MYSQL HIGH-AVAILABILITY ALTERNATIVES WHITE PAPER

The Short Guide to MySQL® High-Availability Options

An analysis of today's MySQL high-availability challenges and what asynchronous, semi-synchronous and fully synchronous replication offer toward reducing downtime, achieving full data consistency, automating failover, and simplifying administration.

SPONSORED BY Schooner Information Technology.

Schooner Information Technology
501 Macara Ave., Suite 101
Sunnyvale, CA 94085, USA
Tel: 408-773-7500
Fax: 408-736-4212
info@schoonerinfotech.com
www.schoonerinfotech.com
Updated 2012-03-01

Executive Summary

The data driving mission-critical business applications is growing explosively. It's getting harder for IT managers to meet the accelerating demands for service availability, data integrity, and performance scalability they need from their database deployments. This white paper discusses MySQL high-availability alternatives, analyzing the range of asynchronous, semi-synchronous, and fully synchronous replication solutions. The alternatives are compared in terms of downtime, data integrity, performance, scalability, and administrative ease.

MySQL High Availability Alternatives

High service availability is critical for virtually all MySQL deployments. To achieve high service availability, MySQL deployment architectures use replication among servers (normally in a Master – Slave setup). This is done to ensure service continuity during single points of failure and disaster recovery, and during routine administrative activities such as hardware and software upgrades, back-ups, schema changes, etc.

There are two fundamental approaches to MySQL replication and failover, implemented in several products:

- MySQL-Specific Replication, including loosely-coupled MySQL 5.1 asynchronous and 5.5 semi-synchronous replication, tightly-coupled synchronous replication in SchoonerSQL™, Oracle MySQL Cluster; and
- MySQL External Replication, including Oracle Golden Gate, Continuent Tungsten, and Linux DRBD.

MySQL-Specific Replication

Loosely-Coupled MySQL Asynchronous and Semi-Synchronous Replication

As shown in Figure 1, MySQL asynchronous and semi-synchronous replication are based on loose coupling between the Master and Slaves. As update transactions execute on the Master their statements or row data are written to a MySQL Bin Log and gradually transmitted to the Slaves. There the modifications are eventually applied, replicating the modifications that were made on the Master. The Slaves operate independently from the Master, deciding how much to read and from which point in the Master Bin Log. The Slaves serially apply the changes using a single thread (to insure ACID compliance). The Slaves do this while also servicing other read transactions (to provide read transaction scalability against the Master's data). With traditional MySQL asynchronous replication (in 5.1), the Master does not wait for the Slaves at all. With newer MySQL semi-synchronous replication (in 5.5) the Master waits for one Slave to acknowledge that it has received and queued (but not applied) the Bin Log data into its relay log before the Master completes the transaction.

