GridGain

An Overview of In-Memory Computing for High Performance Financial Applications

A GridGain In-Memory Computing eBook

July 2016

Table of Contents

Introduction

Imagine that a new technology comes on the scene with the power to revolutionize the financial services industry. Suddenly, commerce can happen at the speed of light. Real-time analytics let you deliver products and services to your clients almost instantaneously. You can achieve real-time financial regulatory compliance while keeping your clients and business safe. You can operate at a level of efficiency that leaves your old systems in the dust.

This breakthrough technology is in-memory computing and it is solving some of the toughest big data challenges in the financial services industry, such as:

- Algorithmic and High Frequency Trading Requirements
- Advanced Analytics (TCA, Risk Management, Derivatives Pricing, Sentiment Analysis, etc.)
- Regulatory Tightening (Dodd Frank, Volker, Best Execution, Basel, MiFID, CCAR, etc.)
- Real-Time Compliance (AML, KYC, Fraud Detection, Trading Compliance)

If you are new to in-memory computing, curious to learn how in-memory computing can be used for financial applications, or seeking to educate a non-technical team member about the benefits of in-memory computing for financial applications, this eBook can help. This beginner's guide to in-memory computing for financial applications will cover:

- How financial services and insurance companies are using in-memory computing
- How in-memory computing can help your financial applications achieve speed and scale
- The difference between an in-memory computing platform such as the GridGain In-Memory Data Fabric and other in-memory computing approaches
- How an in-memory computing platform works with your existing IT ecosystem and how to get started

After reading this eBook, you will know how an in-memory computing platform such as the GridGain In-Memory Data Fabric can address your company's most demanding current and future regulatory and client requirements.

In-Memory Computing for High Performance Financial Applications

In 2014, a company in New York specialized in watching hundreds of news channels around the globe and reporting events to companies in the financial industry. If an expert in Japan went on camera and said, "I just talked to my customers, and they hate the new iPhone update," this company forwarded the news to financial clients. They knew that in twenty seconds, Apple stock would fall. Soon, that specialized company will be replaced by a single computer sitting in the back room. Instead of one hundred human beings glued to a screen, trying to catalog information as quickly as possible, that process will be automated. And that's only the beginning of the transformation in-memory computing will have on the financial industry.

In a few years, mobile banking and payments will be the norm. People will make purchases in a matter of seconds using nothing but their phones. Fraud will be reduced to a fraction of its current presence in the market, keeping everyone safer. High-frequency trading and risk analytics will continue to grow, and trade will become

unbelievably accurate. An enormous amount of money will be saved. And all of this will be made possible by harnessing the speed of memory.

Let's explore how financial services and insurance companies are using the power of in-memory computing for their high performance application needs.

Time Is Money: Straight-Through Processing (STP)

In financial systems, time is literally money. This is true for everything from processing forms to reacting quickly to change. If you can make a stock trade before anyone else realizes there's an opportunity, you make money.

That's why this industry was a pioneer of in-memory computing. Since the mid-2000s, in-memory computing has had a tremendous impact on our financial systems. Algorithmic trading, folder management, and risk analysis have been revolutionized because of the speed and scale that in-memory computing offers.

Straight-through processing, or STP, is the most fundamental example of this. STP is a process that acts like escrow in the financial industry. When you trade something, you have a buyer, a seller, and a contract. The contract ensures that the seller has the goods and the buyer has the money. That contract has to be verified before the trade takes place, which is called closing the books.

In the early 2000s, everyone in the finance industry was closing books under the "T plus 3" method. In other words, they were closing books for each trading day three days after the fact. They had no choice: the financial system is very complex and needs to be analyzed and processed in relation to a number of other factors. It involves countless data points, and a large financial institution makes millions of trades per day. This requires enormous computational power. And at the turn of the millennium, "T plus 3" was all that traditional computing methods could handle.

In-memory computing changed all that by enabling a system that allows for same-day reconciliation of a company's books: STP. By "same day," we are not talking hours. We are talking minutes and seconds. STP allowed companies to convert billions of dollars of operational costs into profits. And that was only the beginning.

High-Frequency Trading

Another revolution that in-memory computing has effected in the financial industry is high-frequency trading. High-frequency trading is pretty much what its name implies: trading extremely fast. In the past, low-frequency trading was the manual way to do it. Human beings—typically trader analysts—had to look at stocks and folders and make each trade manually.

Today, high-frequency trading has taken the human element out of the equation. It is often called "automatic trading," because computers are able to trade directly with each other, according to their programmed algorithms. High-frequency trading allows financial institutions to make hundreds of thousands of trades per second.

Hedge funds reap some of the biggest benefits of high-frequency trading. We have a GridGain customer that embedded our in-memory computing solution into the system they were already using. They didn't purchase any new hardware. And the results were fascinating. They were able to do four times the business they were able to do before. That added up to a tremendous benefit.

Our client was impressed and so were we. They only updated one part of their system, and just from that, they saw four times as much success. That was just the beginning of what in-memory computing could do for them and for hedge funds in general.

Real-Time Backtesting

In the financial industry, real-time backtesting has also seen enormous benefits from in-memory computing.

Automatic trading is based on algorithms. A financial analyst devises trading strategies. The simplified version goes something like "If this particular group of stocks goes up in price this much, we're going to sell it. If it drops below this level, we're going to buy it." That's an algorithm. Once the analyst generates it, it's deployed into the trading system so the computers can execute it.

Naturally, you can't send a new algorithm into the trading system without testing it first. These calculations are dramatically complex. You have to test them before you deploy them. If you make a mistake, you can lose a lot of money. This is automatic trading. You set it up and it goes.

