

## ApDev Next Rung on Grid Ladder

**LONDON**—As compute grids are now well established in investment banks, and data grids and server virtualization continue to develop, investment banks are turning their sights to application development for the new grid environments, say industry participants.

“We are currently focusing on time-to-market,” says Juan Lando, director of the Grid Center of Expertise at Merrill Lynch. “It used to take a couple of weeks to integrate new application code into the grid, but now it can take as little as two hours.”

Not everything is so easy, explains Lando. For a completely new application where extreme performance is required and budget is available, it might take 10 to 15 weeks for a proof-of-concept, followed by a planned conversion and migration.

“It all depends on what performance gains businesses want to achieve with grid,” he says. “Each application is assessed on its merits.”

For example, Lando points to one of the bank’s applications that handles CDO2s, or a collateralized debt obligation (CDO) based on another underlying CDO. The original application in Microsoft Excel was rewritten in C#, runs on 1,000 nodes and experienced a run-time reduction from five to 10 hours down to two seconds.

JPMorgan is also reducing lead times for on-boarding its new applications. “We believe standardization is a key for that,” says Mike Ryan, chief technologist with JPMorgan’s investment bank architecture team.

JPMorgan is creating a grid service-oriented architecture (SOA) to simplify

the programming model for deploying software on the grid.

“Rather than having application teams slice and dice the calculations upstream of the grid, they can submit the whole thing as one unit,” Ryan says. “We have a wedge of code in the grid middleware that knows how to decompose that one job into smaller tasks, then distribute those smaller tasks to the grid compute workers. That wedge of grid middleware code hides a lot of grid complexity from the line of business application teams.”

### Hiding the Grid

Many banks start by putting their analytics libraries on the grid. One method used by an unidentified London-based investment bank involved standardizing the library across all of the desks, and putting the grid behind a standard C++ interface that accesses the grid via a fast, but hidden, message bus, says Vivake Gupta, managing director and co-founder of consultancy Lab49, which helped with the project. “It is now rolling out to credit and equity desks,” he adds.

A second method of accessing a grid transparently is through Excel plug-ins, which is a useful strategy since traders do not really care about the grid as such, but just want to develop their models quickly, says Matt Davey, director of technology in the U.K. for Lab49. “Utilizing the grid is often as simple as clicking a check box to select local or grid-enabled calculation,” he says.

However, there are many ways to invoke the grid, Davey says. “In the case of portfolio pricing, certain

investment banks will run the calculation simultaneously on the grid and locally on the desktop, throwing away the slower answer. Others may choose dynamic strategies for deciding when to invoke the grid—for example, for long credit derivative calculations,” he says.

A third strategy for hiding the grid is to implement grid-enabled, third-party applications. Adam Vile, head of grid, high-performance computing and technical computing with London-based consultancy Excelian, has helped banks do this. “Increasingly, third-party software providers like Murex, Calypso and Sophis are gearing up for grid with adapters for the major grid schedulers or even their own distribution architectures in some cases,” he says.

Financial software vendor Misys is also starting to move its products to the grid with a new release in March of its trading and analytics product, Misys Summit. Dan Cohen, product manager with Misys, describes it as “grid out-of-the-box.”

By using smart distribution techniques, the platform can split up portfolio calculations based on calibrated run times in user testing, Cohen says. “This way we have managed to benchmark performance improvements of 15:1, while making the grid totally transparent to the user,” he says.

There are three key issues when porting software to the grid: “Dividing the calculation for distribution across the grid, load balancing to optimize elapsed times and managing a consistent set of reference data,” he says. This requires considerable software intelligence and a

calibration process for each bank to tune the software to its own portfolios.

However, issues of consistent reference data go beyond that of a common version, say industry experts.

“All the nodes need to use the same snapshot of data, especially if you have a data grid cluster that is being continuously updated,” says Lab49’s Davey. “Speed of data access can also be an issue; caching strategies will need to be considered.”