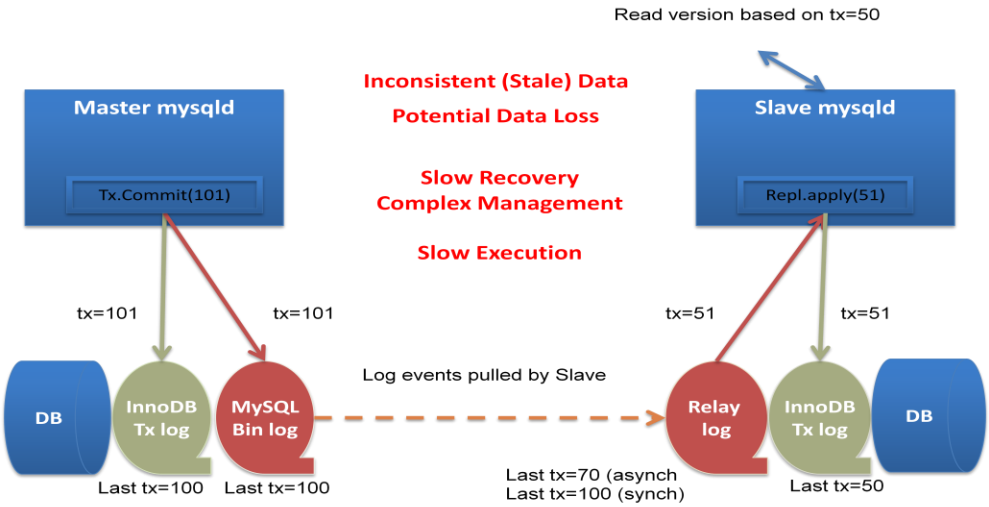

Figure 1: Loosely-Coupled MySQL Asynchronous and Semi-Synchronous Replication Architecture

Loosely-coupled asynchronous and semi-synchronous replication of the Bin Log hurt service availability, data integrity, performance, administration, and cost of ownership:

Reduced Service Availability

- When a Master fails, fail-over to a Slave is stalled until all transactions in the relay logs have been committed and a new Master established, and the remaining Slaves are reconfigured

Reduced Data Integrity

- Since Slaves are arbitrarily behind the Master in reading and applying changes, Slaves can give old (stale) data in response to read transactions.
- Slaves may not have the latest committed data from the Master, so data can be lost when the Master fails. (The semi-synchronous replication in MySQL 5.5 provides a partial solution to this problem compared to MySQL 5.1 asynchronous replication, with one Slave having the committed data in memory.)
- Checksums in the binary and relay logs are not generated and persisted to permanent storage, making data corruption possible (to be addressed in MySQL 5.6).

Poor Performance

- Applying committed database transactions from the relay log on Slaves is single-threaded (so as to provide serial consistency). This results in low utilization of Slaves and low throughput; more Slaves are needed to handle the required read transaction throughput levels, creating "server sprawl".
- Master throughput must be limited to match the Slaves' performance so the Slave databases are not too far out of date (otherwise the recovery time could be very long when the Master fails). This results in lower Master update transaction throughput, forcing additional database partitioning (sharding).

High Administrative Complexity

- DBAs often bear a large burden of tedious, error-prone, and usually manual processes in common tasks such as recovery from a Master failure, Slave migrations or additions, and hardware or software upgrades.

Cost of Ownership

- Low utilization of Masters and Slaves coupled with high administrative costs increase capital and operating expenses, while higher downtime reduces revenue and can cost customers.

Tightly-Coupled MySQL with Fully Synchronous Replication

Figure 2 shows a tightly-coupled MySQL synchronous replication architecture. With synchronous replication that is deeply integrated into MySQL and InnoDB, the cluster can realize complete data integrity, high performance, cluster-wide consistency, and fast automated fail-over and recovery.

