Modernizing your Financial Risk Infrastructure with IBM Spectrum Computing

IT professionals can provide a shared infrastructure for a broad range of financial applications in risk, analytics, machine learning and more

Why read this report?

Financial risk applications are evolving fast as banks grapple with new regulations, vast amounts of data, concerns about cyber-security and changes in how applications are built, deployed and managed.

Advances in artificial intelligence and deep learning promise to revolutionize the customer experience. As applications built using containers make applications more portable, and cloud services become more secure, accessible, and economical, organizations are increasingly turning to the cloud.

In this fast-changing environment, the watchword is flexibility. To ensure competitiveness, banks need applications need to be agile, portable, and deployable across multiple clouds, and be easily integrated with services not yet invented.

Key takeaways

Risk workloads are evolving - Banks need to support a wider variety of applications including big data analytics, machine learning, and AI.

A proliferation of frameworks – While once found only in areas like Monte Carlo simulation, distributed applications are now pervasive. Spark, Hadoop, NoSQL, and other frameworks all rely on distributed compute and storage clusters. The number of tenants on the enterprise grid continues to grow.

Multi-cloud, hybrid computing is here – The mobile revolution, coupled with a proliferation of inexpensive cloud-services and pre-trained AI models is driving applications to the cloud. It is more clear than ever that banks need to plan for secure, burstable, cross-cloud deployments.

Environments must be container-ready – As organizations embrace micro-services, infrastructure environments need to keep pace, orchestrating not only legacy workloads but big data frameworks and modern containerized applications as well.

Contents

Introduction

It's been ten years since the financial crisis of 2007/2008, and banks have spent much of that time absorbing lessons learned, adjusting capital requirements, and putting practices and systems in place to avoid similar future failures. While future crises are inevitable, good risk management can help reduce their frequency and severity and provide banks with a source of competitive advantage.

As we approach the close of the second decade of the twenty-first century, traditional risks have not gone away, and new risks loom large. Sources of risk include things like climate change, political unrest, cyber-attacks, and increased concerns about reputational risk in a hyper-connected world where viewpoints spread like wildfire. Forward-looking organizations are becoming more vigilant about security, and leveraging new techniques like deep learning and behavioral science to help predict and pre-empt new sources of risk before they can impact the business.

New sources of risk and changes in how applications are deployed are causing IT professionals to re-evaluate their infrastructure

In this paper, we discuss how new business requirements and technologies are impacting infrastructure decisions. We introduce IBM Spectrum Symphony, a middleware solution built for modern risk workloads, and explain how it can help enterprises efficiently manage the increasing number of applications that rely on distributed computing environments.

Beyond Monte Carlo

Addressing new sources of risk

Techniques like Monte Carlo simulation have been an essential risk management tool for decades. Banks routinely rely on grid computing environments to calculate the future prices of financial instruments based on various scenarios over multiple time horizons. Examples of applications that involve simulation are market risk, credit risk, quants & modeling, capital adequacy stress testing, CCAR, CVA, XVA, FRTB, and dozens of other compute and data-intensive applications.

New sources of data are giving banks the opportunity to take a broader view of risk, incorporating more inputs into increasingly sophisticated predictive models:

Big Data - big data techniques like text analytics and sentiment analysis can help surface new inputs used in a risk model.

Streaming, real-time data - A predictive model may rely on real-time quotes, prompting the use of streaming frameworks like Spark or Kafka to perform analysis on the fly and populate a distributed Cassandra data store.

Iterative, recursive workloads - For risk and machine learning applications, calculations frequently need to iterate recursively and *fan-out*. Traversing a netting tree, probability treee, or recursive neural networks (RNN) are good examples. As recursion goes many levels deep, it is possible to run out of "slots" (compute resources that can be scheduled) leading to deadlock conditions. Grid schedulers need the sophistication to yield slots to child tasks and re-use them as a recursive calling sequence unwinds. While this sounds arcane, a simple feature like this can dramatically impact the feasibility and cost of running a simulation.

