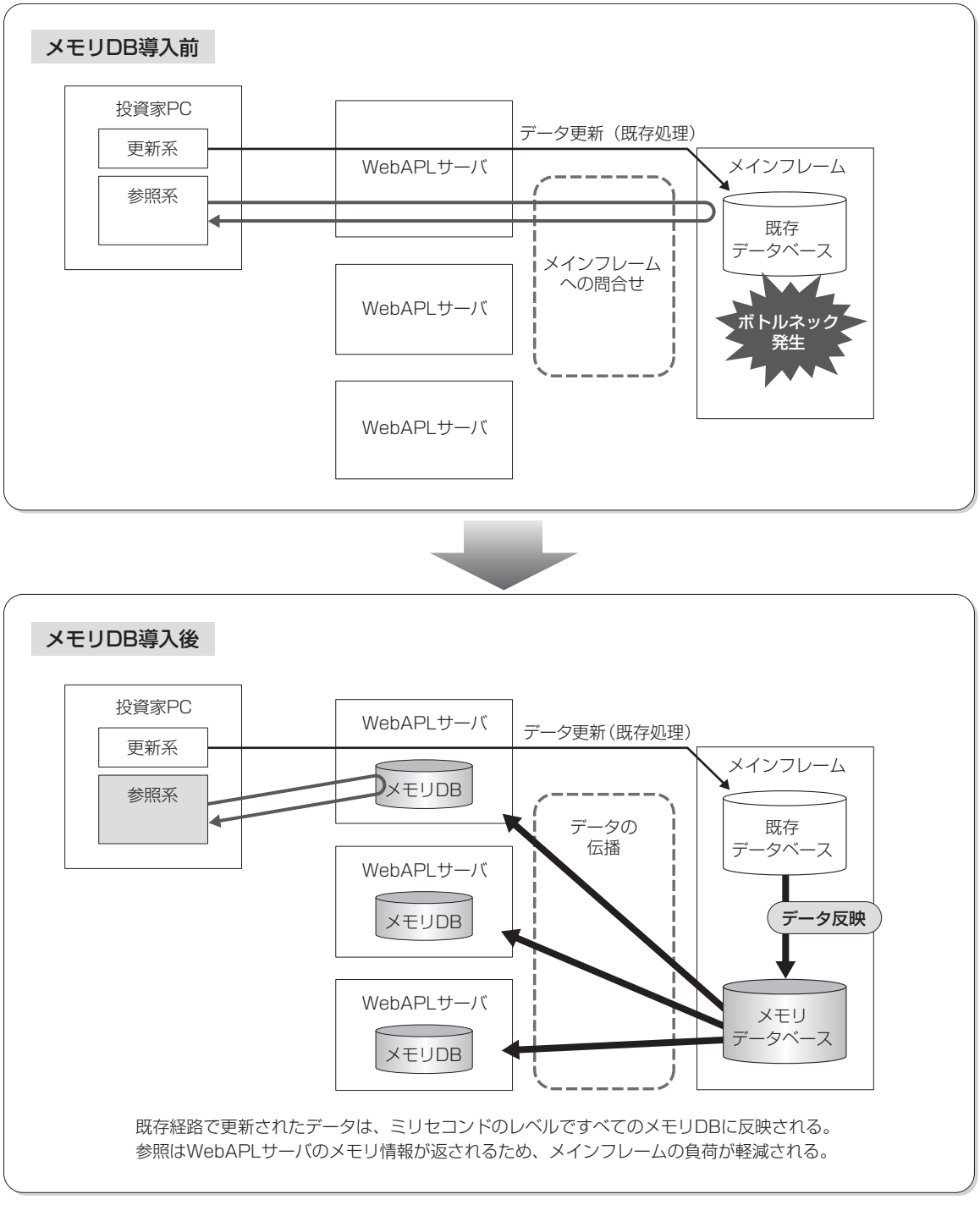

図4　オンライントレード会社（B社）

とで、処理結果はほぼ瞬時にフロント側のアプリケーションサーバ上のキャッシュに反映されるのである。このため、発生していたインターネット経由でのメインフレームへの大量アクセスが、Webアプリケーション上のキャッシュへのアクセスで解決されるため、メインフレームの負荷が減少し、リソース逼迫度が大幅に軽減される。また、インターネット経由の利用者からのアクセスも、フロントのWebアプリケーション上のキャッシュを表示するため、レスポンスタイムが格段に速くなる。前述のA社と同様、米国では、このように複数のサーバ上に配置されたキャッシュに対して、ほぼリアルタイムに更新情報を伝播させ、あたかも1つの巨大データベースのように見せる技術をEDFと呼び、急速に企業システムに取り込まれてきている。いわゆるグリッドコンピューティングの要素技術の1つである。訪問企業での利用方法を見ると、本来のグリッドコンピュータという枠を超え、その要素技術を活用しながら利用方法も多様化している。

③電子取引市場　C社

Windowsベースの取引所システム

C社は、電子取引市場を運営している。近年のアルゴリズミックトレーディングに関連する注文数の増大に対応するため、レスポンスタイムが数ミリセコンドの性能を満たす取引システムをWindowsベースで自社開発している。ベースとなるシステムはメッセージングソフトと自社製のメモリデータベースである。

現時点の処理要件としては、1秒あたりのトランザクション能力は1,000件のオーダーだが、レスポンスタイムについては、数ミリセコンドを達成している。

PCサーバにてシステムを構築する場合、現在はJavaか.NETかの選択肢が一般的である。C社の場合は、ベンダーからの手厚いサポートが期待できる環境にあるため、.NETによる開発を選択している。しかし、C社の技術者によれば、性能や信頼性の観点においては、Javaでも.NETでも大きな違いはないそうである。メッセージングソフトは、日本でも時価フィードの仕組みを実装する場合によく利用されるプロダクトを使用している。

