



**Contents — FINANCIAL MIDDLEWARESPECTRA — November 1997**

2.	<b>Middleware's place in the finance sector</b> Charles Brett, President, C3B Consulting
4.	<b>Middleware makes Financial Exchanges work</b> Mats Anderssen, CIO, OM Group
12.	<b>Why PC servers will not overtake mainframe servers anytime soon</b> Wayne Duquaine, Principal, Grandview DB/DC Systems
18	<b>Schwab: reverting to 2-tier — and saving on middleware</b> James Chong, Vice President, Charles Schwab
24.	<b>BZW: deploying a global financial middleware structure</b> Kieron Drake, Application Architecture Manager, BZW
30.	<b>Introducing middleware to Westpac</b> Arthur Travers, Manager, Strategy and Design, Westpac
36.	<b>Enterprise integration, information flow and middleware</b> John Mann, Vice President, Research, MIDDLEWARESPECTRA
44.	<b>Infrastructure dominates financial service opportunities</b> Michael Killen, President, Killen & Associates
50.	<b>Processes + transactions = distributed applications</b> Gustavo Alonso, Senior Researcher, Swiss Federal Inst. of Technology
55.	<b>Intranet/Enhanced Intranet Options</b>

# Middleware makes Financial Exchanges work

**Mats Anderssen**  
Senior Vice President and Chief Information Officer  
OM Group

## Management introduction

*The OM Group is the derivatives exchange in Sweden. It has two main lines of business:*

- *a 'transaction business' which runs exchange and clearing houses, as well as providing facilities management services (a subsidiary provides 25 banks and brokers in Sweden with back-office services)*
- *a technology business which builds, designs, supports and operates the exchange and clearing house technology which is used to run the OM Stockholm Exchange and which has since been adopted by some 10 other Exchanges worldwide, including those in Hong Kong, Sydney and Milan, as well as the American Stock Exchange in New York.*

*In this interview, Mats Anderssen — the CIO for the OM Group — discusses how middleware technology (primarily RTR) has been developed and deployed to support financial exchanges where transaction integrity as well as real time processing is required. In addition, he talks about how the OM technology is being adapted for non-financial trading — for example for trading in Norwegian electrical power futures and a similar deployment occurring on the Californian Power Exchange.*

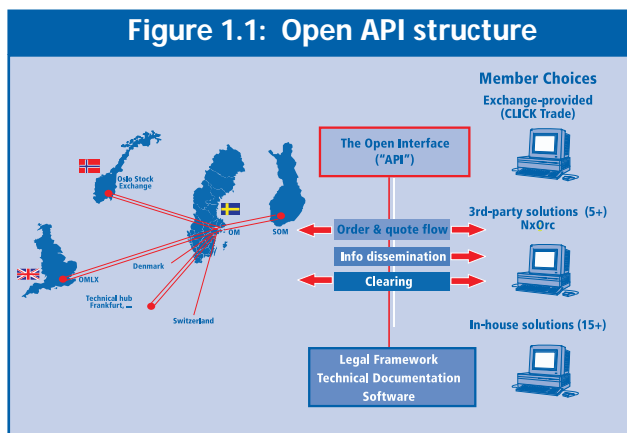
## Planning for the next generation exchange

The genesis of our latest generation of technology started in 1989 when we realized that we needed to provide fully electronic trading. Quickly we understood that the demands of such automated matching and clearing required a completely new structure for the future.

Although the exact reasoning is in the past, we chose to use what we believed would be a highly flexible client/server-oriented architecture. We went out to the market to look for an existing software product solution because we understood that our business knowledge domain — where we add the real value — is in providing the know-how about how to build futures and derivative markets and supporting systems rather than middleware per se.