So, how do you test the algorithms? You take the past six or twelve months of market history and apply the algorithm to see how it does under those conditions. This is called backtesting.

Six or twelve months of market history is a lot of information. Remember, with high-frequency trading, organizations are able to make hundreds of thousands of transactions per second. Before in-memory computing, financial institutions were spending a dramatic amount of time on this. Every time they made a small tweak to their algorithms, they had to do the backtesting all over again.

Once in-memory computing came on the scene that changed. These organizations can now keep all six or twelve months of market history in memory. We've seen users increase their performance to be anywhere from five to ten times faster than before. They can run backtesting in real-time, which makes a big difference. The faster you test and deploy your new algorithms, the quicker you can react to the market—and the more money you make.

Risky Business: Real-Time Risk Analytics

Risk analysis, including settlement risk and trading risk, is another area where in-memory computing impacts the financial industry.

Real-time risk analytics is the ability to calculate risk on the variety of instruments and portfolios in a book of business in real time. Financial organizations constantly buy and sell a bunch of stocks on behalf of their customers. Every time something happens on the market, no matter what it is, the prices of those stocks change.

Corporate events, political events, and macroeconomics—when something changes, risk exposure changes, and the companies stand to lose a lot of money if they're not on top of it.

Whether it's a train crash in China, a horrible storm in Europe, or a major flood in South America, that event is going to have an effect on the market.

News feeds and social media have added huge complications to the risk factor model. So much information is pouring in at once that traditional methods aren't able to process it. In the past, risk analysis required hundreds and thousands of human analysts. But since human talent doesn't scale, the analytical reports were slow. Analysts could tell you tomorrow morning what happened yesterday, in terms of financial risk.

For a long time, this was a big problem. Risk analysis is probably the most important part of a financial services company, because risk underlies everything you do. And risk analytics has been a challenge for financial organizations for hundreds of years. How do you recalculate risk quickly and frequently? In the past, risks were calculated once a week. Then it became twice a week, and then maybe once a day. Today, thanks to in-memory computing, risk analysis happens in real time.

In-memory computing enables real-time risk analysis by taking humans out of the equation. A big piece of the puzzle is natural language processing which empowers systems to gather and understand enormous amounts of audio data. In-memory computing pulls together all the information streaming in through the Web in real time, analyzes it, processes it, and applies it to stock prices and forecasts.

GridGain helped the third-largest bank in Europe build a new real-time risk analytics system using in-memory computing. The numbers we achieved even surprised us. On $25,000 worth of hardware, the bank was able to achieve one billion transactions per second. Before in-memory computing, even companies that spent millions of dollars on hardware could only achieve between five and ten million transactions per second in risk analysis. That is an enormous difference. Not only is this European bank saving a lot of money on hardware, it is able to react to even small events in the market in real time. Before, like a lot of other companies, it had to ignore small events and just concentrate on the big ones. Now that it can factor in everything, its risk calculations are more accurate mathematically, which has increased its margins.

In the financial industry, there is no such thing as too much risk analysis. If financial organizations could recalculate the entire world minute by minute, they would, because that means more accuracy. More accuracy means better margins. In-memory computing brings us closer to that ideal every day.

On the Go: Mobile Payments

We are on the edge of a major revolution in the way we pay for things. That revolution is happening around a single device: smartphones. You only need to look at Apple and its competitors to see how important this is. Companies like Square and PayPal are ready to capitalize on it.

Take one of our clients as an example. This company created an application that allows you to buy things using nothing but your phone. It doesn't matter if it's a bicycle, a laptop, or a cup of coffee. You just present your phone to the merchant, and the charge shows up on your cell phone bill. This app allows you to make that purchase

through your phone service carrier. Your phone makes the transaction so you don't have to exchange money or anything else.

The problem was that when our client first talked to major carriers about this idea, they said, "The idea is great. But we have one requirement: you've got to be able to support millions of transactions in real time. Because we can give you those millions of users, but we don't want to have a negative user experience on our end. If you're going to touch our systems with your payment service, you have to guarantee that it's going to be able to handle this kind of scale."

Suddenly, our client, which was a small startup, was faced with an enormous challenge. They had to find a way to go from a few beta testers in the early stages to signing up with major carriers. They knew that millions of people could start using their payment system almost overnight. They needed a fast expansion that would support that many people. That was when they came to GridGain and implemented in-memory computing into their system. Right away, they had the support they needed to handle a huge amount of demand. They process millions of transactions in real time, so when you pay for something, you can immediately look at your bill online and see that charge.

This is the cutting edge of commerce in the modern world. In-memory computing makes it possible.

Payment Fraud Protection

Ten years ago, fraud protection was always about looking back. Say you had a consumer credit card. Your credit card company had a whole department whose job was just to look at the transactions you made a week ago and try to understand the buying patterns. If they saw a pattern that looked suspicious, they called you and said, "Somebody bought a pair of jeans in China on your credit card. Was it you?" If you said no, you went through the headache of filing a claim.

Today, in-memory computing has allowed us to move to real-time fraud protection. When you swipe your credit card, a computer is able to determine in real time whether the transaction you're making fits your buying pattern. If the purchase is suspicious, it can decline payment on the spot and stop that fraud from happening in the first place.

Here's an example. Let's say you live in San Diego, California. The computer sees you make a transaction at a cafeteria in San Diego for lunch. Then, thirty minutes later, it sees another transaction in China. That computer knows you can't travel from San Diego to China in thirty minutes, and it determines that since you live in San Diego, the cafeteria transaction is probably correct while the China transaction is probably a fraud. It flags the fraudulent purchase on the spot.