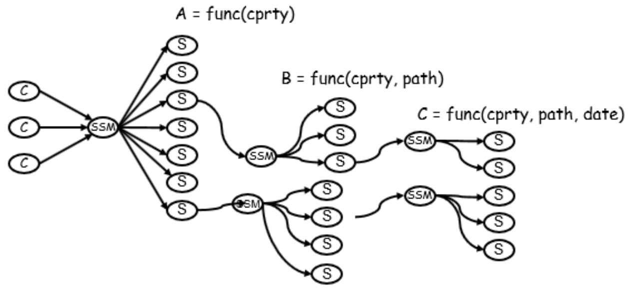

Figure 1: A variety of risk applications require efficient support for recursive workloads

Machine Learning and Deep Learning

Machine learning is a field of computer science that helps predict outcomes based on historical data. Managing risk is all about learning from the past to predict the future, so not surprisingly, machine learning has become a useful technique for many financial applications.

While fraud applications previously relied on statistical methods that scored static risk factors, machine learning enables predictive models that can evolve, detecting fraud with higher accuracy, and avoid costly false positives that can impact customer service and revenue.

Besides risk, other applications including customer service bots, anti-fraud tools, and AI advisors rely on machine learning techniques that depend on scalable compute clusters for model training and serving. Rather than deploying separate clusters for each application, banks are looking for better ways to share resources among these different workloads.

There isn't a single approach to learning from data; rather there are dozens of categories of machine learning algorithms that lend themselves to specific use cases.

The variety of algorithms is why it is important that a grid manager be able to support many different machine learning frameworks concurrently.

Deep learning algorithms can help identify new insights useful in making market predictions, managing risk, and more – but deep learning is notoriously compute intensive.

One widely used technique in machine learning involves the use of artificial neural networks that can be trained via supervised learning techniques to predict outcomes. With modern GPUs, it has become practical to construct large, multi-layer neural networks. We refer to these as "deep" neural networks; the term "deep learning" has come into its own as an important category of machine learning.

Unlike machine learning algorithms that require feature engineering (where data scientists pre-select features that will contribute to the predictive quality of a model), deep learning algorithms can deal directly with raw data and determine features themselves giving them better predictive qualities.

The trade-off is that deep learning models require data scientists to choose from multiple permutations of configuration parameters making deep learning models exceptionally difficult to train. Because of this, data scientists often resort to brute force methods exhaustively evaluating models using different combinations of hyper-parameters until the optimal set of parameters is found.

Deep Learning can detect relationships in data that humans will almost certainly miss. For example, a portfolio manager might have access to thousands of funds and individual securities, each having a prospectus and being subject to continuous sources of new financial news and market events that affecting the value of each security. While a human can't keep up with news about every security, a deep learning model can help identify new linkages between market news and fluctuations in securities values, leading to better predictive models that can help portfolio managers make better investment decisions and reduce risk.

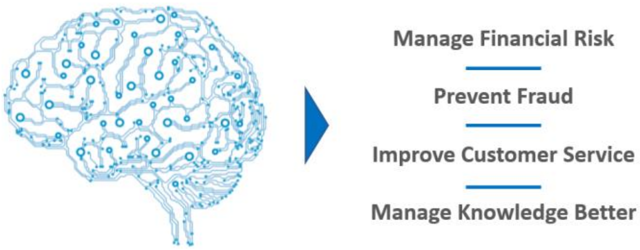

Figure 2: Machine learning and deep learning are central to new financial services applications

New requirements for data

In traditional financial risk simulations, parameters describing various market scenarios and details about financial instruments are passed to a compute service (or engine) that calculates a result based on a mathematical predictive model. Although machine learning can be used to refine the model, the basic approach remains the same. By computing results over millions of scenarios, instruments, and timesteps, and aggregating results across various dimensions, analysts can obtain a picture of likely future results, and assess various types of risks.

As electronic data related to our transactions, activities, and even our movements has become easier and cheaper to collect and store, analytic models have become more data intensive. Data still resides in file systems and relational databases but is also increasingly stored in scaled-out distributed frameworks including Hadoop file systems, object stores, or NoSQL databases such as Cassandra, HBase or MongoDB.

Spark is preferred for new workloads

With so many data sources and formats, Spark has become a preferred tool for efficiently manipulating large datasets. In fact, Spark is already more widely deployed than MapReduce in enterprise big data applications despite being relatively new, released as an Apache project in 2014.