That said, our heritage was in using Digital's VMS, on a homogeneous basis. It made sense to move to OpenVMS as our back end server technology because we already had extensive experience with standard products like RDB — so long as we could meet the key requirements for an Exchange, primarily:

- **transaction integrity**
- **reliability (to 99.9999% or 99.99999% uptime)**
- **stability**
- **manageability.**



In the Exchange business, credibility — as in banking — is all important. Those trading — whether investment houses or individual investors — must possess confidence that their trades and actions will be executed as ordered. Any degree of uncer-

tainty or unavailability can cause tremendous damage in a very short space of time, as well as the loss of transaction income to an Exchange.

What did assist us back in 1989/1990 was that availability, in all senses, had 'only' to be provided for a relatively short period each day — typically a six hour trading day. That trading day is gradually becoming longer. We saw, even then, developments in other markets which pointed towards a need to provide round the clock mission critical services.

For example in California today, where electric power provision is being deregulated, we see a much longer trading day being necessary for the Californian Power Exchange (PX). This is not yet 24x365. But we are edging closer, especially as Exchanges look to link themselves together to provide out-of-hours trading across multiple time zones.

All this mandated that we have a crystal clear strategy for disaster recovery, whether the crisis is:

- **damage by fire**
- **an explosion**
- **terrorist activities**
- **simple system failures**
- **etc.**

To achieve this we had to build for full scale and fast failover. When you have to be able to give full recovery for financial trading applications 'running in place', the attraction of using a client/server basis becomes clear. Not least it offered multiple server support.

In addition, we decided to add what we now refer to as an open API (Figures 1.1 and 1.2). With this we aimed to make development easier both for ourselves as well as for the customers of an Exchange — the banks and brokers. Indeed, one of the old dependencies which had hampered us was that these customers often needed our assistance to implement and link our applications to theirs.

By providing an API — with security embedded — on a number of client platforms, these banks and brokers could integrate our trading flows into their internal back office and internal order routing systems without human intervention. By minimizing the dependency on us, we have been able to focus on constantly improving our trading products and services.

## Choices

In retrospect we made a number of decisions that were either clever or lucky. Our timing was good because client/server technology was clearly just arriving. We also understood that building our own home-created middleware for reliable transaction serving capabilities was something that would demand man years of our effort and at least as much for testing and proving.

Using these as principles we decided that it was preferable — if at all possible — to buy a product rather than build in-house. Therefore, out we went to see what was available.

When we looked at the market, Tuxedo was the obvious candidate along with a number of products that barely exist today. Then, almost by accident, we stumbled on Digital's Reliable Transaction Router (RTR) — even though we were already a Digital user (RTR was well hidden, even then). RTR had the attraction that it:

- was transactional
- did not tie closely into a database.

If this last seems odd — particularly in a client/server environment — you must understand that we wanted a clear cut, reliable and secure communication channel between:

- a number of servers (typically 10+, to cover failover, etc.)
- multiples of clients (hundreds, even thousands, spread across many different organizations).

RTR gave, and still gives, us:

- reliability
- availability
- failover

in software at a time when hardware fault tolerance was expensive. This mattered because — if the total solution was costly — then the per transaction cost would need to be expensive which would deter trading activity and diminish underlying liquidity.

RTR, from day one, gave us the capabilities we needed. We could operate as a small Exchange and yet grow tremendously fast without pain. We could run all the Exchange financial products in one server to start with and, when volumes outstripped that server, we could partition the applications and data across multiple servers.