自社製キャッシュメカニズム

C社独自の技術として自社製のキャッシュメカニズムがある（図5）。レスポンスタイム数ミリセコンドを実現するための鍵となる技術であり、ハード、ソフトの両面でさまざまな工夫がなされていた。キャッシュの仕組みを考える場合、メモリのサイズには限界もあるため、頻繁に更新データが発生する場合や大量データを扱う場合には、ハードディスクにより補完する必要がある。しかし、通常のハードディスクではアクセス速度がネックになり、結局はシステム全体の性能を劣化させ

てしまう。C社の場合は、メモリとフラットファイルの間でデータのやり取りを行う方法を採用している。フラットファイルというと通常のディスクアクセスがイメージされるため、メモリの補完的な役割を果たすかがポイントになるが、C社はSolid State Diskという通常のハードディスクの100倍以上のアクセス性能をもつディスクを採用していた。これにより、メモリをメインとした処理をしながら、一時退避用もしくはワークエリア用としてSolid State Diskを使うため、ハードディスクに起因するシステム全体の性能劣化を極力抑えることが可能になるという。メモリ上のデータは、通常のデータベースへ非同期でデータのバックアップをしている。その場合も、データベースへのIOネックによりシステム全体の性能を劣化させないよう、スーパーコンピュータでは良く使われるSAN（ストレージエリアネットワーク）の技術も用いているという。

4．処理方式のポイント

これまで見てきたように、米国における高性能を実現するための処理方式のポイントは、キャッシュ、イベントドリブン、メッセージング、マルチキャストの4点である。これらの技術により複数のPCサーバで、これまでは大型UNIXサーバでも達成できなかったトランザクション処理量を実現している。

（1）キャッシュメカニズム

キャッシュメカニズムについては、自社で開発する方法と、市販のプロダクトを使用する方法がある。自社で開発する場合には、高速ディスクを採用し、稼動する業務アプリケーションの特徴にあわせた方式を実装してい