That's a big deal. Instead of being retrospective, the fraud protection system becomes reactive. And it couldn't work without in-memory computing. The complex algorithms and mathematics behind your buying history translate into a lot of data. You can't process that data traditionally, because it would worsen the user experience. You don't want to experience a one-minute delay every time you buy something. All of the analysis needs to happen within a single second. With in-memory computing, it does. There is no other way to do it.

One of our payment processing clients in Europe came to GridGain specifically for this reason. They wanted to be able to tell their merchants that, in addition to setting them up with regular payment systems, they could also provide fraud protection to the merchants' customers. That's an important feature because about 10 to 15 percent of transactions are fraudulent. The bigger your business, the more money you stand to lose through fraud.

Using GridGain in-memory computing solutions, our client was able to reduce the rate of fraud from 10 to 15 percent to 2 to 5 percent. Thanks to in-memory computing, it is a win-win situation for everyone but the thief.

Insurance: Sign Me Up

When you call an insurance company looking for insurance, you don't get a quote right away. You talk to the agent and tell them the information they need to know about insuring you. Or you fill out forms online. At that point, you're stuck. There's nothing else to do but wait. So you start checking out other insurance companies.

Whether you want health insurance, home insurance, auto insurance, life insurance, or something else, you have to wait to get your quote. In the meantime, you look at a bunch of other insurance companies and, in the process, you forget about the first call you made.

Insurance companies haven't been able to do it any other way. The process they go through to generate your quote requires intense analysis. They look at all the information you have given them and do a comprehensive risk analysis. The computer runs overnight or for a few days before they get their answer.

If insurance companies could get that answer instantly, they could close an estimated 30 to 40 percent more customers just by being able to deliver a quote while the lead is still on the phone. A large insurance company could gain an extra quarter-billion dollars per year in revenue. In-memory computing enables that instant quote., The process would look like alien magic to competitors who are still working with traditional computing. Suddenly, a company could grab a much larger percentage of market share, quickly.

Multinational insurance corporations already use in-memory computing to do this. It's not a pie-in-the-sky dream anymore. This is the real deal.

How In-Memory Computing Achieves Speed and Scale

Now that you learned what can be achieved with in-memory computing, let's take a step back and discuss how in-memory computing works.

In-memory computing is about two things: making computing faster and scaling it to the collective size of billions of global users, more commonly known as the Internet. In-memory computing leverages two key technologies: random-access memory (RAM) storage and parallelization.

Speed: RAM Storage

The first key is that in-memory computing takes your data off of a disk drive and moves it into RAM storage. Your hard drive is by far the slowest part of your computer. A typical hard drive is literally a spinning disk, like an old-fashioned turntable. It has a lot of moving parts, and it spins in a vacuum where the arm of the turntable physically reads it. Meanwhile, RAM is the second-fastest thing on your computer. Only the processor is faster.

With RAM, you have no moving parts. Memory is just a chip. In physical terms, an electrical signal reads the information stored in RAM. It works at the speed of electricity, which is the speed of light. When you move your data from a disk to RAM storage, your computer runs anywhere from five thousand to a million times faster than before.

The human mind has a hard time grasping that kind of speed. We are talking about nanoseconds, milliseconds, and microseconds. A good analogy is that traditional computing is like a banana slug crawling through your garden at 0.007 miles per hour while in-memory computing is like an F-18 fighter jet traveling at 1,190 miles per hour, or twice the speed of sound. In other words, disk drives are really, really slow. And when you pull all your data off of a disk and put it into RAM, computing becomes really, really fast.

You can look at it like a chef in a restaurant. The chef needs ingredients to cook his meals: that's your data. The ingredients might be in the chef's refrigerator or they might be ten miles down the road at the grocery store. The refrigerator is like RAM storage: The chef can instantly access the ingredients he needs. When he's done with the ingredients and the meal is finished, he puts the leftovers right back in the refrigerator, all at the same time. The grocery store is like disk storage. The chef has to drive to the store to get the ingredients he needs. Worse, he has to pick them up one at a time. If he needs cheese, garlic, and pasta, he has to make one trip to the grocery store for the cheese, bring it back, and use it. Then he has to go through the whole process again for the garlic and the pasta. If that isn't enough, he has to drive the leftover ingredients back to the grocery store again, one by one, right after he's done using each of them.

But that's not all. Even if you could make a disk drive that was as fast as RAM, the system that traditional computing uses to look for the information on a hard disk—processor to RAM to controller to disk—would still make it slower than in-memory computing.

To return to our example, let's say there are two chefs: one representing in-memory computing and the other traditional computing. The chef representing in-memory computing has his refrigerator right next to him and he also knows exactly where everything is on the shelves. Meanwhile, the chef representing traditional computing doesn't know where any of the ingredients are in the grocery store. He has to walk down all of the aisles until he finds the cheese. Then he has to walk down the same aisles again for the garlic, then the pasta, and so on. That's the difference in efficiency between RAM and disk storage.

RAM versus Flash

Flash storage was created to replace a disc drive. When it's used for that purpose, it is also called a solid-state device, or SSD. SSDs are made of silicon and are five to ten times faster than disk drives. However, both flash storage and disk drives are attached to the same controller in your computer. Even when you use flash, you still

have to go through the same process of reading and writing from a disc. The processor goes to RAM, RAM goes to the controller, and the controller retrieves the information from the disk.

Flash accesses the information faster than disk, but it still uses the same slow process. Moreover, because of the inherent limitation in flash's physical design, it has a finite number of reads and writes before it needs to be replaced. Modern RAM, on the other hand, has unlimited life and takes up less space than flash. Flash may be five to ten times faster than a standard disk drive but RAM is up to a million times faster than the disk. Combined with the other benefits, there's no comparison.

