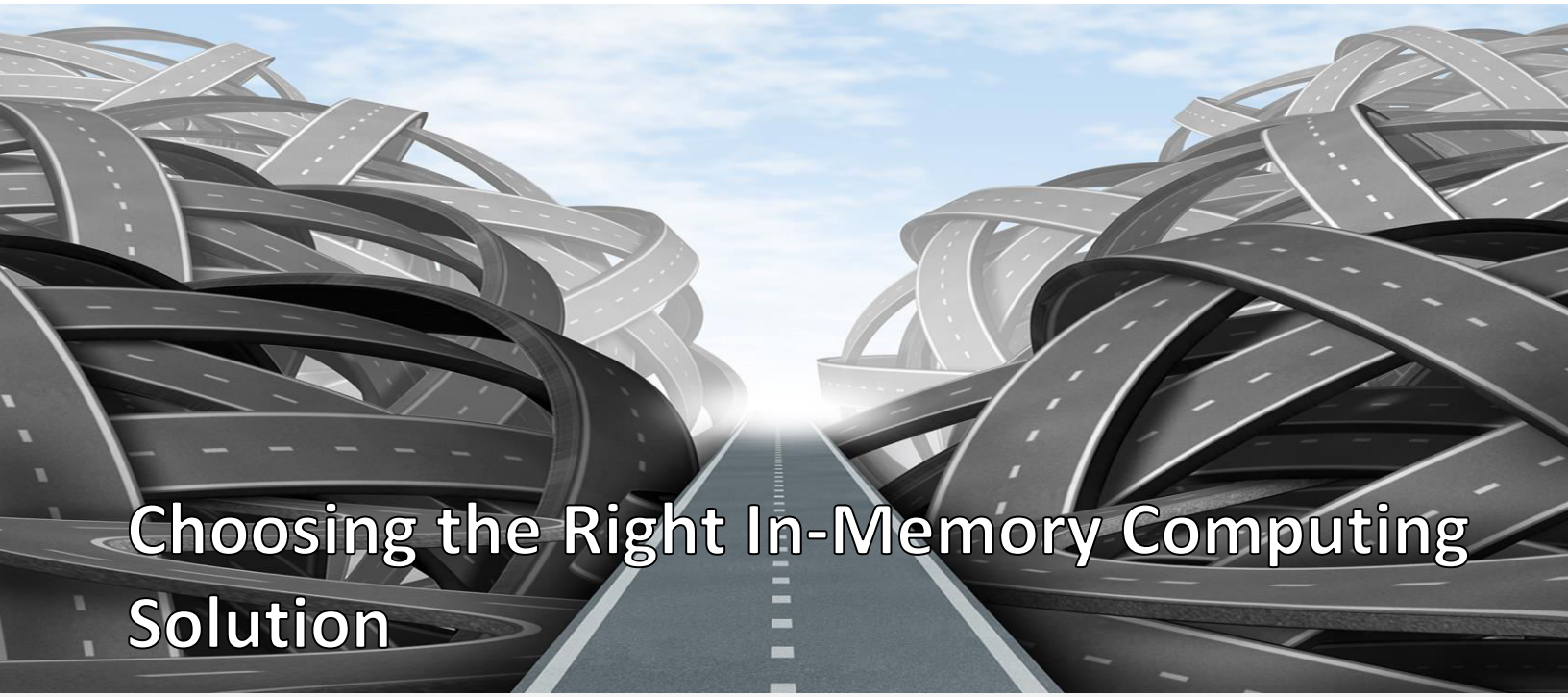
WHITE PAPER

GridGain



Choosing the Right In-Memory Computing Solution

A GridGain Systems In-Memory Computing White Paper

September, 2017

Contents

As in-memory computing (IMC) gains momentum across a wide range of applications, from fintech and ecommerce to telecommunications and IoT, many companies are looking to IMC solutions to help them process and analyze large amounts of data in real time. However, the variety of IMC product categories and solutions can be confusing, making it difficult to determine which solution is best for a particular application.

This white paper reviews why in-memory computing makes sense for today's fast-data and big-data applications. It dispels common myths about IMC and clarifies the distinctions among IMC product categories to make the process of choosing the right IMC solution for a specific use case much easier.

