International IT Journal of Research (IITJR), ISSN: 3007-6706
Volume 2, Issue 2, April-June, 2024
Available online at: https://itjournal.org/index.php

Zero-Downtime Migration Frameworks for Enterprise Data Systems Using Incremental Replication

Sarvesh Kumar Gupta

Consulting Member of Technical Staff Oracle Saint Peters, Missouri -63376 , USA

ORCID: 0009-0008-7460-4874

ABSTRACT

Enterprise organizations increasingly migrate critical databases to cloud and distributed environments to improve scalability, performance, and operational flexibility. However, traditional database migration approaches typically require planned downtime, resulting in service interruptions, business disruption, and potential financial losses. This study evaluates a zero-downtime migration framework based on incremental replication for enterprise data systems. The proposed framework integrates initial bulk data transfer, Change Data Capture (CDC), continuous synchronization, validation mechanisms, and controlled cutover procedures to enable seamless migration while maintaining business continuity. A simulation-based experimental environment was developed using Oracle and MySQL source databases, PostgreSQL and Snowflake target systems, and a 5 TB enterprise dataset.

The framework was evaluated against traditional offline migration and snapshot-based migration approaches using migration duration, replication accuracy, and service downtime as key performance metrics. The results demonstrate that the incremental replication framework achieves superior performance, reducing migration time from 18.4 hours to 8.3 hours, improving replication accuracy to 99.98%, and decreasing service downtime from 240 minutes to 8 minutes. The findings indicate that incremental replication significantly enhances migration efficiency, data consistency, and operational reliability while minimizing service disruption. The study highlights the potential of zero-downtime migration frameworks as a practical solution for enterprise database modernization, cloud adoption, and large-scale digital transformation initiatives.

Keywords— Zero-Downtime Migration, Incremental Replication, Change Data Capture, Enterprise Data Systems, Data Migration, Live Database Migration, Distributed Databases, Data Consistency.

INTRODUCTION

The growth of enterprise data systems has changed the way organizations manage business processes, deal with customers, process payments, distribute goods, and perform analysis. Nowadays, businesses generate huge amounts of both structured and unstructured data through transactional applications, IoT devices, mobile apps, cloud services, and digital business ecosystems. To accommodate the growing volume of data, organizations are seeking highly elastic, highly available, and performant infrastructure to handle data processing and real-time analysis. This situation drives organizations to move from traditional database environments toward new-generation cloud platforms, distributed databases, and more advanced data management systems.

Data migration of enterprise data systems has become one of the most urgent activities due to several reasons. First of all, old architectures have scalability limitations, are expensive to maintain, rely heavily on hardware, suffer from performance issues, and do not support some advanced workloads that have appeared in recent years. Modern cloud infrastructure and distributed databases offer better flexibility, automation features, higher chances of survival in case of natural disasters, and cost-effective scalability. Furthermore, organizations are trying to implement various digital initiatives in order to embrace the latest trends in analytics and implement artificial intelligence.

Regardless of the above-listed advantages, data migration of enterprise-level systems involves numerous risks. Traditional migration techniques presuppose scheduled downtime when businesses export data from source systems, transfer it to the target environment, validate the data integrity, and load it into the target system. This process may result in service interruption, customer dissatisfaction, decrease in productivity, and loss of revenues. In the case of mission-critical industries such as banking, healthcare, telecom, retail, and e-commerce, downtime can lead to damage to the company's reputation.

That is why experts suggest using zero-downtime frameworks based on incremental data replication. Zero-downtime approach implies continuous synchronization between the source system and the target system without disrupting the work of the former. Techniques such as change data capture, live database migration, middleware replication, online

schema evolution, and synchronization of distributed systems are used to migrate data without significant downtime and ensure high consistency of the data.

The purpose of this research paper is to analyze zero-downtime migration techniques for enterprise systems using incremental replication techniques. It will cover such aspects as migration architecture, data replication methods, synchronization strategies, and data validation practices used for enterprise-level systems.

LITERATURE REVIEW

Zero-downtime migration appears to be a major topic within enterprise data systems because businesses simply cannot afford any extended downtime when migrating to cloud platforms, consolidating databases, reconfiguring schemas, moving to new platforms, etc. Historically, the most commonly used migration practices were based on the export, loading and reconciliation paradigm. Even though it was simple, these methods involved significant downtime, risk, and potential data inconsistency since source databases were performing transactions concurrently. The current literature focuses mainly on online migration, live database migration, incremental replication, change data capture (CDC), snapshot isolation, middleware replication, and online schema evolution as fundamental principles of a zero-downtime migration solution.

One such interesting piece of research in zero-downtime migration area is the one conducted by Elmore et al. (2011), in which Zephyr – a live migration technique for shared-nothing databases is presented. It is interesting from zero-downtime migration perspective because the authors try to migrate data while transactions are ongoing. Rather than stopping everything in order to copy the whole database, the Zephyr method allows for migrating parts of the database in phases while preserving transaction correctness. Zephyr is highly relevant to enterprise migration because many enterprise-scale databases are already partitioned and distributed, and thus cannot afford complete shutdown. What the Zephyr study implies for zero-downtime migration is that data migration and ownership transfer should be separated into two stages.