Scale: Parallelization

RAM handles the speed of in-memory computing. But the scalability of the technology comes from parallelization. Parallelization came about in the early 2000s to solve a different problem: the inadequacy of 32-bit processors. By 2012, most servers had switched to 64-bit processors which can handle a lot more data. But in 2003, 32-bit processors were common and they were very limited. They couldn't manage more than four gigabytes of RAM memory at a time. Even if you put more RAM on the computer, the 32-bit processor couldn't see it. But the demand for more RAM storage was growing anyway.

The solution was to put data into RAM across a lot of different computers. Once it was broken down like this, a processor could address it. The cluster of computers looked like it was one application running on one computer with lots of RAM. You split up the data and the tasks, you use the collective RAM for storage, and you use all the computers for processing. That was how you handled a heavy load in the 32-bit world and it was called parallelization.

When 64-bit processors were released, they could handle more or less an unlimited amount of RAM. Parallelization was no longer necessary for its original use. But in-memory computing saw a different way to take advantage of it: scalability.

Even though 64-bit processors could handle a lot more data, it was still impossible for a single computer to support a billion users. But when you distributed the processing load across many computers, that kind of support was possible. Better, if the number of users increased, all you had to do was add a few more computers to grow with them.

Picture a row of six computers. You could have thousands of computers but we'll use six for this example. These computers are connected through a network, so we call them a cluster. Now imagine you have an application that will draw a lot of traffic, too much traffic to store all of the data on one computer. With parallelization, you take your application and break its data into pieces. Then you put one piece of it in computer 1, another piece in computer 2, and so on until the data is distributed optimally across the cluster. Your single application runs on the whole cluster of computers. When the cluster gets a request for data, it knows where that data is and processes the information in RAM from there. The data doesn't move around the way it does in traditional computing.

Even better, you can replicate specific parts of your data on different computers in the same cluster. In our example, let's say the data on computer 6 is in high demand. You can add another computer to the cluster that

carries the same data. That way, not only can you handle things faster, but if computer 6 goes down, the extra one just takes over and carries on as usual.

If you tried to scale up like this with a single computer, it would get more and more expensive. At the end of the day, it would still slow you down. With parallelization, in-memory computing allows you to scale to demand linearly and without limits.

Let's return to the chef analogy, where a computer processor is a chef and memory storage is the chef's stove. A customer comes in and orders an appetizer. The chef cooks the appetizer on his one stove right away and the customer is happy.

Now what happens when twenty customers order appetizers? The one chef with his one stove can't handle it. That twentieth customer is going to wait three hours to get her appetizer. The solution is to bring in more chefs with more stoves, all of them trained to cook the appetizer the same way. The more customers you get, the more chefs and stoves you bring into the picture so that no one has to wait. And if one stove breaks, it's no big deal: plenty of other stoves in the kitchen can take its place.

The Internet has created a level of scale that would have been unheard of just fifteen or twenty years ago. Parallelization gives in-memory computing the power to scale to fit the world.

Types of In-Memory Computing

For the more than twenty-five years, in-memory technology has transformed itself multiple times. You can't read about it in a canonical way without getting confused. Is it database caching or a data grid? Is it streaming or acceleration? And what about computing platforms and data fabrics?

Let's discuss why we have so many different terminologies for types of in-memory computing, where they came from, and how you can unify them under an in-memory computing platform such as a data fabric. We'll take you through the evolution of this technology, starting at the beginning: with in-memory database caching.

In-Memory Database Caching

Fundamentally, in-memory computing was born in the late eighties and early nineties. And originally, it was developed as database caching.

What is database caching? Imagine a typical enterprise application such as one for the insurance industry. An insurance agent sits in front of his computer entering information. He presses a button, and a policy proposal for a US customer comes back to him. The agent's request goes from his browser (or his "terminal application," back in the day) to an application layer on his company's servers. The application layer is where the logic of the request is written, using a programming language. This application layer needs to get some data to work and the data is stored in a database located on another set of servers, potentially in a different company's data center. This is a "three-tier architecture" and even today, it's common of most enterprise applications. You have a browser that you interact with, an application where the logic is written, and a database where data is stored.

As explained earlier, traditionally every time you needed a piece of data, you had to fetch it from a database. Back in the seventies and eighties, that worked fine – the amount of data was so small that the system could handle it. But as we moved into the nineties, things changed. We started to realize, "We must constantly fetch data from the database over the network to the application layer, discard it almost immediately after we're done processing it, and repeat the whole process over and over again for each request. This is clearly becoming a real efficiency bottleneck."

So, people thought, "Why don't we just keep the most frequently accessed data in the application layer so we don't have to go and fetch it every time?" That's what caching is. It's keeping your most frequently used data in the application layer so it's much closer to you than it would be in your database. And you keep that data in memory, because that's where the application is running.

That's how the very first application of in-memory computing came about, more than twenty-five years ago. And even though we've grown to use other forms of caching, database caching is still one of the dominant use cases for in-memory computing. From a technology perspective, the practice of keeping frequently accessed data closer to you is used everywhere in computer science, not only in databases.

We even do this ourselves. We keep things closer when we know we need to access them more quickly and more often. In the chef analogy, database caching is like having your groceries with you in the kitchen, instead of driving all the way to the grocery store for each item.

The first cache systems were very primitive. You'd ask an engineer to set one up, and he'd say, "I can do it in a week." But then caches started to grow in complexity. People started asking, "What if I have multiple servers in my application layer? How do I keep the data fully in sync for transactions?" or "What if I keep my data in memory and my server crashes? Do I lose the data I was keeping in cache? How do I protect it?" And so on. By the early 2000s, the complexity of database caching had grown so much that we needed a new way to keep everything organized. The solution that emerged was called an in-memory data grid.