Why IMC Is Right for Today's Fast-Data and Big-Data Applications

Many data problems can be split into two groups: fast data and big data. Fast data generally applies to online transaction processing (OLTP) use cases that are characterized by high throughput and low latency. It's important for OLTP solutions to be able to process hundreds of thousands of operations per second with latencies in milliseconds or nanoseconds. This represents operational data sets that must be consistent with reliable performance.

There's also the big data side of the world, which is characterized by online analytical processing (OLAP) applications. In OLAP scenarios, high throughput isn't as important because they are designed for analyzing data and frequently process data offline in a batch-oriented fashion. Low latency is still critical in OLAP scenarios so queries will run fast.

Before focusing on IMC product categories, let's review what IMC is and why now is the right time for memory-first data solutions.

What IMC is. In-memory computing involves storing data in RAM instead of on disk, optimally across a cluster of computers that can process the data in parallel (though some IMC product categories are more limited). This architecture can provide a way to compute and transact on data at speeds that are orders of magnitudes faster than disk-based systems, providing response times in the range of milliseconds to nanoseconds. IMC can also provide other important benefits such as scalability and data consistency, depending on the product and category.

How technology evolved towards today's IMC technology. In-memory technology has been around in some capacity for decades although its early forms were expensive and had limitations. Not until the 2010s did two factors combine to make in-memory computing the technology of choice. Those two factors were RAM cost (now reduced to less than a penny per megabyte, compared to $35 per megabyte in the mid-1990s) and the advent of 64-bit processors.

The 32-bit processors of the past had a significant limitation: they could address only four gigabytes of RAM. The 64-bit CPUs could address much larger spaces: multiple terabytes. This ability to put much more data in RAM gave a big push to in-memory computing.

Why IMC performs better than disk-based architectures. The reason that memory-first architectures perform faster than disk-first architectures has to do with the way the data is accessed. Disk-first architectures use disk as primary storage and use memory only for caching. Memory-first architectures use memory as primary storage and disk for backup. This is important because there are several reasons why disk access is slower than memory access.

With disk-first architectures – such as RDBMS and NoSQL databases and file systems such as Hadoop's HDFS – there is a multi-step process that must occur to access a byte of data. The request starts with an API call, then it goes to the operating system and through the I/O controller before finally accessing the disk to load the data. At this point, even if only one byte of data is needed, more must be loaded, because disk is block-addressable and not byte-addressable storage. This is significant because an entire block (whichever block size has been configured in the OS, typical configurations for a data store volume could be four kilobytes or eight kilobytes) must be loaded. Each step introduces latency, from several milliseconds to hundreds of milliseconds.

With a memory-first architecture, the process is much simpler and faster. Accessing a byte of data requires only the referencing of a pointer (which specifies the location of that byte in memory). No loading is needed. The system can read kilobytes of data immediately without loading or processing it. There is direct access to any portion of the memory and that access is measured in nanoseconds, not in milliseconds like the disk access described above.

For applications requiring truly fast performance – nanoseconds or microseconds instead of milliseconds – memory-first architectures are the clear choice.

Dispelling Myths About IMC

While performance is a key factor in deciding on a data architecture, there are other factors as well. Sometimes myths about in-memory computing come into play when those factors are considered. Let's look at four common myths about in-memory computing and why they are not based in fact.

Myth #1: Too expensive. In the mid-1990s, loading a terabyte of data in-memory cost millions of dollars. However, the cost of RAM has been decreasing about 30% per year. Memory is now very affordable and it is being used as a main system of record in many use cases.

Myth #2: Not durable. While it is true that DRAM will not survive a server crash, most of today's in-memory computing systems are designed to solve that problem through fault-tolerance strategies such as automatic data replication and integration with persistent data stores. For example, the Apache® Ignite™ in-memory computing platform (and its commercial version, GridGain), will automatically persist and load data from any RDBMS, NoSQL, or Hadoop disk-based database.

Also, there are some promising developments on the horizon in the area of non-volatile memory. For example, Intel's 3D XPoint is a byte-addressable, flash-based memory that will plug into PCI slots and act

almost like current DRAM. Because of its non-volatile nature, this type of memory will be able to survive several restarts. While it will not be as fast as DRAM, it will be faster than today's flash.

Myth #3: Flash is fast enough. Today's flash-based memory is faster than spinning disks but it is still generally a block-level device and must go through OS I/O, the I/O controller, marshaling, and buffering. For applications that need to access individual bytes of data, DRAM is about a hundred times – or, in some cases, a thousand times – faster than flash-based access. If a ten-millisecond latency is good enough for a particular application, then flash-based memory is fine. However, if the application requires latency of one millisecond, there is a benefit to paying a little more and doing most of the processing through DRAM.