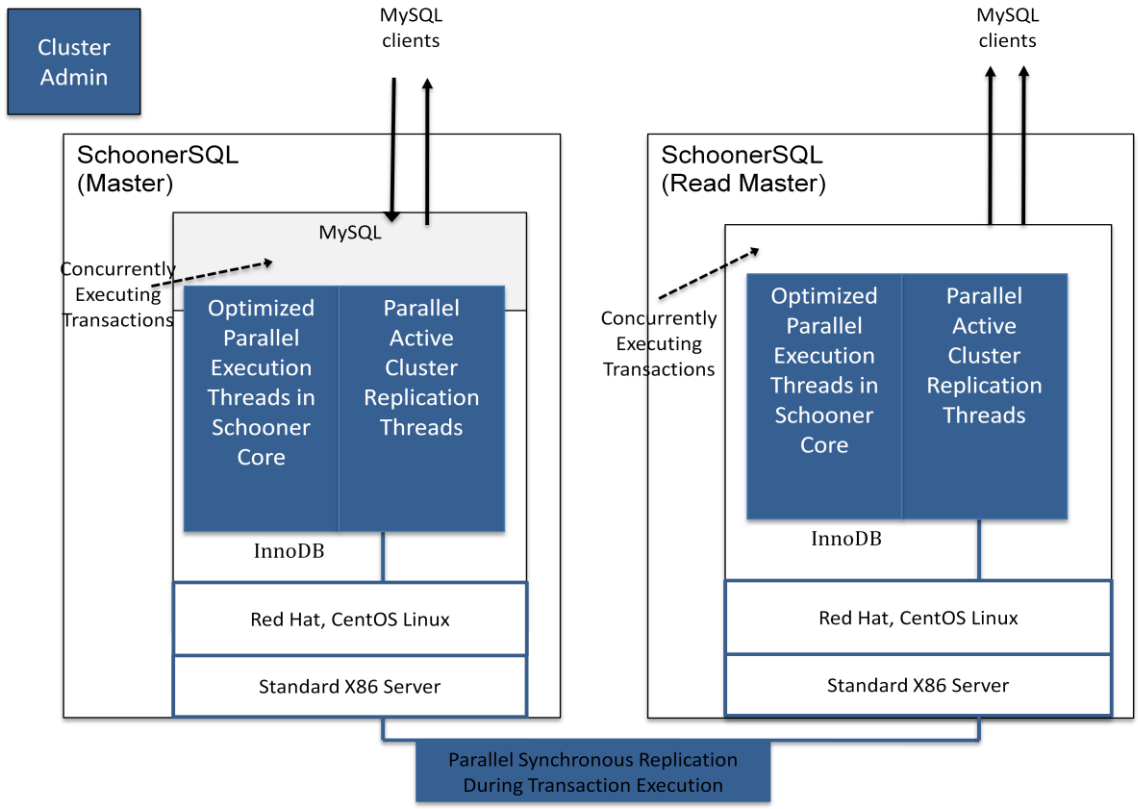

Figure 2: Tightly-Coupled SchoonerSQL Synchronous Replication Architecture

This is the architecture of SchoonerSQL™, a full distribution of MySQL + InnoDB that provides 99.999% availability. Multi-core and thread parallelism are used to concurrently communicate, replicate, and apply master update transactions on all Slaves with extremely high throughput and low latency. When all updates are initiated on the Master, no two-phase commit is required. This improves performance and eliminates forced application roll-backs. When the Master commits a transaction, all Slaves are guaranteed to have received and committed the update.

Tightly-coupled synchronous replication can dramatically simplify fail-over and on-going administration:

- When a Master failure is detected, fail-over can be automated and completed within a few seconds with no service interruption and no data loss.
- Since the Master and all the Slaves are at the same consistency level, any Slave can be automatically promoted to become the new Master.
- The Master's VIPs (Virtual IP addresses) can be instantly and automatically switched to the newly promoted Slave, so updates continue to be processed without any service interruption.
- When a Slave failure is detected its load can be automatically switched and load-balanced to other nodes.
- Once a failed Master or Slave is repaired, it can be automatically brought current with the Master and other Slaves, then automatically made active and load balanced.
- The Cluster Administrator, through a graphical user interface (GUI) or a Command Line Interface (CLI), can provide a central point for easy and powerful administration, monitoring and tuning.
- Hardware and software upgrades, on-line consistent back-ups, and instance migration can be accomplished with a simple point-and-click or a CLI command, all without service interruption.

Improved Service Availability

Tightly-coupled MySQL synchronous replication can provide much higher service availability than that achievable with asynchronous or semi-synchronous replication. For example, Figure 3 shows that SchoonerSQL reduces downtime by 85% or more compared to the asynchronous or semi-synchronous replication in MySQL 5.5 / 5.6, through automated Master and Slave fail-over and the ability to do on-line upgrades.