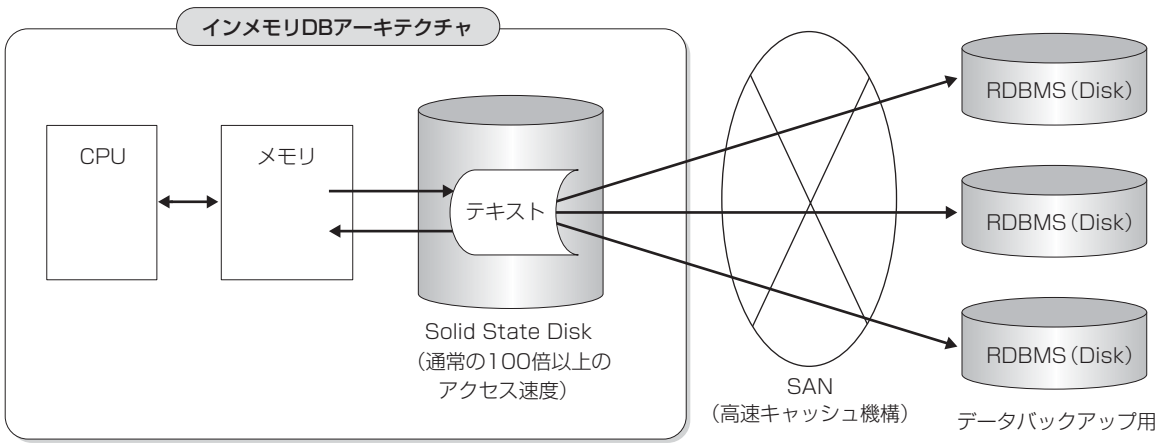

図5　電子取引市場　C社

る反面、リアルタイムに切り替え可能なクラスター構成まで実装するのは困難である。他の方法としては、市販のプロダクトを利用する方式がある。現状では、特徴の異なる2つのプロダクトを適材適所で採用している。1つは、単一サーバ内に完結したプロダクトである。特徴としては、メモリ上でハンドリングできるデータ量が大きいことが上げられる。高度なクラスター機能を持っており、バッチが多用されるシステムや、あまりスケールアウト構成を意識しない場合に適している。もう1つは、マルチサーバ構成にてキャッシュを共有できるプロダクトである。この場合は、一度に処理できるメモリサイズはあまり大きくないが、更新結果が、即座に全サーバに伝播されるため、性能面での拡張性があり、スケールアウト構成が可能という特徴を持つ。

(2) キャッシュの伝達技術

イベントドリブン、メッセージング、マルチキャストについては、マルチサーバ構成でキャッシュを伝播させる方式にも採用されており、システム全体で処理を1か所に集中させないために有効な手段となる。イベントドリブンについては、更新情報を自動的に認識し、更新情報の内容に従って、あらかじめ定めた処理へとつなげる役目をもつ。あらかじめ定めた処理としては、たとえば、他サーバへのイベントの通知や更新情報の伝達がある。情報を受けたサーバは、イベント受信処理により更新情報を受信し、次の処理につなげていく。イベントの受信と送信の間の仕組みはメッセージングにより実現するが、マルチキャストを実装することにより送信元サーバのリソースへ負荷をかけることなく、何台ものサーバへ情報の伝達が可能になる。

5. NRIでの実装事例(リッチクライアント)

NRIにおいても、個人投資家向けのトレーディングツールを開発し、本場運用されている。その中で、米国の実例で見られるような最新技術と同様の処理方式も実現している。

(1) 業務面での要件

個人投資家向けトレーディングツールは、売買系機能と情報系機能が一体化されており、顧客は、リアルタイムに証券時価、ランキング情報、ニュース情報等を見ながら、機会を逃さずに売買を出すことができる。

(2) 基盤面での要件

こういった基盤では、

①秒間1,500以上の取引所からの時価データを、遅延なく投資家のPCに到達させられるか。
②高速処理下においても障害に備えて時価データを保全できるか。
③利用する投資家が増大しても、スケールア