Demand for such strategies can only increase. By hiding the grid, Excelian’s Vile says, “people can focus more on building new models and creating new products, even on a daily basis. This is where they gain competitive advantage.” Vile also says that the industry will start to see new scripting languages emerge to support this, where use of the grid is built in.

However, as the CDO2 application at Merrill Lynch demonstrates, an explicitly parallel program design is sometimes essential and hiding strategies are not enough, according to industry participants.

“People are using these kinds of new abstraction mechanisms to avoid the problem of coding for the grid,” says Lab49’s Gupta. “But it only goes so far.” To gain the full grid savings, he insists the industry needs to redesign its applications.

## Thinking Parallel

Parallel programming is a paradigm shift, says Gupta: “Software is not really parallel until it is designed parallel.” However, he says that good grid software is as different from conventional programming as English is from Mandarin Chinese. “Hiding is only an intermediate solution until we get university students trained to think parallel,” he adds.

Thinking parallel starts with the user interface (UI), says Gupta, who sees the Windows Presentation Foundation (WPF) from Microsoft .Net 3.0 facing the same problem. “Before, we used to program in pixels. Now we need to use

vectors. It is a different paradigm and thought process and so requires a shift in how you approach UI development,” he says.

“All data presented on the screen should be made, in some sense, interactive, and this interactivity makes it inherently more parallel,” Gupta continues. “In effect, we have a grid of cells on the screen that can each initiate a separate server action. You declare the vector, which kicks off the cell processing beneath it.”

---

**“It used to take a couple of weeks to integrate new application code into the grid, but now it can take as little as two hours.”**

Juan Lando, director, Grid Center of Expertise, Merrill Lynch

Nevertheless, Gupta says that people are already moving in this direction on .Net 2.0, by using the toolkit known as the Consolidated UI Application Block (CAB). “It allows you to build little windows that are tied down to server components, so you can execute on the server side in parallel and in effect encourage desktop consolidation,” he explains.

For Gupta, CAB is just another factor increasing the use of SOA and grid. “It breaks the monolithic client-side application down into parallel components,” he says. “A number of sell-side firms have started to use it to merge three or four desktop applications into one trader cockpit.”

“CAB services offer a natural façade to grid-enabled services,” says Lab49’s Davey. “The CAB event broker provides a loosely coupled event system, encouraging asynchronous communication with a grid.”

Part of thinking parallel, say industry insiders, involves viewing the world as a stream of events.

Indeed, data grid vendors are deeply immersed in events. Cameron Purdy,

president and CEO of data grid platform provider Tangosol, is very clear: “The next big challenge is event-driven architectures, taking a high-level, systemic approach to managing event streams, by constantly recalculating thousands of predictive risk scenarios. Here, information change leads to rapid adjustments and more changes: The process is infinite.”

Michael Di Stefano, vice president and architect for financial services at data fabric vendor GemStone Systems, agrees: “Distributed processes need distributed data architectures based on event-driven, rather than request-reply, paradigms.”

“The data management itself needs to be policy driven—cache nothing, cache only keys, cache a subset of content or all content,” says Di Stefano. GemStone’s aim is to be able to support millions of transactions per second across a Milliband network, he says. “In practice, we have already achieved 200,000 to 300,000 meaningful transactions per second.”

The problem is that both transaction and event stream processing applications require a high level of interconnect between the nodes, points out Excelian’s Vile. “Compute grids can’t handle this, so we see the rise of data fabrics or specialized interconnect technologies,” he says.

Multi-core technologies are also important, say the experts.

The discussion has become more interesting because of the sharp increase in market data, says Lab49’s Davey. “Once again, we need a change of mindset to implement the orchestration of the larger number of parallel activities and interconnections,” he says. “We are starting to see a few large U.S. houses beta-testing some early products. Market prediction and the distribution of processing power will also be important factors in the success of this technology.”

*Bob Giffords is an independent banking and technology analyst and can be reached at bob.giffords@btinternet.com.*