Spark is compatible with Hadoop HDFS, supports SQL processing, stream processing, and graph processing, and uses Resilient Distributed Datasets (RDDs) to efficiently and reliably store objects in memory across a cluster. RDDs can be created from a variety of data sources including text or CSV files, JSON, NoSQL stores (Cassandra and MongoDB), Hive, various HDFS file formats and even Amazon S3. Spark also exposes a simple programming model and various built-in parallel transformations ideal for filtering, sampling, joining and aggregating datasets. Developers can choose to program using Spark SQL, use Spark's native functions, or even combine both techniques in the same program.

Spark's in-memory processing coupled with parallel transformation and analytic functions makes Spark up to 100 times faster than MapReduce for common big data operations. Even when datasets are too large to reside in memory, for multi-stage ETL operations Spark can be up to ten times faster than MapReduce owing to its ability to store intermediate results in memory.

Delivering between 10 and 100x the performance of MapReduce for data transformations, Spark is essential for data-intensive applications

---

Spark's built-in functions make it particularly powerful for data transformation. Built-in transformations including *map()*, *groupBy()*, *reduceByKey(), filter(), sample(), join()* and other operations are widely used in ETL. As new regulatory requirements such as FRTB and BCBS impose new data aggregation and reporting requirements on banks, and banks seek to employ techniques like machine learning requiring significant pre-processing of data for model training, the importance of Spark is hard to overstate.

The widespread use of Spark imposes new requirements on grid managers:

- First, the grid manager needs to be able to orchestrate Spark services, so that they run alongside other distributed frameworks like HDFS and service-oriented middleware without conflict.
- Because languages like Python, Scala, and Java are used with Spark, the service-side / engine API for the middleware needs to support Python to allow grid services to access distributed Spark data, call Spark's built-in parallel functions, and run various Spark SQL queries.

- Finally, Spark workloads need to be treated as first-class citizens on the shared, multi-tenant grid. The workload manager needs to be able to share resources according to policy among online and batch workloads implemented using Spark along with other programming languages and frameworks.

The rise of cloud computing

HPC environments for financial risk have been historically deployed on premises for a variety of reasons including cost, proximity to data, software licensing considerations, and the proprietary nature of the datasets and algorithms involved. As cloud offerings continue to evolve, these barriers are falling away, and a variety of factors are compelling banks to re-assess how cloud offerings fit into their infrastructure strategy:

- Increasing price-competitiveness of cloud services
- Shortages of data center space or power
- Improved security of cloud offerings
- Access to utility cloud-services that would be costly to deploy on premises
- Challenges finding skilled people to manage complex data center environments

Often, new applications depend on cloud-based frameworks. For example, machine learning frameworks and pre-trained AI modules delivering capabilities like image recognition, speech-to-text, sentiment analysis, and natural language processing are readily available in the cloud. These services would often be cost prohibitive or impractical to deploy on local clusters.

Similarly, some information about customers and their financial transactions are gathered from electronic sources outside the corporate firewall, making data convenient to process in the cloud.

For at least some workloads, banks may be considering public clouds to address specific use-cases:

- *Cloud bursting* – augmenting capacity on busy local clusters to accommodate peak or periodic workloads to avoid new capital investments.
- *Hybrid implementations* **–** running core services locally, but tapping cloud services where it makes sense giving banks the flexibility to shift workloads to where they can run most economically.
- *Selective outsourcing* – moving specific applications to the cloud as it makes sense. For example, applications that depend on “born-in-the-cloud” data may be best served by cloud-based deployments whereas applications interfacing with bank systems-of-record might best be deployed on-premises.