ウトできる構成になっているか。
④投資家のPC上で遅延なく時価情報を受信できるか。
⑤マルチチャネルの実装になっているか。

といったことが重要なポイントとなる。

(3) JMS/MDB技術の活用

システムの実現にむけてはJavaのエンタープライズ向け規格であるJ2EE（JAVA2 Enterprise Edition）を採用している。J2EE基盤は、すでに枯れた技術の感はあるものの、実際のシステム構築では有益な機能が使われぬまま眠っていることも多い。今回は、そういった機能の1つであるJMS（Java Messaging Service）とMDB（Message Driven Bean）というインターフェイスに注目し、システムを構築している。JMSは非同期メッセージングサービス機能を実現する。MDBは、メッセージをトリガーにして起動し、メッセージに応じた次の処理へと引き継ぐ。これらのメッセージは時価情報がカプセル化されたもの

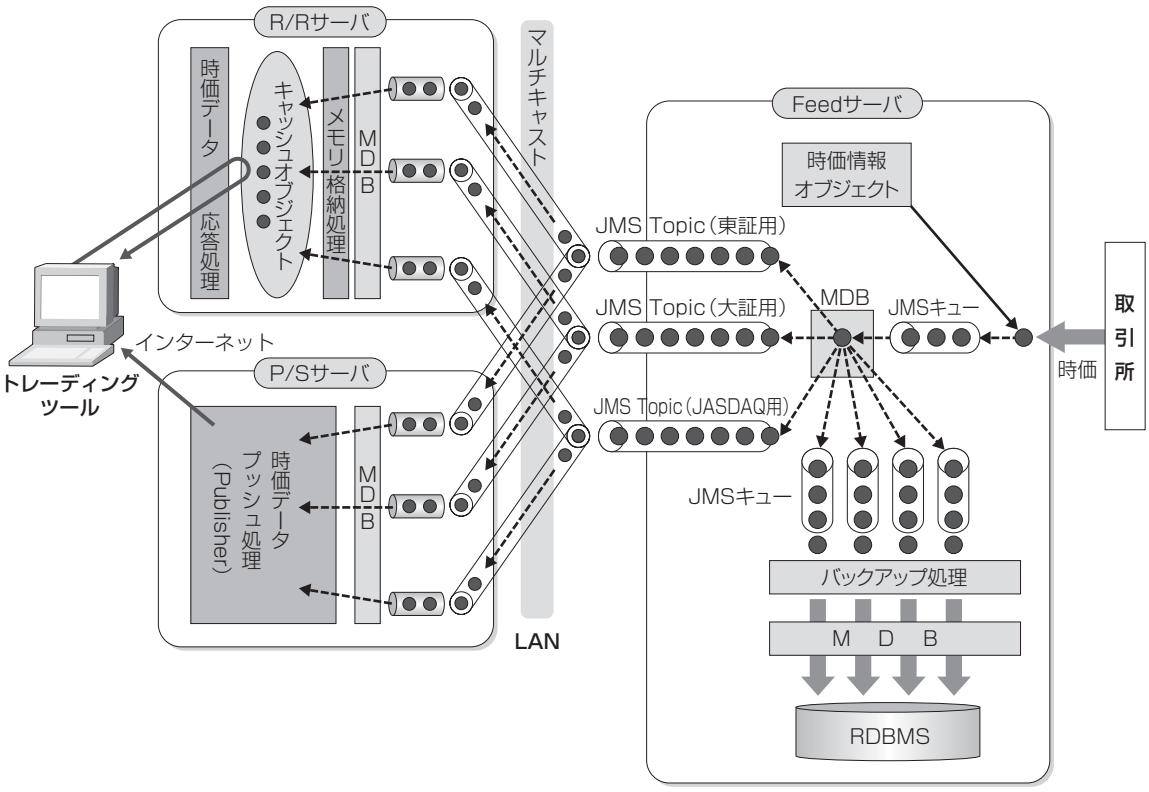

図6　NRIリッチクライアント　サーバ基盤（時価データプッシュシステム概念図）

で、市場別に紐付けされたJMSトピックと呼ばれるJMS上の経路を通って、各サーバへ伝播していくことになる。

(4) マルチキャストのメリット

先ほどのキーテクノロジの枠で考えると、新しく取引所から到達した時価情報は、メッセージを配信するFeedサーバ内に到達するとMDB（Message Driven Bean）がイベントを感知する。MDBは、そのイベントを伝播させるための経路の口であるJMSトピック（送信側）に情報を引き渡す。JMSトピックは取り込まれたメッセージオブジェクトを、必要としているすべてのサーバに対して送信する。このとき、実際の実装方式としては、ネットワークルータがもつマルチキャストの機構を使って電文が配布されている。マルチキャスト機構を使うことで、高性能ルータが、電文を物理的にコピーして複数台のサーバに配信するため、受信サーバの台数が増えても送信サーバの負荷は増大しないので増設の必要はない。以上の流れでフロント側のP/Sサーバまで到達した時価データは、インターネット越しに接続されているTCP/IP上のコネクションを通り、投資家のPCへ到達して画面に表示される。サーバ側では、障害に備えて時価情報を通常のデータベースにバックアップする必要があるが、全体の性能を劣化させないために、バックアップ処理を行うアプリケーションが非同期にMDBから起動されるため、全体の性能に影響を与えない。

(5) NRIの高速オンメモリ処理技術