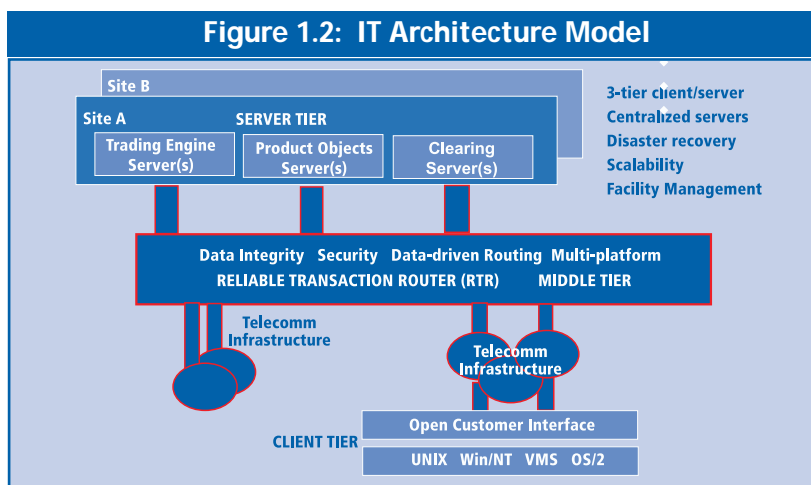
For instance, taking a simplistic example in an equity trading system, equities A to E could be on one server, F to N on a second and the rest could be on a third. The nature of RTR facilitates this — and the associated backup and recovery — because RTR acts as our middleware software system infrastructure.

## The database dimension

Now returning to the database dimension, we need high performance transaction throughput. This means that we need to hold large amounts of data in main memory. We are typically seeing order rates of 50-100 order transactions per second.

We keep the order book, which is the main 'data dispenser' in a mainstream trading system, in memory so that we can offer very short turn-around times. As Figure 1.3 shows, on a typical day we have around 250,000 transactions per seven hour trading window with an average response time of around 0.2 of a second. This is measured from:

- the bank or broker client — where our interface receives the original buy or sell order



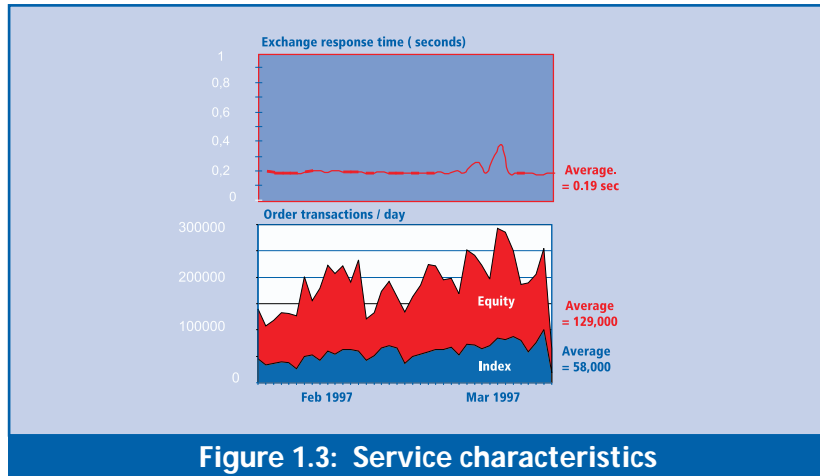
- London (for example) to Stockholm into the central server where the trade is matched
- Stockholm back to London (again, for example) — when it offers the response (to the interface at the Exchange Member's site) about whether the order closed, was partly filled, became a trade, etc.

In effect, RTR is our database and it supports a pan-European wide-area network. That indicates why we need RTR's level of sophistication for handling failover as well as multiple sites. If something falls over then we depend on RTR because it can automatically continue from servers located elsewhere (Figure 1.4).

Of course we had already put in place — not least for regulatory reasons — an audit trail log. We could have always re-played from that log, although this would always have been slower and infinitely more cumbersome. Instead, the basic facilities of RTR offer us:

- flexibility, especially across multiple systems (for example, we have data driven routing so transactions can go to different servers depending on content)
- high throughputs and short turn-around times
- connections to a high speed memory database (developed in-house)
- automated failover and recovery

without our having to develop the basic middleware to enable all this. With RTR we could build our applications on top of an existing transactional infrastructure.



### Development and deployment issues

We use an object oriented database — on top of Oracle's RDB — to hold the definition of financial products, user characteristics, market rules and regulations. That is used as a source by our main memory, high performance databases which:

- hold the details of the marketplace and make these available for dissemination ...
- provide market data to the traders.