In-Memory Data Grid

The transition from database caching to data grids was a key step in the evolution of in-memory computing. Database caching alone worked as long as you were only using it for a few specific applications. But once you started using it in a broader way, people began asking for a lot of new features: transactions, reliability, distribution, splitting the load among servers in a cluster, and so on. Data grids came on the scene as an answer to these complex, technical questions.

The term "data grid" denotes a cluster of computers that can hold a large set of data across those multiple computers. Fundamentally, it works just like database caching. The difference is that it allows you to keep data in memory across multiple computers in a very sophisticated way, so it can scale much better and handle tremendous complexity.

Back to the chef example, imagine that one chef cooking for a few people is database caching. The chef can keep a few tomatoes, potatoes, and a gallon of milk in his fridge, and that's enough to take care of all the customers.

With a data grid, that chef is dealing with exponentially higher demand. Now he needs thirty bags of tomatoes, six crates of potatoes, and a hundred gallons of milk. One fridge of supplies doesn't cut it anymore. He needs to install a whole set of refrigerators—an entirely different infrastructure—to feed hundreds or thousands of people. The idea is the same but the complexity has multiplied many times over.

Stock trading data is an example of how we benefit from data grids in the real world. Nobody keeps stock trading data, called "ticks." You have ticks for every stock in the market and their prices change many times per second. Because it changes so quickly, stock-trading data isn't kept in a database. Instead, we keep the ticks in memory, because memory is fast enough to keep up with the constant changes. If the system crashes, it's not a big deal because we still have the data elsewhere. This is why financial institutions were among the first companies to get started with in-memory computing, and data grids specifically, about ten years after the invention of database caching.

In-memory data grids are still one of the most prominent in-memory computing products on the market today.

In-Memory Databases

In the late 2000s, another player came on the scene: in-memory databases. In-memory databases didn't bring much of a technological advantage to the table. They were more of a reactionary development in the industry. Even though data grids were incredible from a technological standpoint, they weren't easy to use. Consumers didn't know how to deal with them. They were comfortable and familiar with what they had used in the past: SQL-based databases.

So a few of the more traditionally minded folks in the industry thought, "Okay, we've been doing business with disks for the past twenty-five years, and we're outdated. This new in-memory technology has become so cheap and available that there has to be a way to use it. Why don't we make a faster version of databases by using memory as the primary medium of storage instead of disks?"

This is how all the startup companies in the in-memory database space came about. It's also why giants like Oracle, IBM, and Microsoft can claim that they have an in-memory option for their database products.

But in-memory databases aren't the cutting edge of this technology. They're faster than old disk-based databases, but they're slower than data grids. Their main appeal is that users are more comfortable working with a database system because it's familiar.

You can think of this in terms of our chef. He's facing an explosion of demand and he knows he needs to change the way he works in order to deal with it. One option is to set up a very complex infrastructure of fridges and keep them full with the necessary items. But he's not used to that. He's used to having one fridge and filling it up with new items from the grocery store (a database) when he needs to.

Then a grocery store chain comes to him and says, "You don't have to put in this complicated refrigerator system. We can sell you a local store, just like you're used to. We can install it right on your block. It won't be as fast as the refrigerator system so you'll still have to walk a little bit. But you won't have to get in your car and drive fifteen

minutes to the store and back every time you need an ingredient, so it will still be a lot better than what you had before. Most importantly, you already know how to use it. You've been doing it for years!"

And that approach works because the chef says, "That's a good idea. I don't want to deal with this new infrastructure of refrigerators. I've been using the grocery store my whole life. I don't want to learn anything new. If you guys can build a store on my block, then why not? True, the price gives me sticker shock, but I'll take it."

In-memory databases aren't as fast or as convenient as other types of in-memory technology. But their learning curve is almost zero. For some people, that makes all the difference.

In-Memory Streaming

So far, we've seen three main uses for in-memory computing: database caching, in-memory data grids, and in-memory databases. Along the way, these evolved into a variety of different flavors. One of those flavors is in-memory streaming.

In-memory streaming is also called in-memory complex event processing in some circles. The whole streaming process is a new way to use in-memory computing. Streaming is categorized by a huge, endless flood of information. It came to prominence in the early 2010s, with the explosive growth of the Internet of Things, wearable devices, sensory data, and machine-to-machine data generation.

For example, imagine you're using a database and someone purchases something from you online. You take that request and process it, no big deal. But it's another story when you have tens or hundreds of thousands of automatic devices (sensors) constantly sending data to you at the same time. You can't process them one by one. There's physically no way to do that. So you have to take a very different approach. You have to handle it in memory.

In-memory streaming is still a form of data processing. But instead of processing stationary data from a database, in-memory streaming processes data that's in motion. This is the type of data payload where you have an extremely large amount of data pouring into your system each second, endlessly. And that kind of demand takes in-memory technology to a whole new level.

We can connect this back to our chef. He's still cooking the same food but now his operations model is different. Even with a new infrastructure of refrigerators installed, he can't prepare his dishes one by one anymore. There's too much demand. He has to find a new, faster, and more agile way of making the food. So he starts preparing large batches of food in advance and he stores and organizes those dishes in the freezer. Then, when the requests pour in, he can fill the orders quickly because a lot of pre-packaged, pre-stored, and pre-sorted food is in stock.