これらの処理はすべてメモリ上で行われるため、取引所からの秒間1,500件以上のメッセージを受信しても、PCサーバのCPU利用率は80％以下で処理することが可能となる。投資家のPCへの送出能力についても、性能評価の結果1台のP/Sサーバが秒間24,000件でメッセージを送出することを確認している。アプリケーションの違いを考えると一概には比較できないが、米国の事例と比較しても処理方式自体には遜色はない。

(6) Nimbusフレームワーク

これらのキャッシュメカニズムやデータフィードの仕組みを実現するにあたっては、JBOSS上のNimbusフレームワークの存在が欠かせない。Nimbusフレームワークは、NRIが積極的に開発に参加したオープンソースであり、その要素技術は社内のJavaをベースとしたミッションクリティカルなシステムにも少なからず反映されている。今回のリッチクライアントのプロジェクトでは、J2EEのインターフェイスであるJMS/MDBについて、Nimbusフレームワークにより実装している。

(7) Nimbusの既存機能

①インテリジェントキャッシュ機能

RDBMS相当の操作性をもつPCサーバ上のキャッシュ機能。キャッシュの変更前と変更後のデータを保持し、アプリケーションの参照中の参照一貫性を保証する機能、行（オブジェクト）単位での挿入、変更、削除の機能、オブジェクトごとのタイマー設定による自動削除機能といったきめ細かいアクションが可能となっている。

②ロジックの部品化と簡易スクリプトによる部品の再利用化

Javaの分散アプリーション処理を実現する方式の1つに、J2EEの規格としてEJB（Enterprise Java Bean）がある。しかし、データベースへのアクセスがブラックボックス化されるといった問題があり、高性能が求められるシステムに対してはその利用を控えるケースも多い。Nimbusフレームワークでは、内部処理としては、EJBを利用しつつも、アプリケーションの実装方式としてEJBは前提とせず、Nimbus IoC（Inversion of Control）という仕組みで対応している（図7）。これは、Javaアプリケーションを部品化できるレベル（Bean）まで分割し、それらを外部から簡単なスクリプト（メインフレームでのJCL（Job Control Language）のイメージ）により呼ぶ出すことで実現される。また、部品（Bean）のフローをBean Flowとして記載する際に、トランザクション制御も自由に設定できる。アクセスするテーブルや送受信のキューもXMLにより定義できるため、これによりデータベースアクセスのブラックボックス化の排除と、生産性の高いアプリケーション開発を実現できる。

TPモニタの振り分け制御と同様に、最上位のクライアントからのコマンドに従ってサーバ側のロジックが動くため、各ロジックとトランザクションおよびフローが分離される。これにより、フローやトランザクションの変更はクライアント側に集約することができる。