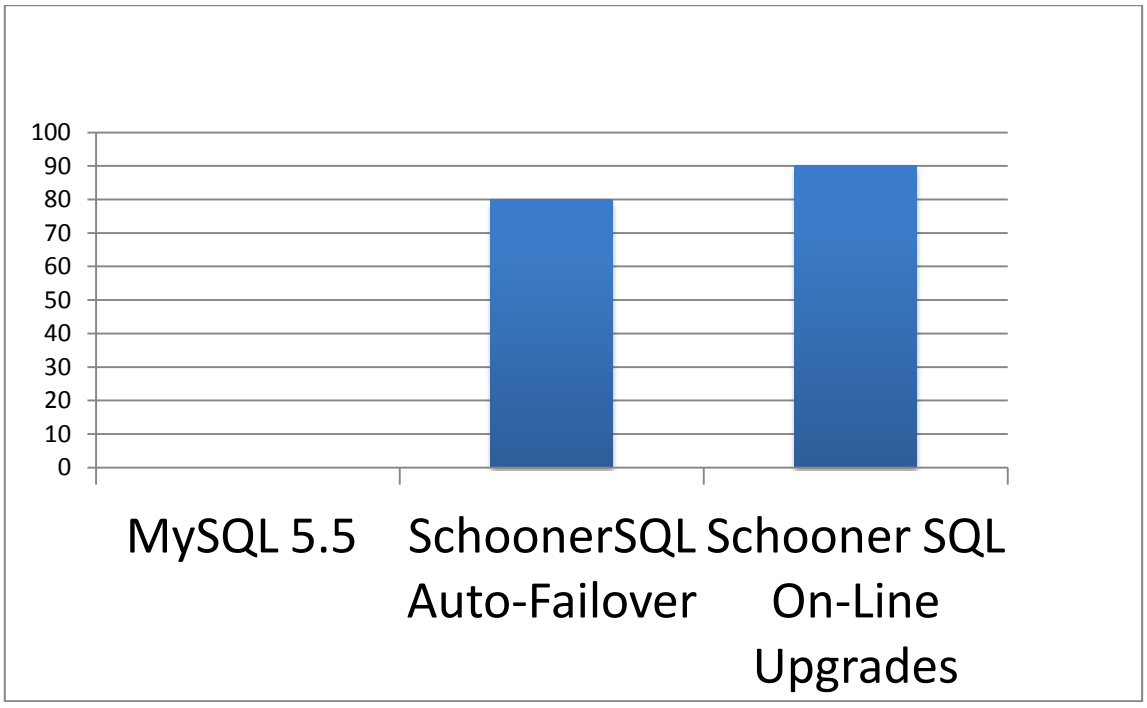

Figure 3: Percent Cumulative Downtime Reduction from SchoonerSQL

Improved Data Integrity

Tightly-coupled synchronous replication can completely eliminate the data loss and data inconsistencies present with MySQL asynchronous or semi-synchronous replication. With tightly-coupled synchronous replication:

- Reads on Slaves always provide the latest committed data, resulting in full cluster-wide data consistency
- Slaves always have the latest committed data from the Master, so there is no data loss if the Master fails
- Checksums on logs are persisted, so data corruption is detected and corrected.

Reduced Administrative Complexity

With tightly-coupled synchronous replication, fail-over and recovery can be completely automatic and instant, requiring no administrator intervention. Also, a synchronous replication Cluster Administrator GUI and CLI can provide a single point for cluster-wide and per-instance management, monitoring, trouble-shooting, and tuning. For example, the SchoonerSQL Administrator supports single-click actions to perform hardware and software upgrades, slave migrations, provisioning, and concurrent on-line backups. The SchoonerSQL Administrator GUI, shown in Figure 4, allows simple yet powerful management of cluster nodes, replication groups, database instances, and on-going administration activities.