Streaming works in a similar way. You're not dealing with each pickle and tomato individually anymore. That's not physically possible. Instead, you group things together in order to deal with the incoming flow. For example, let's say you have a sliding window of the last five minutes of events. You concentrate your processing power on that window of events, constantly aggregating, pre-processing, and pre-sorting its data. The in-memory aspect of that streaming processing is what makes it possible.

There is literally no end to in-memory streaming. More and more devices are added to the global network every day. Soon things like watches and cars will be connected to the Internet. Not only that, but devices will become more complicated as technology advances. That's why streaming will never end. The demand will only increase as civilization continues to evolve.

In-memory technology is very advanced but in-memory streaming is still fighting to keep up with demand. One of the largest trucking companies in the world spoke with us about installing dozens of sensors in each of their trucks globally. The sensors would send information about the wear and tear on each part of the engine in each of the trucks. If they can tell when a part is about to break, they can let that truck driver know before it breaks down and tell them how to get to the nearest service station. But they ran the calculations and found that the amount of data streaming in would easily consume all the processing power they have. They didn't have the processing power to sustain that kind of load and they were searching for hardware and software solutions to help them launch the project. That is just one example of how, even today, streaming can be a real challenge.

In-Memory Accelerators

One of the latest and most logical evolutions of these various types of in-memory products is in-memory accelerators. We explained how in-memory technology started with database caching, then matured into data grids, in-memory databases, and in-memory streaming.

In the early 2010s, a new trend started to appear. People began to say, "In-memory computing is pretty cool and obviously very important but it's so complex. Can there be some kind of 'plug and play' acceleration that makes it easier for us to use in-memory technology with our systems and products?" That line of thinking comes from a basic assumption: as in-memory computing becomes more mainstream, it needs to be simplified. We can no longer rely on top-notch engineers and PhDs to use this technology. It needs to be a hands-on experience for the average user. That's where in-memory accelerators come into the picture.

In-memory accelerators give people the "plug and play" capability they're looking for with in-memory technology. They combine the performance and scalability of in-memory computing with a dramatically simplified method of usage. You install something, and suddenly everything just gets faster. This is what the larger market has been awaiting.

Look at it through the eyes of our chef, or better, through the eyes of the company trying to sell him on a new infrastructure of refrigerators. The company that makes the refrigerator systems suddenly realizes, "We're losing customers to grocery stores just because the grocery store is promising chefs an easy transition. Our business model is all wrong. We can't ask chefs to learn and develop a completely new process on how they operate and cook their food. We need to give them an option where they can just sign up and push a button and suddenly they're cooking a lot faster."

This is what companies like GridGain are developing today. They're creating the technology that can inject the high performance of in-memory computing into a system without asking the user to significantly redevelop that system, all while providing the end user with an easy-to-use and familiar paradigm for processing data. The chef still gets his complex refrigerator infrastructure but it looks like a grocery store right in his kitchen, just like he used to have. And everybody's happy.

In-Memory Computing Platforms

If you are seriously looking into in-memory computing, you should look at the ultimate form of this technology: in-memory platforms.

Where do platforms fit into the picture?

Here's an example. Large companies have dozens and dozens of projects. Some of them use database caching. Some of them use data grids. Some of them use streaming, and some of them use accelerator products. Major data centers don't want to deal with each of those systems individually. They want a single platform that includes and supports them all. They want a strategic approach to in-memory computing, and that solution is an in-memory computing platform.

The GridGain In-Memory Data Fabric is the culmination of a quarter century's worth of in-memory development. It is a comprehensive in-memory computing platform that encompasses all of the other branches of in-memory computing. Products like the GridGain In-Memory Data Fabric support all of the different uses of in-memory computing that we discussed, from very old databases to data grids to streaming to "plug and play" accelerators. The GridGain In-Memory Data Fabric product is one of the key reasons that GridGain is the leading innovator in this industry.

Using an in-memory computing platform is natural and logical. By 2020, most applications will be supported by in-memory computing. With something that large revolutionizing the landscape, you need to take a strategic view. People don't want to buy individual piecemeal products. They want a comprehensive strategic platform for this technology.

To return to the chef, an in-memory computing platform is like the fridge vendor saying, "You can build this new system however you want. You want a grocery store outside your kitchen, you can have it. You want a grocery store *in* your kitchen, you can have that, too. And if you want some parts of the store in your kitchen and some parts outside of it, you can even have it both ways. You can build your new kitchen any way you like and you'll still get the same benefits of speed and convenience. You don't even have to deal with dozens of vendors to handle the different pieces. We can give you a simple tool to manage it all."

One real-life example of a company that benefits from an in-memory data fabric is a GridGain client that happens to be one of the largest banks in the world. This bank has one of the largest installments of in-memory computing ever built, to date. In 2014, they used in-memory technology for eighty different applications within the bank. They started with a data grid and then strategically decided to move away from that and towards an in-memory computing platform because it made sense for them as a global company. They didn't want to worry about coordinating ten different vendors. They wanted an in-memory computing platform because it included support for all the different uses of the technology they have today and will have in the future.

The old system of picking a specific vendor for a specific use case was prudent at the turn of the decade, around 2010. This major bank client only had five or ten applications to deal with then. But as the number of applications grew with enormous demand, they realized they couldn't play the vendor-by-vendor game anymore. An in-

memory computing platform was the comprehensive solution to their problem. It solved that problem for other companies of similar size as well.

Platforms are the latest advancement in the industry today. Any analyst will tell you so. They see their emergence as a comprehensive solution to the piecemeal approach in-memory computing has taken in the past.

In-memory computing platforms are the way things are going to work, moving forward. The user of the technology gets a single product with a single learning curve, a single configuration, and unified management and monitoring. This kind of streamlined technology solution allows users to harness the speed of memory in ways never before possible and the first companies to get there are outsmarting their competition in the process.