Ideally, banks need middleware solutions that help them stay flexible supporting applications locally or in their choice of public cloud depending on how business needs evolve. Some considerations are:

- **Avoiding cloud lock-in**: Banks would like to stay neutral and easily shift to different cloud providers based on capabilities, geographic or regulatory considerations.
- **Transparent bursting**: Cloud bursting should be minimally invasive to applications to avoid new development or training costs. Applications should be able to dynamically provision and use cloud infrastructure in a fashion that is transparent to application users.
- **Support for modern container frameworks**: Many banks are adopting containerized, micro-services architectures and CI/CD approaches for new application development. For these application environments, Kubernetes is a popular choice, and most cloud providers offer Kubernetes PaaS solutions in addition to their traditional IaaS and cloud HPC offerings. Examples include Amazon EKS, Azure AKS, Google’s GKE and IBM’s Cloud Container Service. The grid management middleware should be able to run natively on VMs, bare-metal hardware, and on Kubernetes clusters to maximize deployment flexibility.

Performance and scalability

Now more important than ever

Calculating various risk measures via simulation is a highly compute-intensive requirement, and banks compete based on their ability to run models quickly.

New requirements like FRTB (Fundamental Review of Trading Book) and other BCBS (Basel Committee on Banking Supervision) regulations are placing more demands on banks, and requiring that new risk measures be computed.

The capacity of a grid manager to efficiently run risk models is a function of both latency and scale. With the increased needs for computer modeling, stress testing, and sensitivity analysis, banks need grid middleware that can deliver high throughput and maximize resource usage to keep costs low. The grid software also needs to facilitate efficient resource sharing between different tenant applications.

Resource sharing

In modern risk environments, deploying separate clusters for each application, line of business, or software framework is simply not practical. Not only do siloed environments dramatically increase cost, they also constrain flexibility and prevent applications from sharing resources and data.

Ideally, businesses need to be able to share resources between applications, organizations, and across different software environments. Whether workloads are jobs, tasks, containerized workloads, or long-running services underpinning frameworks like HDFS, Cassandra, or Spark, the grid manager needs to facilitate flexible resource sharing both on-premises and across clouds.

New requirements demand smart middleware

To address these new business and technical challenges related to risk, customers are re-evaluating their approach to grid middleware. Enterprises need the middleware to be not only fast and scalable, but they need to it to support the full breadth of frameworks on-premises or in the cloud.

IBM Spectrum Symphony

IBM Spectrum Symphony is an enterprise-class grid manager for distributed application services on a scalable, shared, heterogeneous grid. It can support up to 5,000 compute nodes, 128,000 cores, and 300 applications. It has the flexibility to adapt to changing priorities and can reallocate more than 1,000 compute engines per second to different workloads according to sharing policies and application priorities you define. Spectrum Symphony delivers better application performance, better utilization and can respond quickly to business demands.

Spectrum Symphony helps banks:

- Accelerate time-to-results with faster throughput and performance
- Achieve higher levels of resource utilization
- Seamlessly blend on-premises and cloud-based clusters
- Gain the agility to respond easily to new business requirements

IBM Spectrum Symphony addresses the various technical and management issues that have historically made resource sharing difficult. It helps banks to get the most out of their infrastructure investments while ensuring quality-of-service and resource guarantees to each application tenant.

Software environments supported include:

- Data-analytic frameworks including MapReduce, Spark, Kafka, Storm
- Distributed data stores including Cassandra, MongoDB, and various other NoSQL stores
- Recursive workloads often required for modeling counterparty credit risk or machine learning workloads
- Machine-learning and deep-learning frameworks including Caffe, Tensorflow, H2O.ai and other popular AI environments
- Popular collaborative notebooks including Apache Zeppelin, Jupyter and R Studio
- Traditional service-oriented applications, batch workloads, or multi-step processes flows implemented using multiple programming languages and software environments