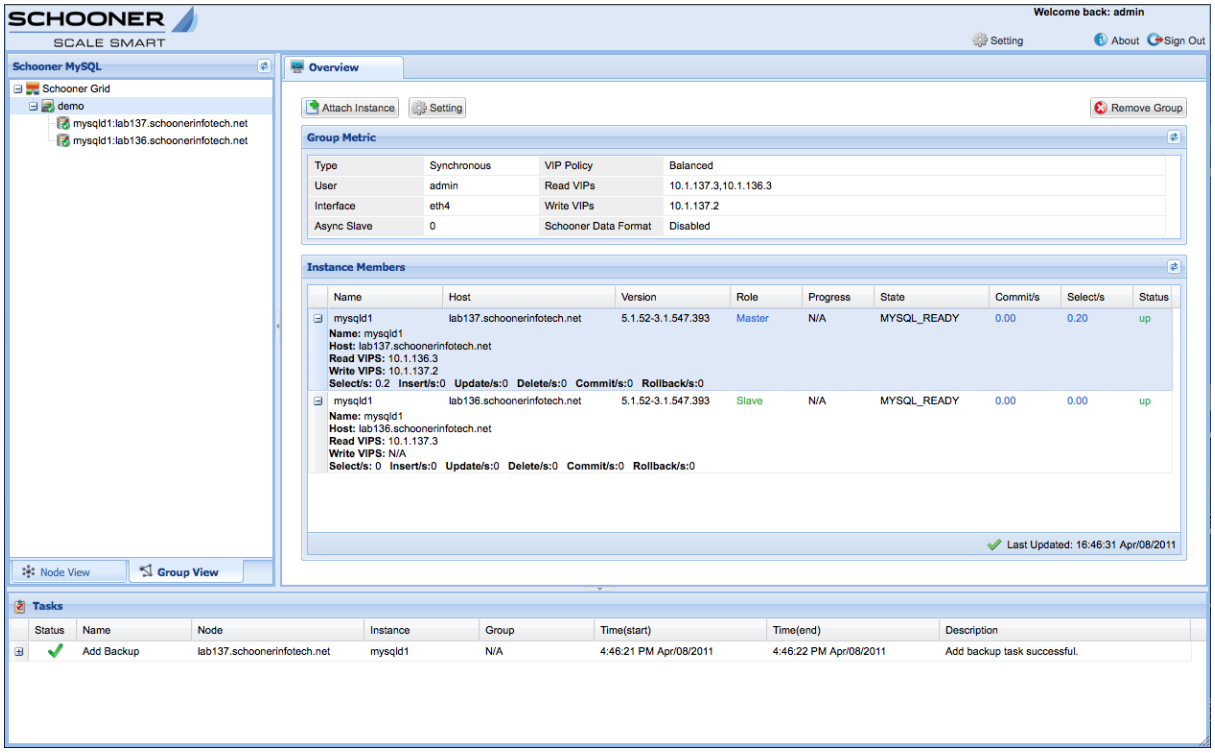

Figure 4: SchoonerSQL GUI to Administer Synchronous Replication

High Performance

Tightly-coupled MySQL synchronous replication can increase performance dramatically over asynchronous or semi-synchronous replication when implemented with high thread, core and data structure parallelism. This is shown in Figure 5, which compares SchoonerSQL synchronous replication to MySQL 5.5 and its asynchronous and semi-synchronous replication.

These measurements used the DBT2 open-source OLTP version of TPC-C at 1000 warehouses and 32 connections and zero think-time. The configuration under test was a 2-node Master-Slave cluster, with the Master and Slave each running on a standard 2-socket Westmere server with 72GB DRAM. The measurements were taken in steady state after 2 or more hours of warm up. Measurements were made with two storage configurations: one with the database stored in 8 typical (15,000 RPM) hard drives (HDD) and the other with the database stored in flash memory (using Fusion-io).

The performance of the MySQL 5.5 Master is the throughput at which Slave lag is stable and does not exceed 1 second. (Allowing Slave lag to grow can result in unbounded Bin Log size and unbounded recovery time, and is not an acceptable practice with MySQL). SchoonerSQL guarantees zero slave lag due to synchronous replication and immediate, transparent fail-over, so Master updates execute at full throughput.