Myth #4. Memory is only for caching. While caching is an important aspect of memory, many systems use caching only as a small component of their architectures. Today, many types of processing happen directly in memory: machine learning, streaming, queries, transactions, and so on.

As awareness grows that these common misperceptions are not based in fact, interest in in-memory computing products is growing substantially.

In-Memory Computing Product Categories

There are many terminologies for types of in-memory computing solutions. It's important to understand where the solutions came from, their capabilities, and how they can be unified into a complete in-memory computing platform.

In-memory computing products generally fit into the following four categories:

- In-memory options for disk-based databases
- In-memory databases
- In-memory data grids
- In-memory computing platforms

The first of these categories, in-memory options for disk-based databases, is very limited in scope, and not scalable to fit with the distributed architectures of the future. The remaining three categories involve distributed-memory architectures, so they are better suited to the fast-data OLTP applications that are becoming more prevalent today. These categories also include some products that can capably address big-data OLAP issues by integrating with products such as Apache Hadoop®, Apache Spark®, and others.

The table "IMC Product Categories" on page 8 provides an overview of the advantages and disadvantages of the four product categories, which are described in more detail below.

In-memory options for disk-based databases. This category describes configuration options for disk-based databases that improve their performance by increasing the amount of data maintained in memory. Such options are available with Oracle® Database 12c, Microsoft SQL Server®, and many other RDBMS products.

The main advantage of using one of these products is that it works with a company's existing database, with no need to migrate data, change APIs, or change any code. Generally, all that is needed is a configuration change and the addition of more memory. Database performance will improve immediately.

However, this is a very limited solution, because it involves using the memory on only one server. There is no scalability, unlike other in-memory solutions that scale out in a distributed fashion and allow access to the total memory available on all of the servers in a cluster.

In-memory databases. Products in this category, such as MemSQL® and VoltDB®, behave like databases that keep all of their data in memory (distributed across multiple servers) for fast access. While they have some persistence capabilities, they are basically memory-first architectures. They are called databases because the way to interact with them is to use SQL with standard database drivers. Users can create schemas, issue queries, and use DDL or DML statements such as inserts, updates, deletes, and so on.

These systems have very high throughput because the data is accessed in memory. They also exhibit good scalability because they distribute their data across multiple servers. However, they also have some significant disadvantages.

With an in-memory database, SQL is the only way to talk to the system, and SQL does not fit into all distributed use cases. Sometimes it is important to be able to perform computations directly on the data which resides on a specific server in order to optimize performance.

Another major disadvantage is that a company's existing database must be ripped and replaced to install the new in-memory database and the in-memory database becomes the system of record. Most companies have a lot of infrastructure code around their existing databases so migrating to a new database requires a massive effort. For this reason, in-memory databases are most feasible for companies that are either selecting a first database or need to completely replace their existing database. Companies that are seeking to accelerate their existing databases are generally better off with products such as in-memory data grids or in-memory computing platforms.

It's worth noting that Apache® Ignite™ and GridGain, full-featured in-memory computing platforms, fully support ANSI SQL-99 including DML and DDL.

In-memory data grids. This group of solutions includes products such as Hazelcast, GigaSpaces, Oracle Coherence, Apache Geode / Pivotal GemFire, and Infinispan. In some cases, like that of Apache Ignite or GridGain, this category is a subset of the larger category of in-memory computing platforms (IMCP). IMCPs include in-memory data grids among their other in-memory components.

Products in this category distribute data across multiple servers. However, the methods of accessing the distributed data are different: data grids typically provide data access through a key-value API and sometimes a custom query API, with limited or nonexistent SQL support. In addition, data grids generally provide the ability to collocate custom computations for specific data.

The ability to collocate computations with data provides significant performance advantages. A typical application might involve accessing many accounts and closing their balances or performing other computations on them. By performing such actions directly on the server where the data resides, data grids can provide very high throughput and very low latencies. In a distributed system, the more the system can collocate computations with their associated data, the faster and more scalable the system will be because this eliminates network bottlenecks.

The downside of this architecture is that it is necessary to develop code to integrate the logic with the data grid. However, the benefit is extremely fast performance, with hundreds of thousands of operations per second throughput possible.