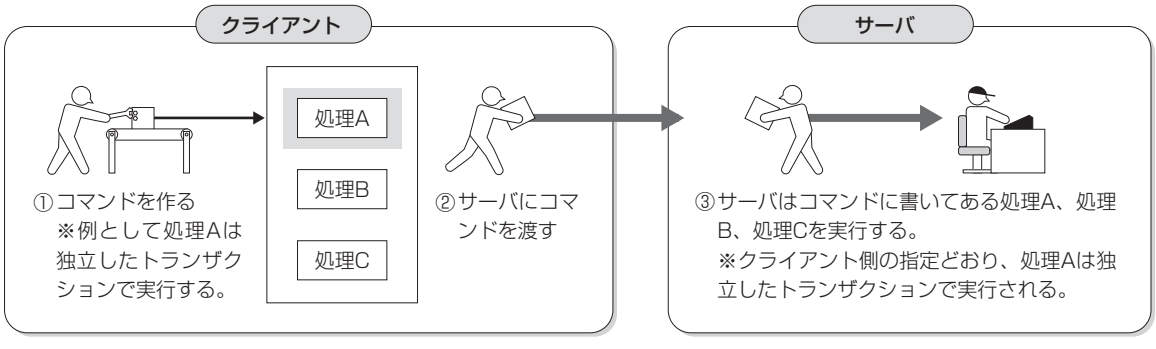

図7　Nimbusの概念　～ Inversion of Control ～

③ジャーナルログの変数レベルでの出力制御

Nimbusフレームワークにて稼動するソフトウエアは、必要に応じてジャーナル出力や変数単位でのログ出力のレベルを設定したり、実行されるSQL文の出力が可能である。これは、アプリケーションロジックの任意の場所に、既存ロジックに影響することなく処理を追加できるNimbusインターセプターというアスペクト機能で実現される。

今回追加の新機能

Nimbusフレームワークは、オープンソースという性格上、常に機能が追加されており、今回のプロジェクトでも、データのリアルフィード機能としてマルチキャスト配信やインターネット上のクライアントとの双方向通信や同期/非同期データフィードの仕組みを実装した。

今回は、Nimbusの既存機能であるキャッシュメカニズムと、今回追加したデータフィードおよびクライアントとの通信機能により、取引所からフィードされる時価データをデータベースに高速に書き込みながら、数ミリセコンドのタイムラグでフロントサーバのキャッシュにも反映させることができたのである。

米国の事例同様に、イベント、メッセージング、マルチキャスト、メモリキャッシュを重要な要素技術として活用しているが、主たる違いは、トランザクション量の規模と採用しているシステム基盤である。米国の事例では、アルゴリズミックトレーディングやPTSのように秒間10,000件から100,000件のトランザクション量を前提としており、障害時対策を含めた運用面での工夫には格段に違いがあると思われる。システム基盤の違いは、キャッシュデリバリーに特化したプロダクトを利用しているかどうかである。米国の事例では、ミッションクリティカルに適用されるシステムについては、メモリキャッシュ、イベント、メッセージング、マルチキャストの技術に特化した専用プロダクトを使うケースが多い。NRIのリッチクライアント基盤では、NRIが主体的に開発に関わったNimbusフレームワークを取り込んでいるが、Javaをベースとしているためアプリケーションの生産性が高い反面、性能を出すには基盤レベルでかなり高い人的なスキルが必要となる。今後は開発するシステムの規模や性能要件に応じて、キャッシュデリバリーに特化したプロダクトの選択も視野に入れていく必要がある。

6．さらなる大規模オンライントレードシステムへの対応

米国の事例や社内での構築経験は、今後、大規模オンライントレードシステムを構築する際の重要な示唆を与えている。今後実践として大規模オンライントレードシステムを構築する際に必要となるこれらの技術につい