We develop primarily in C and C++ for performance although we have many new clients being developed in Java (and we will shortly have NCS connected to a single client NT server which will run our API into our RTR servers). We expect much of our future to be developed, especially as we move to accommodate browser-based clients without losing our transaction capabilities. We also see that, over time, we will probably introduce more of the server products based on Java.

In this context it is, I think, worth mentioning RTR's location independence, particularly at the transaction level. Our developers can issue a transaction and not worry where it will be processed. In the context of client/server application development, this is a real bonus. Removing the need for developers to have to know which processes work where in the RTR network is a real productivity boost.

In consequence, we have been able to extend transparently the OM Stockholm Exchange with a real time link to our Exchange in London (OMLX). It is to this infrastructure that our client API talks.

Of course as with all client/server implementations, testing was and remains complex — especially with our set up consisting of:

- **multiple servers**
- **hundreds of clients**
- **thousands of financial instrument and settlement contents.**

Throughout we had to be rigorous with all the transaction implications as well as fail over scenarios. The deployment advantage that RTR brought us was:

- **we did not have to test RTR itself (as we would have had to do if we had built our own middleware)**
- **the ability to separate the different pieces enabled us to structure application testing in ways which would probably not have been possible in a host-type solution, although client/server introduces its own complexities.**

Over time what has consistently helped us is that we have come to know RTR inside out. We are past the threshold of viewing it as complex. It has become part of our culture and we only wish that it was more widely sponsored by Digital as being one of the best — in our view — middleware products available in the market place.

### **At the user end**

While we use the facilities of RTR extensively, we do not need to present these to our users — the Members of the various Exchanges that use our software. As I mentioned earlier we developed an API which effectively:

- **hides much of the RTR complexity**
- **adds security between client and server (we envelope the data, including authentication)**
- **offers an interface which is financially-oriented.**

This interface is what presents the financial prod-

ucts, prices, trade details, etc. to the user (trader). It is very much application and industry-oriented and it is this that we now support on several platforms, including:

- **UNIX (HP-UX, Solaris and AIX)**
- **Windows NT**
- **OpenVMS**
- **etc.**

Using this approach we actively try not to lock Members into our choice of platform or screen. In the past it was a terminal into the Exchange. Today, exploiting client/server, Members can:

- **choose their platform(s)**
- **write to our interface**
- **integrate their selected platforms into their existing and future systems in-house (without our involvement).**

In turn, this enables Members to introduce 'straight through processing' — from trade to settlement. This can hugely cut costs because the human element has been reduced to the bare minimum. Therefore, while Members can build their own applications, part of our strategy has been to promote the creation of a portfolio of commercially available applications software which add functions and features above our API.

Currently, there are at least 15 third party-sourced applications and we provide several others, including Research and Trade, a small OM-majority-owned company with products which are popular in the market. It provides highly advanced market maker portfolio analysis workstation software.

### **Transaction implications**

In our environment the transaction is sacrosanct. We must be able to deliver it with full integrity. The way we ensure this from the client is that we always commit — using RTR — at the server. As part of this, the application which issued the original order always obtains a commit status back from the server.

If the transaction commits at the server but the client 'has been lost' (for whatever reason), the originating application will look for the status

when it returns. If a would-be transaction fails on transmission between the client and the server, it has not acquired 'transactional status'. This is only obtained once the server accepts the transaction.

In the transaction flow, the links from our central servers to clients are real time. It has to be this way to maintain context. We monitor RTR all the time. If a trader puts an order in and the client 'disappears' but meanwhile the market moves, we have to act. The trader may want to delete the transaction but we have rules that can be set up so that orders which 'belong' to clients that have disappeared are automatically deleted from the order book.

A further, if background, check comes with our demographics. We tend to have a relatively small number of clients — despite the high number of transactions. In Stockholm we have around 250 clients from about 40 organizations. Similarly, in the various Exchanges using our software — Sydney, Milan, Hong Kong, etc. — we have roughly between 800 and 1000 clients connected (in total) or about one to three screens per organization. This means that it is reasonably straightforward to resolve any problems which do occur.