Another major benefit that data grids offer is compatibility with existing databases. Unlike in-memory databases, which must become the system of record, in-memory data grids can be inserted on top of an existing database and integrate with it. For example, if a company already has a database installed – such as Oracle, Microsoft SQL Server, or Apache Cassandra – the data grid provides an API that fits between the database and the application, providing caching and data distribution.

Because not all data grid products are created equal, it's important to look for the following features:

- **JCache implementation**. Java garbage-collection pauses can add significant time delays in situations where hundreds of gigabytes of data are stored across clusters containing tens of nodes. Data grids that implement the JCache standard, JSR 107, can avoid the garbage-collection process by storing and managing data outside of Java in off-heap memory. For better performance, choose a data grid that implements the JCache JSR 107 standard, such as the data grid component of the in-memory computing platforms Apache Ignite and GridGain, Oracle Coherence, or the enterprise version of Hazelcast.

- **Transactional consistency**. Managing transactions across a distributed grid is more challenging than managing them in a non-distributed architecture. In choosing a data grid, consider what level of control it offers over transactional consistency. This is particularly important for financial applications. Eventual consistency is not as strong a guarantee as fully ACID-compliant transactions. Look for a data grid with strong transactional semantics, such as the data grid component of the in-memory computing platforms Apache Ignite and GridGain.

- **Full SQL support with in-memory indexing**. Some data grids offer much more robust query support than just key-value access to the data. Some products provide limited SQL or custom APIs for querying data. However, the in-memory data grid of Apache Ignite and GridGain supports full ANSI SQL-99, including DDL and DML. This support lets users leverage their existing SQL code through JDBC and ODBC APIs. Also, Ignite and GridGain stores indexes in-memory, which is important because indexes substantially improve the performance of queries and joins.

Another important point to remember in choosing a data grid is that some products include other important in-memory components in addition to the data grid, placing them in the more encompassing category of in-memory computing platforms.

In-memory computing platforms. Anyone seriously evaluating in-memory computing solutions should focus their attention on in-memory computing platforms. With in-memory technology in particular, the platform approach holds greater value than the point solution approach. Companies don't want to install and manage myriad IMC point solutions. They want a single platform that supports a wide variety of in-memory computing use cases. They want a strategic approach to IMC and that solution is an in-memory computing platform.

The in-memory computing platform category includes products such as Apache Ignite, GridGain (the enterprise version of Apache Ignite), and Terracotta®. While data grids are essentially distributed caches with some basic transactional and compute capabilities, featuring key-value data access, in-memory computing platforms are more comprehensive products with more extensive capabilities.

In the case of Apache Ignite and GridGain, the extra features which put these products in the category of an in-memory computing platform include:

- **Multiple, well-integrated in-memory components**: Apache Ignite and GridGain include not only a database-agnostic in-memory data grid, but also an in-memory transactional SQL database with distributed SQL capabilities, and a real-time streaming analytics engine, all well integrated with each other. For example, computations can be sent to the nodes where the data resides, and a service can be attached to the data as well. With streaming, the data is streamed to the node where it will be eventually stored and processed. All of these components work with each other automatically, without any effort from the user. The GridGain Enterprise Edition and Ultimate Edition integrate additional features with these basic components, including enhanced security, management and monitoring tools, data-center replication, network segmentation resolution, and more.

- **Multiple ways to talk to data**: Data access is another area that distinguishes in-memory computing platforms from in-memory databases, which support data access only through SQL, and in-memory data grids, which provide data access primarily through a key-value API. In-memory computing platforms provide multiple ways to access the data. For example, Apache Ignite and GridGain provide both ANSI SQL-99 and key-value access, plus a wide range of other access mechanisms, via a Unified API, including C++, .NET, Java, PHP, and ODBC and JDBC drivers (which allow integration with data visualization tools such as Tableau). In addition, the Ignite file system integrates with the Hadoop file system (HDFS) for persistence to memory instead of disk. There is also a pluggable, in-memory MapReduce component that integrates with Spark for additional big data cluster-computing power.

Thanks to features such as these, in-memory computing platforms provide performance, scalability, availability and flexibility not available with a product that is only a data grid. For example, with Apache Ignite and GridGain, data might be stored using the key-value API and accessed using the SQL API, or stored in Java and accessed in .NET – all with the streamlined performance of multiple integrated in-memory components.