て、NRIが現在取り組んでいるポイントについて説明する。 具体的には、

1. 注文と約定のトランザクション集中の回避
2. メモリキャッシュの活用によるレスポンス時間の短縮
3. 疎結合化によるサーバ単位での容易な機能追加

があげられる。3.については、XMLをベースとしたSOA(Service Oriented Architecture)による疎結合化の方式がすでに広まってきているため、ここでは高性能処理の実現にあたる1.と2.にフォーカスを当てる。

(1) 注文と約定のトランザクション集中の回避

前述の通り、寄り付き時には閑散時と比較して100倍以上のトランザクションが集中することもあり、そのようなトランザクションピーク時でも大幅な処理遅延が発生しないようにリソースを確保している。このピーク時のリソース利用状況が緩和されれば、そのまま、システム全体での必要なリソース要件も小さくなる。つまり、5倍の性能向上が実現されれば、これまで必要とされていたサーバ台数が5分の1に削減可能となる。これまでの米国での事例や、リッチクライアントの処理方式からみて、メモリキャッシュを活用した場合の性能向上にはかなり期待でき、同一コストで比較した場合、5倍の性能向上もターゲットになり得る。これを実現するための処理方式上のポイントは、注文サーバ、約定サーバ、口座サーバがそれぞれ自サーバで処理した結果を、そのデータを必要とする他のサーバに高速で転送し、メモリ上で持ち合うことにある(図8(B))。これまでのいわゆる枯れた方式では、これら3つのサーバが1つのサーバに集約されており、同一サーバ上で投資家からの注文入力にともなう注文処理、取引所からの約定データに関する約定処理、これをトリガーに発生する口座データ更新処理が同居していたため、RDBMS上でトランザクション処理が集中し、処理が頭打ちとなっていた。メモリキャッシュを使った場合は、注文処理、出来処理、口座更新処理を分散されたそれぞれのサーバ上で実行される。処理上必要となる他サーバの情報は、すでに自サーバ上のメモリに展開されているため、通常時間のかかる他サーバへの問合せは発生せず高速処理が可能となる。出来処理、口座処理についても同様のことが言える。

(2) メモリキャッシュ活用によるレスポンス時間の短縮

(1) のメモリ共有の範囲をフロントサーバまで拡大することで、投資家の問合せに対する応答はすべてフロントサーバ上での折り返しとなり、レスポンス時間の著しい短縮化が実現する。注文サーバであれば、処理後の注

文データをフロントサーバともメモリ共有する（図8（A））。約定サーバや口座サーバについても同様にフロントサーバとメモリ共有する。これにより、寄り付き直後に集中的に発生する出来状況のリクエストや、口座状況のリクエストが、すべてフロントサーバのメモリキャシュでの折り返しになるため、投資家にリクエスト応答を高速に返すことができる。つまり、投資家へのレスポンスタイム向上と、各サーバの処理負荷を軽減することができる。通常3階層からなるディスクアクセスを前提としたシステムの場合は、少なくとも3台のサーバを通るので、1～2秒ほどのターンアラウンド時間がかかる。一方、メモリキャッシュを活用してフロントサーバの折り返し処理で済む場合には、サーバ間通信の負荷や他サーバの処理結果待ちから開放され、数ミリセコンドから数十ミリセコンドのターンアラウンド時間が期待できる。ターンアラウンド時間の短縮は、サーバ内に滞留するトランザクションも削減され、リソースの余裕度も高くなるため、さらに受けつけられ