Impact on Your Existing IT Ecosystem

The big question companies have when they approach in-memory computing is: If we implement this technology, what is going to break? This is why banks don't install upgrades in their systems during the holiday season. For the last quarter of the year, nothing changes in their normal operating procedures. These are the biggest months of the annual sales cycle. They can't afford to update anything, because they know how much they stand to lose if those upgrades come with glitches. So they leave everything the same.

New systems and new technologies tend to have surprises lurking in the shadows. And if you're a CIO or technician maintaining the technology of a company, you want to know what new challenges you'll face once you sign up. Do you have to make a drastic overnight switch? How much time are you going to lose figuring it out and correcting mistakes? If this technology is new, does it pose a security risk?

The truth is this: In-memory computing doesn't come with hidden challenges. It scales with you, at your own pace. It's safe. And best of all, you get to skip the "adjustment period."

When CIOs and other business leaders ask GridGain if implementing in-memory computing is going to wreak havoc on their systems, we always give them the same answer. "No, your system will keep running exactly as it does now. Nothing is going to break."

The IT Ecosystem

An IT organization has to deal with a lot, including applications, databases, computers, virtualization, data centers, heating and cooling, and layers of software. This is the ecosystem we're dealing with when we talk about in-memory computing. It's complex and specialized. Any time you introduce something radical into this network of technology, it can have unintended consequences.

Why doesn't in-memory computing rock the boat the way other new technologies have done in the past? In-memory computing doesn't have to change the fundamental structure of what you're already working with. You don't have to worry about things like replacing databases. It uploads data from your database a lot more quickly, but the database itself doesn't move.

If you look at it through our chef analogy, the chefs aren't losing their ovens and refrigerators. They get to keep the same kitchen appliances they had before. We've just changed the arrangement of those appliances, and now they work a lot faster and are much less expensive to use. Compare this to a system that disrupts the fundamental way things work. Virtualization was like this. It was a nightmare for large companies to implement, because they had to start from scratch. They had been running their operating systems in a specific way for forty years, and when virtualization appeared in the mid-2000s, they had to change literally everything — hardware, software, networking, every single process they used. It was a necessary move, but it was extreme, and it took a decade to make the shift.

The change to virtualization for the IT ecosystem was like the invention of the car. Before cars showed up, the whole transportation system was about horses and carriages. Everything was structured around horses. You needed food, water, and shelter for the animals. You needed drivers who could work with them. The roads were designed for horses and manufacturers produced the parts to make carriages. Then, suddenly, you had cars, and you didn't need any of that anymore. Now you needed new things. The people who used to deal with horses were out of a job. You had to destroy the entire infrastructure and build paved roads that cars could drive on. Manufacturers had to stop producing wagon wheels and start building metal parts instead. You needed gas stations all over the place.

With in-memory computing, you bypass all of that, because you keep the same infrastructure you had before. If you want to eventually upgrade your freeways and gas stations, you can do that. But in the meantime, you get to use a blinding-fast car that works on the roads you already have. Nothing "breaks" when you add in-memory computing to the IT ecosystem. Everything in the ecosystem just gets faster.

Scale at Your Own Pace (Enterprise Application Integration)

Another question our clients ask is, "Do we have to make the shift from our current systems to in-memory computing all at once? How is this new technology going to fit in with the applications and databases we already have?" The merging of in-memory computing with your current systems is called "enterprise application integration." Not only is it easy and crisis free, it scales to exactly the level you need, when you need it.

When you bring in-memory computing into your business, you can keep using your existing databases and applications. You have to do some work to make the applications sync with the in-memory structure. How much work depends on how far you're looking to go and how fast. Again, you're not replacing anything. You're just syncing different factors. That sync can take anywhere from two hours to two months.

Here's a "two hours" example. At GridGain, we offer an in-memory accelerator for Apache Hadoop. Our accelerator is plug and play. You can download, install, and be up and running with this product in less than ten minutes. Your systems work up to ten times faster because you're uploading a lot of data processing into memory and you haven't changed any of the infrastructure in your system. You now have in-memory computing up and running and it took less than one business day.

That's all some companies need. Maybe you're a data analyst for a travel agency and put together vacation packages. You have to analyze a lot of data to do that—hotels, flights, hot spots, destinations, and everything else. This is all manual labor. You're sitting in front of your computer assessing and reassessing this information before it

ends up in a nice promotion spot on Expedia or Travelocity. With traditional computing, every time you launched a query, it took you five minutes to get an answer. With in-memory computing, you get that same answer in twenty seconds. And that really improves your performance as a data analyst.

Not all companies are that simple. The bigger and more complex you get, the longer it takes to integrate in-memory computing with your systems. For example, at GridGain in 2014, we worked with one of the largest banks in America to implement this technology. They wanted to rewrite their current system to get the most out of what in-memory computing had to offer them. They didn't have to do that but they wanted to because the way the system was very slow as it was written.

That's a situation where you're looking at a couple months of work instead of a couple of hours because you have to spend some time rewriting the system. Even though in-memory computing will work with what you have, you can help it reach its maximum potential when you change the way you access data. A lot of large companies put in the extra work for that reason. For them, even a little bit of time is money. Seconds saved can translate into millions of dollars of profit at the end of the year.

Regardless of how long it takes, however, in-memory computing fits nicely with everything you already have. You don't need to rip out the whole engine and replace it right away. You can keep your old databases as long as you want. At the same time, in-memory computing gives you the ability to build new classes and categories of applications that have the power to expand your reach to horizons you've never glimpsed before.