Table of IMC Product Categories

Category	Description	Advantages	Disadvantages	Best Suited For
In-memory option for disk-based databases Examples: Oracle Database 12c, Microsoft SQL server, and others	Configuration option for an existing disk-based database that boosts performance by caching data in memory	Works with existing database	Not scalable; Limited to memory on one server; Not a distributed architecture	Companies that want to improve performance of an existing database and don't anticipate scaling to work with large volumes of data at real-time speeds
In-memory database Examples: MemSQL, VoltDB, SAP HANA	RDBMS that stores data in memory instead of on disk	High throughput; High scalability; Disk persistence (disk is a copy of memory); Full SQL support; Applications can remain unchanged	Requires complete replacement of current database; Can interact only using SQL; no distributed parallel processing	Companies in the process of selecting a first database that require speed for fast-data (OLTP) situations
In-memory data grid Examples: Hazelcast, GigaSpaces, Oracle Coherence, Apache Geode / Pivotal GemFire, Infinispan	Distributed in-memory key-value store	High throughput; Low latency; Very high scalability; No need to replace existing databases	Requires code to integrate logic with data grid; Key-value stores may not include complete SQL support	Companies requiring very high speed and scalability for fast-data (OLTP) applications and the flexibility to work with either new or existing databases
In-memory computing platform Examples: Apache Ignite / GridGain In-Memory Computing Platforms, Terracotta	Includes an in-memory data grid, in-memory database, and streaming analytics	Same as in-memory data grid, plus: Multiple ways to access data; Integrated additional components	Requires code to integrate logic with data as data is stored in the distributed in-memory computing platform	Companies that require a full-featured platform with support for OLTP plus SQL-based OLAP and HTAP applications

Sberbank Case Study: In-Memory Computing in Action

One of the most noteworthy GridGain Systems financial services customers is Sberbank, the largest bank in Russia with over 130 million customers and more than 16,000 branch offices. Sberbank was faced with a similar problem to the one currently facing companies engaged in building a modern in-memory computing architecture. The bank was switching from a more traditional, brick-and-mortar setup – one in which people came into their offices and processed a limited number of financial transactions each day during a limited time period – to a new world with online and mobile customers transacting 24/7. The number of transactions increased from about 30 per second to 3,000 to 4,000.

The company forecasted future throughput requirements and determined that it needed to move to a next generation data processing platform to handle the expected transaction volume. Sberbank analyzed more than ten potential solutions in the in-memory computing space and found that the GridGain in-memory computing platform was the most comprehensive and performant solution. The bank concluded that GridGain would provide their next generation platform with a significant improvement in performance and scalability.

The GridGain in-memory computing platform provides several other important capabilities that Sberbank's next-generation platform will require such as ease of deployment, hardware independence of cluster components, and a rigorous level of transactional consistency, and machine learning and analytics. Of particular importance was the ability to conduct integrity checking and rollback on financial transactions. Sberbank could not find that level of consistency with other in-memory computing solutions.

In a January 2016 article in RBC, Herman Gref, the CEO of Sberbank, said that the bank selected the GridGain Systems technology to build "a platform that will enable the bank to introduce new products within hours, not weeks." He went on to state that the GridGain in-memory computing platform enables Sberbank to provide "unlimited performance and very high reliability" while being "much cheaper" than the technology used previously. Sberbank is using GridGain's in-memory computing platform to implement capabilities that could not be provided by the other vendors evaluated – a group that included Oracle, IBM and others.

GridGain Systems: A Leader in In-Memory Computing

With companies grappling with the challenges resulting from increasing fast data and big data workloads, demand for the GridGain in-memory computing platform is growing dramatically. This comprehensive platform contains a complete feature set that surpasses the capabilities of in-memory database or in-memory data grid point solutions, and it is well suited for OLAP, OLTP and HTAP use cases.

GridGain allows users to consolidate on a single high performance and highly scalable solution for transactions and analytics, resulting in a lower TCO versus running separate transactional and analytical architectures. Advanced SQL functionality and API-based support for common programming languages

enable rapid deployment. This, coupled with the rapidly decreasing cost of memory, boosts ROI for in-memory computing initiatives, enabling businesses to build less expensive systems that perform thousands of times better.