At the same time, one of the attractions is that we can support any size of organization — from tiny to massive. This is reflected in the way in which we charge Members. We do not use a per-transaction charge. Instead we charge per API — meaning that all our client software has a definable upper limit on the number of transactions per second it can put through. This means that we are charging for potential peak capacity rather than actual use.

This has one specific benefit. It enables us to plan and configure our systems to the peak level which the total of Members want. On the other hand we have to remember that in our type of market we have automated trading behavior — more or less expert systems which are triggered on prices. One buy order can spawn dozens if not hundreds of smaller transactions.

That is not all. Rules can be set up to trigger price changes in related instruments on the buy side or on the sell side of different strike prices — all of which is handled by RTR's transaction facilities.

At yet another transaction level, our interface provides the transaction flow, including the destination, prices and information for clearing and

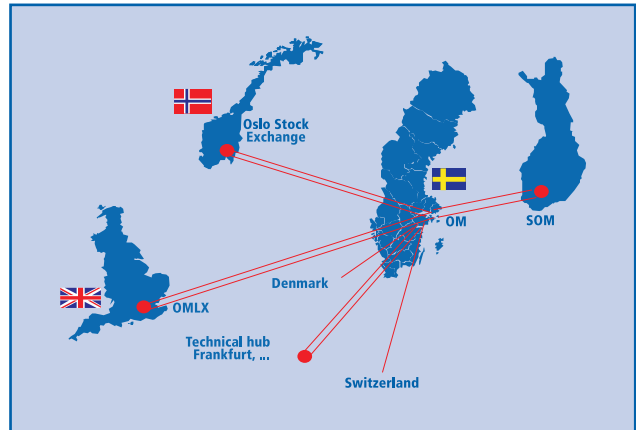


Figure 1.4: Location flexibility

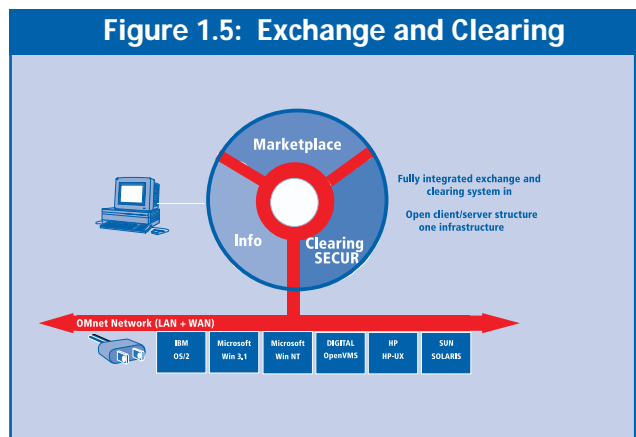


Figure 1.5: Exchange and Clearing

Figure 1.6: Transaction services

	Exchange				Clearing			
	Business Operations	Facilities Management	System & Support	Dev't	Business Operations	Facilities Management	System & Support	Dev't
OM	X	X	X		X	X	X	
OMLX	X	X	X		X	X	X	
PULPEX	X	X	X		X	X	X	Started 5/97
NordPool		X	X		X	X	X	
EL-EX		X	X		X	X	X	
Oslo SE		X	X					
OTOB			X				X	
AMEX			X					
Milan SE			X					
HKFE			X					
ASX			X					Start 10/97
PX			X					Start 1/98
TOTAL	3	6	12		3	5	6	

settlement information. We have integrated the functions of the OM Exchange and clearing house so that the same transaction structure holds the flow:

- **from order**
- **to trade(s)**
- **through clearing**
- **to settlement.**

This is wholly paperless. It is handled through the same API. What is also often overlooked is that this is more than just software and its documentation. As with other Exchanges this embraces a legal framework covering security, method of operation — and liability.