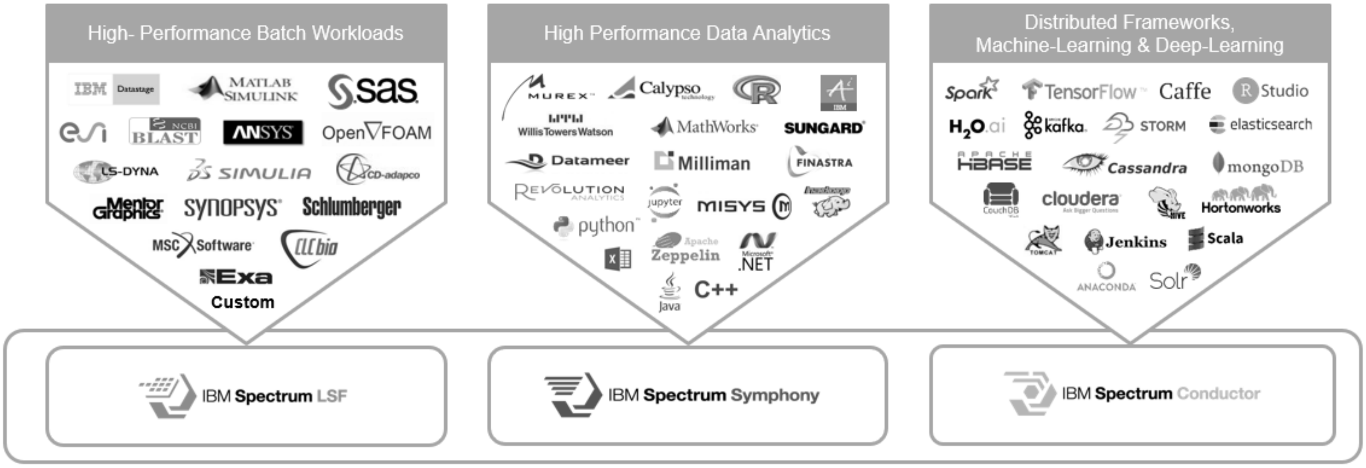

Figure 3 - IBM Spectrum Computing supports the full range of workloads needed in modern risk environments

Leading Financial Risk solutions

In addition to supporting a wide variety of in-house developed risk applications and third-party frameworks, Spectrum Symphony supports leading financial risk solutions including:

- Murex
- Xcelerit
- IBM Algorithmics
- Microsoft (Excel)
- Gemstone
- Sophis Risque
- Milliman
- Misys
- Calypso
- Towers Watson
- Mathworks
- SAS
- GGY
- Fermat
- SunGard
- PolySystem

By supporting commercial ISVs and a wide array of programing languages and frameworks Spectrum Symphony helps ensure that customers can easily adapt to future requirements and support new workloads without the need to deploy additional infrastructure.

Your Cloud, Your Way

IBM Spectrum Symphony also offers deployment flexibility across clouds, helping customers manage applications that span multiple geographic regions and cloud providers, shifting workloads automatically to where they can run most efficiently.

- Run on your choice of public or private clouds using bare-metal, virtualized or container-based deployment models
- Automate workload-aware cloud-bursting with built-in connectors to AWS, Microsoft Azure and the IBM cloud and auto-provision cloud resources based on policy
- Deploy containerized, ready-to-run Spectrum Symphony clusters to your choice of container environment on-premises or in the cloud including popular Kubernetes environments

Proven, secure, production-safe

As application architectures evolve, banks are running an increasing variety of distributed frameworks. As the lines blur between traditional risk, big data analytics, distributed data stores, and various back-end services for mobile applications and intelligent advisors, it is increasingly important that these distributed systems share both resources and data while delivering service-level guarantees.

As banks expose interfaces to internal systems from cloud-based services, software environments need to be robust and secure and current with the latest security updates and patches.

Capability	Competitive Grid Manager	Spectrum Computing
Performance (Typical latency)	~25 ms	~1 ms
Scale (cores per application)	8,000	128,000
Spark workload support	No	Yes
Python language support	No	Yes
Hadoop framework	No	Yes
Advanced batch/workflow management	No	Yes
Recursive workloads	No	Yes
Multi-Cluster support	No	Yes
Regular updates and patches	No	Yes
Multi-Cloud support, workload-driven cloud-bursting	No	Yes
Engineering / Implementation / Support capacity	Minimal	Hundreds

IBM Spectrum Symphony has a vibrant customer base, a clear product roadmap and with more than one hundred developers, consultants and support personnel.

- Clear technology roadmap
- Continuous engineering, software updates, and security enhancements
- Global implementation expertise

Learning more

To learn more about the IBM Spectrum Symphony and IBM Spectrum Computing software, contact your IBM representative or IBM Business Partner, or visit:

ibm.com/systems/spectrum-computing/products/symphony/