Users enjoy the following:

- **A unified high-performance architecture.** The GridGain in-memory computing platform consists of multiple features connected by a clustered, in-memory file system. The In-memory Data Grid, Compute Grid, Distributed SQL, Streaming Grid and Service Grid are interconnected. Computations occur as close as possible to the data. Additional features such as high throughput, low latency, load balancing, caching, in-memory indexing, streaming, Hadoop acceleration and other performance improvements are crucial to success in real-time modeling, processing, and analytics.
- **Scalability**. The GridGain platform is massively scalable, allowing companies to add nodes and memory to the cluster in real-time with automatic data rebalancing. Because it is hardware agnostic, users can choose their own preferred hardware for scaling out or up.
- **Full SQL support**. GridGain is ANSI SQL-99 compliant and supports DML and DDL so users can leverage their existing SQL code using the GridGain JDBC and ODBC APIs. Users with existing code bases that are not based on SQL can leverage their existing code through supported APIs for Java, .NET, C++, and more.
- **A Distributed ACID and ANSI-99 SQL-Compliant Disk Store.** The optional GridGain Persistent Store is a distributed ACID and ANSI-99 SQL-compliant disk store which addresses the issues of memory overflow and fast restart for the GridGain in-memory data grid. The Persistent Store may be deployed on spinning disks, solid state drives (SSDs), Flash, 3D XPoint or other similar storage technologies. If used, the optional Persistent Store keeps the superset of data and all the SQL indexes on disk, which allows GridGain to be fully operational from disk. The combination of this new feature and the platform's advanced SQL capabilities allows GridGain to serve as a distributed transactional SQL database, spanning both memory and disk, while continuing to support all of the traditional in-memory only use cases. Persistent Store allows organizations to maximize their return on investment by defining their own optimal tradeoff between infrastructure costs and application performance by adjusting the amount of data that is kept in-memory.
- **High availability**. GridGain provides essential high-availability features such as data-center replication, automatic failover, fault tolerance, and quick recovery on an enterprise scale.
- **Transaction processing**. The GridGain platform supports ACID-compliant transactions in a number of user-configurable modes.
- **Security features**. GridGain supports authentication, authorization, multiple encryption levels, tracing, and auditing.
- **Open Source framework.** GridGain is based on Apache Ignite, a popular Apache Software Foundation open source project. GridGain Systems created the code which was contributed to the Apache Software Foundation that became Apache Ignite and fully supports the technology behind Apache Ignite. The GridGain Enterprise Edition extends the features in Apache Ignite to provide enterprise-level capabilities and services, such as enhanced security, data center replication, management and monitoring tools, rolling production upgrades, and network segmentation protection. The GridGain Ultimate Edition includes all the features of the GridGain Enterprise Edition plus a Cluster Snapshots feature for automated for the GridGain Persistent Store feature.
- **Production Support**. GridGain Support, available to GridGain Professional, Enterprise, and Ultimate Edition users, includes faster availability of software patches and 24/7 enterprise-gradel support.

Making the Choice

Companies that need to process massive amounts of data in real time can benefit from a robust and affordable range of in-memory computing solutions. From the simple and limited performance boost of in-memory options for disk-based databases to the scalable and high-performing distributed architectures of in-memory databases (which replace existing databases), in-memory data grids (which integrate with existing databases), and in-memory computing platforms, there are applicable solutions for every use case. Understanding the distinctions among these solution categories is the key to making the best choice for any big data or fast data application requiring in-memory OLTP, OLAP or HTAP strategies.

Contact GridGain Systems

To learn more about how the GridGain in-memory computing platform can help your business, please email our sales team at sales@gridgain.com, call us at +1 (650) 241-2281 (US) or +44 (0) 7775 835 770 (Europe), or complete our contact form and we will contact you.

About GridGain Systems

GridGain Systems is revolutionizing real-time data access and processing by offering an in-memory computing platform built on Apache® Ignite™. GridGain solutions are used by global enterprises in financial, software, e-commerce, retail, online business services, healthcare, telecom and other major sectors, with a client list that includes Barclays, ING, Sberbank, Misys, IHS Markit, Workday, and Huawei. GridGain delivers unprecedented speed and massive scalability to both legacy and greenfield applications. Deployed on a distributed cluster of commodity servers, GridGain software can reside between the application and data layers (RDBMS, NoSQL and Apache® Hadoop®), requiring no rip-and-replace of the existing databases, or it can be deployed as an in-memory transactional SQL database. GridGain is the most comprehensive in-memory computing platform for high-volume ACID transactions, real-time analytics, web-scale applications and HTAP. For more information, visit gridgain.com.