### **Multi-location, inter-enterprise capabilities**

A dimension which I have not discussed so far is the capability which RTR offers for multi-location — even inter-Exchange — operation. As I have discussed above, RTR gives us automated fallback and failover. Necessarily, to make this workable, we have multiple sites with multiple machines running. This is what provides the speed, resilience, recovery, etc.

If you follow the implications through, this enables us to link to other instances of RTR. For example — although not yet implemented — the various Exchanges using OM Group's software could link all their systems and provide trading between markets. Sydney might be accessible to Stockholm and/or Milan — with a minimum of additional technical effort (the legal and other implications would be more difficult to solve).

In one sense we are already utilizing this. In Stockholm we provide the facilities management for the Oslo Stock Exchange and the Norwegian market. In Norway they have a full terminal and client infrastructure, but the servers are located here — physically distinct from our Swedish market.

In another variant, we are providing the services for the Norwegian, Swedish and the Finnish Electricity Exchanges. Electricity is a commodity, in the same way as oil or pork bellies. In terms of trade processing, electricity is little different from other futures.

Scandinavia is one of the first places where electricity has been fully deregulated. With multiple

sellers and buyers, it is possible to establish a fair market. In such circumstances — as either consumer or producer — you can offer prices to create a transparent market.

This is not limited to Northern Europe. It is, for example, also happening in the UK and in the USA. In California, we are building the systems to support the California Power Exchange.

In effect, we are using our financial expertise plus knowledge of middleware and highly reliable infrastructure to create an integrated and automated market for those buying and selling power. We saw the opportunity to deliver an expertise which was unfamiliar to the power industry. We have the know-how to build automated exchanges and to provide the infrastructure to make these markets liquid. But, at the end of the day, the technicalities and procedures are much the same as for a financial market — or even internally within an organization which needs 'internal trading'.

### **Lessons learned**

Early on we decided, as policy, not to mix the middleware structure with data content. We have stuck to this — to our advantage — and have kept the network (what we call OMNet — RTR plus our security solutions) separate from, and ignorant of, data content.

This gives operational flexibility in that we have been able to add servers and applications without requiring code changes in the infrastructure. We have also seen that infrastructure has a longer life span than server applications themselves.

On a less positive note we had to learn the hard way about RTR and its complexity (it is not intuitively simple). That we use almost 100% of the functions and features means we are very much dependent on it as a supported product. If we ever wanted to swap out RTR, this would be technically possible because of the separation of middleware and data/applications. It would also be a major undertaking, for nothing else in the market place — after a feature-by-feature comparison — can provide the same real time transaction processing which RTR gives us.

My last lesson is that we have learned that middleware needs overt management facilities. You cannot leave it alone. You have to know the status of transaction flows, of the middleware itself as well

---

as the applications and the underlying operating system(s). To us, the middleware management domain should be able to be integrated with traditional network and system management tools.

As a business, we have standardized on BMC's Patrol and have integrated network system management as well as all application events and failures into one colored map/console. RTR does not fit into this because it has yet to be integrated with SNMP alarms. This is a feature we feel users should demand, especially if you are running high exposure — or enterprise — middleware.

### **Management conclusion**

*There can be few such high pressure environments as an automated futures and options exchange. Not*

*only is there financial complexity in the extreme (multiple instruments, swaps, derivatives, etc.) but individual transactions can be for many millions of dollars. None can be lost.*

*At the same time, such markets depend on real time access to information and execution. Without execution there is no liquidity.*

*As Mr. Anderssen describes, building a dependable and trusted environment to deliver all this is not simple. It needs middleware, especially if it is offered seamlessly, to provide location and process independence with transaction integrity. The OM Group has achieved this — and spread it out to 10+ Exchanges world-wide. That is no mean achievement for a piece of middleware that Digital barely acknowledges.*

---

---

**Members of the International Advisory Board**

---

**Charles C.C. Brett**

President, C3B Consulting Limited

---

**Michael Killen**

President, Killen & Associates, Inc.

---

**Kathryn Dzubeck**

Executive Vice President,  
Communications Network Architects, Inc.