Figures 5a and 5b show the performance throughput, measured in thousands of transactions per minute. In each case all measurements were done on the same hardware: a 2 node Master - Slave configuration; each node running 2 sockets with a total of 16 cores, and using 72 GB of DRAM. The benchmark is the DBT2 open-source OLTP version of TPC-C. It was set for 1000 warehouses using 32 connections, with zero think time.

In 5a, with the database stored on hard drives (HDD), SchoonerSQL transaction throughput is over three times higher than that of MySQL 5.5 asynchronous replication and over two times higher than MySQL 5.5 semi-synchronous replication. In 5b, with the database stored on flash drives, SchoonerSQL transaction throughput is four to five times higher than that of MySQL 5.5 asynchronous or semi-synchronous replication, and 10 times higher throughput than can be achieved on a pure HDD configuration.

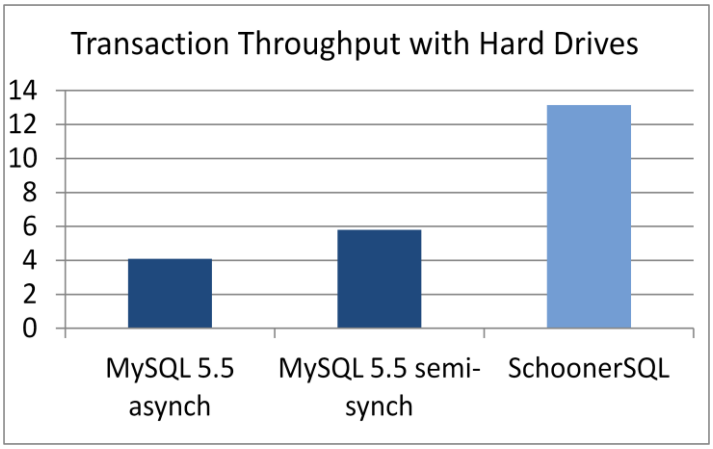

Figure 5a: Synchronous Replication Cluster Performance Throughput (kTPM, 1,000 warehouses)

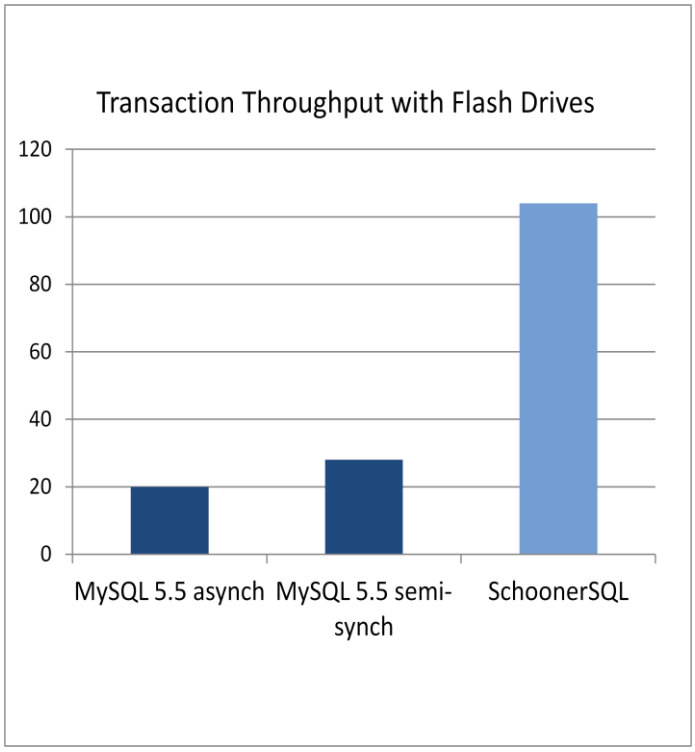

Figure 5b: Synchronous Replication Cluster Performance Throughput (kTPM, 1,000 warehouses)

Highly-parallel synchronous replication can also deliver much more stable throughput and faster response times compared to asynchronous or semi-synchronous replication. It can provide large immediate availability and performance benefits with HDDs and an easy in-place growth path for increasing throughput by an additional 10 times by adding flash memory.