You don't have to overhaul your whole system. But if and when you do choose to make big changes, in-memory computing enables you to do radical new things that will revolutionize your current paradigm and position you for greater success.

A Secure Shift

Finally, most companies have invested a lot in security: perimeters that protect the incoming data to their network, technologies that track data through that network, anti-virus and anti-malware technologies, and all kinds of encryption. Those who are new to in-memory computing often ask, "Does in-memory computing pose a safety threat to our systems? Does it decrease our security liability?"

This is a hot topic. Fifteen years after major organizations began investing seriously in security technologies, CEOs were still losing their jobs over data breaches. Before you can move to a new paradigm, you need to know that it's safe. In-memory technology does not pose any new challenges to security. Since it is fundamentally just moving your data from your disk drive into RAM storage, all of the security technologies that you currently use to protect your data will continue to work exactly the same way they do now. You get an upside from a speed standpoint, with no security downside. In fact, your data is even slightly safer, because transient memory can't be stolen like data from a hard disk can.

In-memory computing does not fundamentally change the security paradigm for your business. The same security technologies that worked to safeguard your old systems will work with in-memory computing. The only difference is that everything becomes faster.

Unleash Innovation, Safely

The bottom line when it comes to bringing in-memory computing into your business is this: Everything is going to be fine. Nothing will break or create new challenges.

In-memory computing fits into what you're already doing and makes it better. But it also enables an entirely new paradigm that will save you money, make your company faster, and introduce incredible amounts of innovation to your business.

When you strategically harness the potential of an in-memory computing platform, you have the power to unleash tremendous possibility. The economics are unmatched. You will use less space and less heat than your competition. Everything about it will be superior to what you have now.

You can think outside the box, where the potential of parallelization becomes limitless. And you don't have to send your current system into upheaval in the process. That's just another natural advantage of in-memory computing: it builds a gateway into your future, hassle free.

GridGain In-Memory Data Fabric as Your In-Memory Computing Platform

For financial services, the question is how best to leverage in-memory computing technologies. Some of the largest financial institutions in the world are using the **GridGain In-Memory Data Fabric**, built on Apache Ignite. GridGain is an in-memory computing platform they use to achieve the speed and scale needed for their high performance applications. It can help your business do the same.

The solution delivers unprecedented speed and unlimited scale to accelerate your business and time to insights by enabling high-performance transactions, real-time streaming and fast analytics in a single, comprehensive data access and processing layer that spans all key applications (Java, .NET, C++) and data stores (SQL, NoSQL, Hadoop). It is a comprehensive in-memory solution that includes a database-agnostic data grid, a clustering and compute grid, a real-time streaming engine as well as plug-and-play Hadoop and Spark acceleration.

The GridGain In-Memory Data Fabric is available in an Enterprise and a Professional Edition. Both the Enterprise and Professional Editions are subscription-based products that are included with **GridGain support**. Professional support is an important service for most clients for both getting started with in-memory computing, making sure they have the most current releases and bug fixes, and reducing the impact of any potential problems should they arise. Learn more about the editions **here**.

To get started using GridGain, you can **download a fully functional 30-day trial version** of the GridGain Enterprise or Professional Edition.

Contact GridGain

To learn more about how GridGain In-Memory Data Fabric can help your business, please email our sales team at sales@gridgain.com or call us at +1 (650) 241-2281 (US) or +44 (0) 7775 835 770 (Europe).

About the Authors

Nikita Ivanov
Founder & CTO

Nikita Ivanov is founder and CTO of GridGain Systems, started in 2007 and funded by RTP Ventures and Almaz Capital. Nikita has led GridGain to develop advanced and distributed in-memory data processing technologies — the top Java in-memory data fabric starting every 10 seconds around the world today.

Nikita has over 20 years of experience in software application development, building HPC and middleware platforms, contributing to the efforts of other startups and notable companies including Adaptec, Visa and BEA Systems. Nikita was one of the pioneers in using Java technology for server side middleware development while working for one of Europe's largest system integrators in 1996.

He is an active member of Java middleware community, contributor to the Java specification, and holds a Master's degree in Electro Mechanics from Baltic State Technical University, Saint Petersburg, Russia.

Dmitriy Setrakyan
Founder & CPO

As a founder and Chief Product Officer at GridGain, Dmitriy Setrakyan is responsible for leading product development, professional services, and customer support operations. Dmitriy has been designing, architecting and developing software and applications for over 15 years and has expertise in the development of distributed computing systems, middleware platforms, financial trading systems, CRM applications and similar systems.

Prior to GridGain, Dmitriy worked at eBay where he was responsible for the architecture of performance sensitive high-traffic components of an add-serving system processing several billion hits a day. Before that Dmitriy served as a Lead Architect at Fitech Labs, focusing on high-performance software for trading systems, where he jump-started a new distributed caching and grid computing product line scaling out to 100s computers.

Dmitriy holds a Bachelor of Science in Computer Science from University of California at Davis specializing in Networking and Algorithms.

ABOUT GRIDGAIN

GridGain is revolutionizing real-time data access and processing by offering the enterprise-grade GridGain In-Memory Data Fabric built on Apache Ignite™. The solution is used by global enterprises in financial, tech, retail, healthcare and other major sectors. GridGain solutions connect traditional and emerging data stores (SQL, NoSQL, and Hadoop) with cloud-scale applications and enable massive data throughput and ultra-low latencies across a scalable cluster of commodity servers. A converged data platform, the GridGain In-Memory Data Fabric offers the most comprehensive, enterprise-grade in-memory computing solution for high-volume transactions, real-time analytics and hybrid data processing. The company is funded by Almaz Capital, MoneyTime Ventures and RTP Ventures. For more information, visit www.gridgain.com.