---

**Fiona A. Winn**

Managing Editor & Publisher

---

**Dale Kutnick**

President, Meta Group, Inc.

---

**Paul Hessinger**

Vision UnlimTed

---

**Pierre Hessler**

Deputy General Manager,  
Cap Gemini Sogeti

---

**H. William Howard**

Vice President, Inland Steel Industries, Inc.

---

**Ellen M. Hancock**

---

**Norris van den Berg**

General Partner, JMI Equity Fund, LP

---

**John E. Mann**

Vice President, Research

---

**MIDDLEWARESPECTRA**

---

**Philip Manchester**

Consulting Editor

---

**Additional contributors include:**

---

**Francis X. Dzubeck**

Communications Network Architects, Inc.

---

**James V. Franch**

System Software Associates

---

**Keith Jones**

IBM

---

**David McGoveran**

Alternative Technologies

---

**John Tibbetts & Barbara Bernstein**

Kinexis

---

**Amy Wohl**

Wohl Associates

---

**Martin Healey**

Technology Concepts Limited

---

**Allan Lees**

Software

---

**Dennis Ford**

NobleNet

---

**Les Yeamans**

North American Systems Group

---

**Gary Weis**

Advantis

---

**David Sutherland & June Hacker**

Carleton University

---

**Jonathan van den Berg**

Premier Software Technologies

---

**Tom Heywood**

University of Southampton

---

**Eric Leach**

ELM

---

**Glen Macko & John Parodi**

Digital Equipment Corporation

---

**Randy Rhodes & Troy Terrell**

Black & Veatch

---

**John Carter**

IBM UK Laboratories

---

**Roy Schulte**

Gartner Group

---

**Jim Johnson**

Standish Group

---

**Tom Curran**

TC Management

---

**Alfred Spector**

IBM Corporation

---

**Max Dolgicer**

International Systems Group, Inc.

---

**David Baer**

Consultant

---

**Jerrold M. Grochow**

American Management Systems, Inc.

---

**Ken Orr**

The Ken Orr Institute

---

**Peter Houston**

Microsoft Corporation

---

**Jeff Tash**

Database Decisions

---

**Ed Cobb**

BEA Systems

---

**Bernard Abramson**

Merck & Co.

---

**Mirion Bearman and Kerry Raymond**

CRC for Distributed Systems Technology

---

**Oliver Sims**

Integrated Object Systems

---

**Jim Gray**

Microsoft Research

---

**Pierre Jouanny**

Thomson Software Products

---

**Wayne Duquaine**

Grandview DB/DC Systems

---

**Steve Craggs**

Candle Corporation

---

**Colin White**

DataBase Associates International

---

**Gustavo Alonso**

Swiss Federal Inst. of Technology

---

**Peter Mark**

Lotus Corporation

---

**MIDDLEWARESPECTRA is published and distributed worldwide by:**

---

**USA and Canada:**

Spectrum Reports, Inc.

---

**Research Office**

64 Hudson Street  
Somerville, MA 02143, USA  
Telephone: 617 628 8225  
Fax: 617 623 4639

---

**Subscription Center**

PO Box 301368,  
Escondido, CA 92030, USA  
Telephone: 1-800-933-5997  
Fax: (760) 432 6560

---

**UK and Rest of the World:**

Spectrum Reports Limited

---

**Research and Editorial Office**

St Swithun's Gate, Kingsgate Road  
Winchester SO23 9QQ  
England  
Telephone: +44 1962 878333  
Fax: +44 1962 878334

---

**Subscription Centre**

St Swithun's Gate  
Kingsgate Road  
Winchester SO23 9QQ  
England  
Telephone: +44 1962 878333  
Fax: +44 1962 878334

---

**Email and Internet**

Email:

[spectrum@middlewarespectra.com](mailto:spectrum@middlewarespectra.com)

World Wide Web:

[www.middlewarespectra.com](http://www.middlewarespectra.com)

---

**ISSN 1460-7220**