Lower Cost

Tightly-coupled synchronous replication can deliver much lower cost of ownership compared with asynchronous or semi-synchronous replication. For example, Figure 6 shows that SchoonerSQL is 70% cheaper than MySQL 5.5. Its primary savings come from reduced capital and operating costs realized through a reduction in servers based on a minimum 3x performance improvement. Further savings come from the 90% reduction in downtime and the associated lower risk of losing revenue or customers.

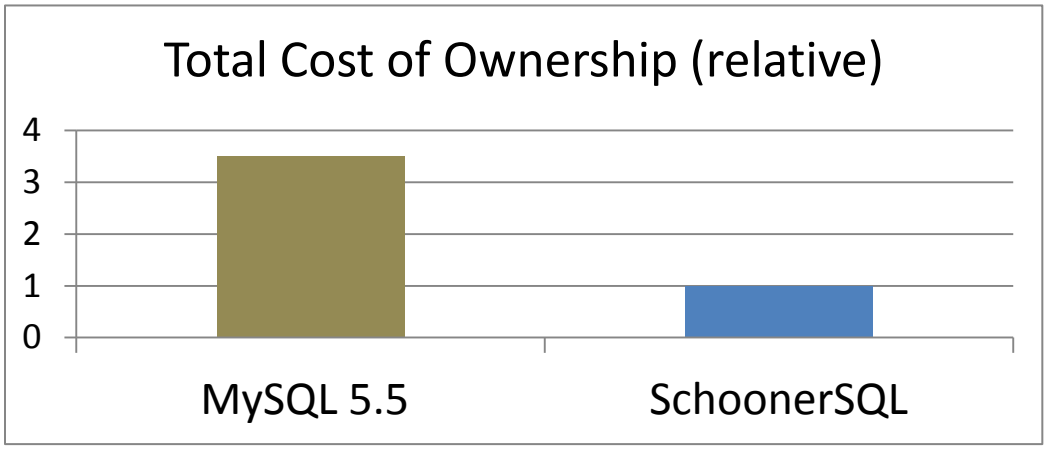

Figure 6: Cost of Asynchronous, Semi-Synchronous, and Synchronous Replication

TCO and ROI models are specific to each customer's application, architecture, and workload. Major variables include throughput per server; server, rack, and network costs; software license and support costs; the cost of space and power; the cost of DBAs to administer the servers; and the value of avoiding downtime. Nonetheless, this data is illustrative of the compelling economics of SchoonerSQL.

Compatibility Considerations

When selecting a synchronous replication solution, compatibility may be an important consideration. Both SchoonerSQL and Oracle MySQL Cluster provide synchronous replication and its high-availability benefits including automated fail-over, on-line addition and upgrades of nodes, etc. However, Oracle MySQL Cluster is not compatible with MySQL Enterprise with InnoDB applications. SchoonerSQL is a full MySQL with InnoDB distribution and is fully compatibility with all InnoDB applications and data, so no migration is required.

MySQL External Replication

In addition to MySQL–Specific replication, there are several solutions external to MySQL which address service availability and data integrity issues. The most widely-used are:

- Oracle Golden Gate: This converts the MySQL asynchronous Bin Log to a common log format to provide heterogeneous database replication interoperability with Oracle, IBM DB2, and Microsoft SQL Server, or other MySQL instances. Since Golden Gate is a very loosely-coupled external replication service, its performance is significantly worse than that of MySQL 5.5 for MySQL Master - Slave environments. Since Golden Gate uses the MySQL asynchronous or semi-synchronous replication Bin Log, it is even more loosely-coupled. It therefore suffers from the same reduced service availability, poor data integrity, high administrative complexity, and high cost discussed above for MySQL 5.5 when used in MySQL Master - Slave deployments.