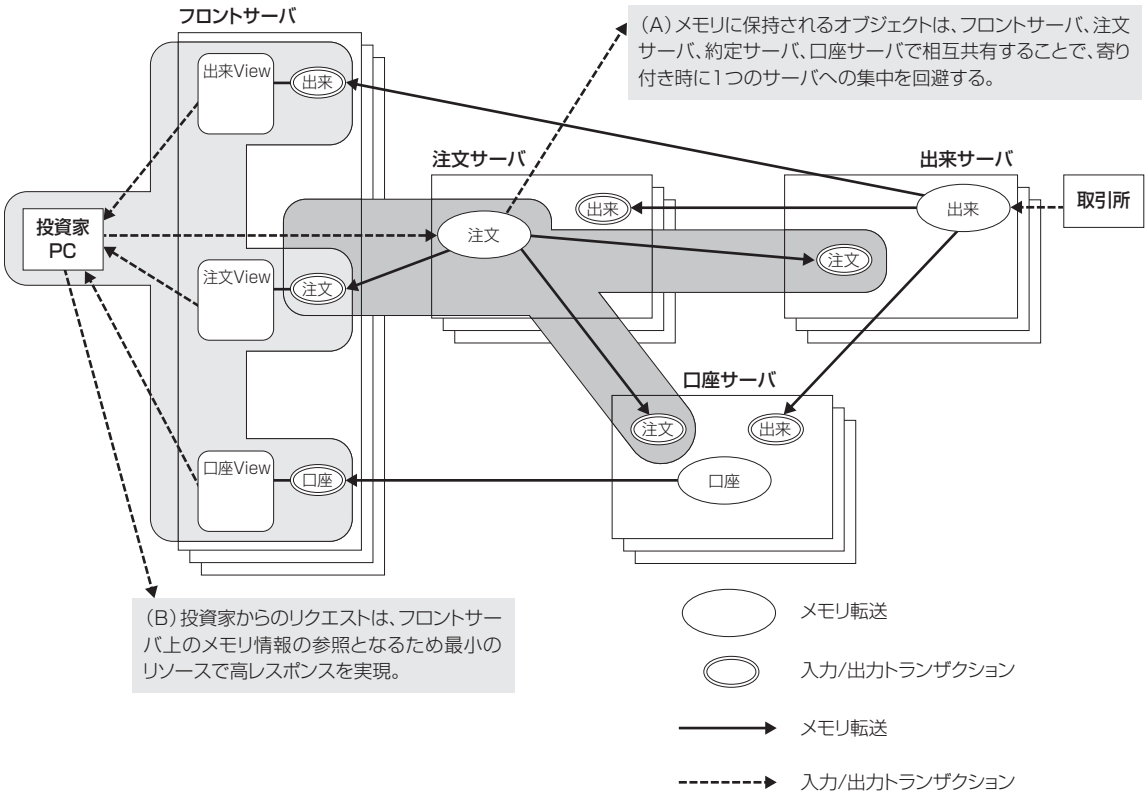

図8　メモリ共有によるシステム全体負荷の軽減

るトランザクション量が増大するという相乗効果もうまれる。

以上の2つの処理方式案は単純化したモデルだが、NRIではすでに実装に向けた設計を行っている。

7．おわりに

以上見てきたように、インターネット上の技術が進化していく中で、日本の証券業界ではオンライントレーディングシステムが技術的に先行してきた感がある。しかし、昨今の恒常的な利用者数の拡大や、本来のインターネット利用のメリットである低コストを維持しながらアルゴリズミックトレーディングやPTSといった高性能システムに対応していくためには、これまでの枠に捕われない新しい技術や挑戦的な処理方式を積極的に取り込んでいく必要がある。その1つの例がPCサーバ上のメモリキャッシュを活用して、秒間10,000件以上の性能を出すシステム基盤であり、処理方式といったものである。こういった技術をミッションクリティカルなシステム上で実現することで、これまで容易には解決できなかったデータベースサーバのスケールアウト構成を含む効率的なシステムの構築や、アルゴリズミックトレーディングやPTSといった超高速処理の実現が可能となる。米国ではそれらの技術や処理方式もデファクト化しつつあり、さらに先を目指す動きも見受けられる。日本では品質や障害に対する考え方が米国に比較して厳しい感があり、大規模システムになるほどリスクヘッジ的考えのもと既存技術を踏襲する傾向が強いように見受けられるが、昨今のオンライントレードを取り巻く環境を考えると、いよいよ技術的なターニングポイントに来ている感が否めない。