- Continuent Tungsten Replicator: This converts the MySQL asynchronous Bin Log to a transaction history log and uses JDBC through a client proxy to access MySQL indirectly. This approach enables heterogeneous database replication interoperability with PostgreSQL and other MySQL instances. Tungsten also provides a Global Transaction ID, which is used by Slaves to point to a new Master when an old Master fails. Tungsten has a SaaS & ISP capability allowing parallel replication across independent,

multi-tenant MySQL databases, but has no parallel replication within a single database. Since Tungsten is a very loosely-coupled external replication service, its performance is significantly worse than that of MySQL 5.5 for MySQL Master - Slave environments. Since Tungsten uses the MySQL asynchronous replication Bin Log, it has the same problems of reduced service availability, poor data integrity, high administrative complexity, and high cost discussed above for MySQL 5.5 when used in MySQL Master - Slave deployments.

- **Linux DRBD (Distributed Replicated Block Device)**: This provides active-passive mirroring at the block device level. After each commit, the stand-by server is guaranteed to have identical blocks on the device, so the stand-by storage is kept in lock-step with the Master. DRBD does prevent data loss, and limits downtime for Master failure since the stand-by Master can be typically restarted in minutes. But the stand-by server does not service any load, and all the Slaves are still operating with asynchronous or semi-synchronous replication, so the same issues of reduced service availability, poor data integrity, high administrative complexity, high cost, and poor performance discussed above for MySQL 5.5 Master - Slave deployments are still present.

The SchoonerSQL™ Advantage

Schooner Information Technology created SchoonerSQL to solve the problems of other MySQL replication solutions. SchoonerSQL is a full distribution of MySQL + InnoDB, not an add-on module. SchoonerSQL reduces downtime, eliminates data loss, ensures cluster-wide consistent data, provides instant and transparent fail-over, and dramatically simplifies administration, all with great performance and scalability.

SchoonerSQL is an out-of-the-box software product that makes it easy to use MySQL in the most demanding mission-critical applications, with confidence. SchoonerSQL is downloadable to any x86 commodity server, and is completely compatible with all existing MySQL applications and datasets.

Let Schooner Prove Our Breakaway Benefits in Your Shop with Your Data

Highest Availability

- No service interruption for planned or unplanned database downtime
- Instant automatic fail-over
- On-line upgrade and migration
- 95% less downtime vs. MySQL 5.X
- High performance WAN with auto-failover

Highest Data Integrity

- No lost data
- Synchronous cluster-wide consistency
- Asynchronous cluster consistency
- Reduces slave lag by 95%

Visibility and Control

- Easy cluster administration
- No error-prone manual processes
- Monitoring and Optimization

Highest Performance and Scalability

- 4-20x more throughput/server vs. MySQL 5.5
- High performance synchronous replication
- Transparent sharding (with dbShards)

Out-of-the-box Product

- Full MySQL + InnoDB: not a toolkit
- Free your staff to build your business, not a custom database

Compelling Economics

- Cut server capex (consolidation)
- Cut opex (power, pipe, DBA time)
- TCO 70% cheaper than MySQL 5.5

100% MySQL/InnoDB Compatible

- **You can download and try on your servers today for free at www.schoonerinfotech.com/free_trials!**
- SchoonerSQL replication provides higher availability, better performance with clustering and replication, and easier administration than any other MySQL distribution, where the database is on hard drives, SAN, or flash memory (including Flashcache).
- SchoonerSQL works with all MySQL InnoDB applications and databases; no schema or query changes or any other kind of migration is required.
- SchoonerSQL also supports traditional MySQL asynchronous replication. You can keep using your existing WAN replication. And you can interoperate in mixed environments with legacy MySQL Masters and Slaves as either an asynchronous Slave or asynchronous Master.
- Schooner and our partners are here to support you with advice on your MySQL replication architecture and with extensive customer support to ensure your success with SchoonerSQL.