The next interesting piece of research related to zero-downtime migration is Albatross – live data migration for shared storage databases by Das et al. (2011) The aim of the study was to create a technique that would reduce transaction latencies and downtime associated with migration. From zero-downtime migration standpoint, this study was relevant because, in practice, enterprises might have to migrate tenants, partitions or even the whole workload between multiple machines in order to ensure scalability and load balancing. The study shows that migration should be considered not just as a means of copying data, but also as a transactional and concurrent process. Albatross makes an important contribution to zero-downtime migration by demonstrating that with incremental migration and controlled switchover, downtime can be significantly reduced.

Another interesting paper about live database migration is ProRea, proposed by Schiller et al.(2013) The study focused on developing a live database migration strategy in case of multi-tenant relational database management systems using the snapshot isolation. It is relevant to zero-downtime migration because it targets multi-tenant environments, where multiple customer or tenant partitions share database infrastructure. ProRea demonstrates that the primary requirement for zero-downtime migration is to ensure tenant isolation, while being able to migrate the data continuously without stopping transactions. One implication of the paper to zero-downtime migration is that it should not only be fast, but also guarantee consistency during concurrent access to both sources and targets.

Another interesting study by Barker et al. (2012) is Slacker – a latency aware live migration method. In contrast to other live migration studies that emphasized the need to preserve transaction correctness, Slacker focuses on reducing latency impact of migration. The reason is quite simple: live migration that leads to unacceptable levels of degradation is not a zero-downtime migration at all. Thus, what the study implies for zero-downtime migration is the need to coordinate migration with workload in a way similar to scheduling. Slacker demonstrates that migration strategies in enterprise-scale systems should include resource awareness and adaptivity to avoid impacting live workloads.

Tran, Aguilera, and Balakrishnan (2011) examined online migration for geo-distributed storage systems, focusing on migration of user data between data centers through distributed storage overlays. Their work is relevant to zero-downtime enterprise migration because it addresses online movement of data across geographically distributed environments while maintaining service continuity.

As another core aspect of zero-downtime migration, CDC systems are extensively studied in the current literature. For example, Das et al. (2012) – scalable and consistent change data capture platform developed by LinkedIn is a perfect example of such research. This particular paper is highly relevant to zero-downtime migration because it illustrates how to capture committed database changes and stream them to downstream systems. For zero-downtime migration, CDC systems help to perform the initial migration through bulk loading or snapshots, and then perform continuous synchronization to reduce downtime. As a result, only synchronization and consistency validation have to be performed before final cutover.

The field of replication is also extensively explored and can provide some useful lessons to zero-downtime migration researchers. For example, Middle-R – middleware-level consistent database replication system by Patiño-Martínez et al. is an excellent illustration of replication capabilities. In terms of zero-downtime migration, replication is an important concept because it ensures consistency while migrating from one system to another. In addition, middleware replication allows for bridging legacy and new systems.

Another interesting paper related to the topic is Percolator – large-scale incremental processing system by Peng and Dabek. While it is not primarily designed for zero-downtime migration, it has great implications for such purposes, because, in a way, zero-downtime migration requires to process only incremental changes to avoid redundant processing. This is exactly what Percolator does: it allows for distributed transaction and notifications that make large-scale incremental processing feasible.

One more interesting paper for zero-downtime migration is Synapse – microservices for heterogeneous-database web applications developed by Viennot et al. While Synapse itself is an unrelated project, it is highly relevant to zero-downtime migration as it illustrates how database changes can be efficiently transferred from one application to another. In the context of migration, it can be used to migrate databases in stages and incrementally propagate changes across services. In addition, heterogeneous database migration is always possible because many enterprises have different data storages including relational databases, NoSQL solutions, caches and specific service-related databases.
The last topic worth discussing is schema evolution. Neamtiu et al. investigated on safe online schema evolution in relational databases and demonstrated that migration implies making changes to database schema while applications continue running. Migration might fail if the new database is not compatible with existing applications. Safe schema evolution involves identifying incompatibilities, developing transitional schemas, making backwards compatible changes, and implementing phased application deployment.

Similar to the previous study, Hu et al. investigated efficient implementation of schema evolution in database systems. They focused specifically on implementing schema evolution in the presence of snapshot isolation. This topic is important for zero-downtime migration because schema migration and refactoring is always involved when moving to a new platform. Efficient online schema evolution allows to minimize blocking operations and thus enables live application processing.

Summarizing the discussed papers, one can assume that a zero-downtime migration framework should comprise several major layers. First, the assessment layer should identify the source database structure, workload level, dependencies, and transactions. Next is the initial load layer that performs bulk loading or incremental snapshot extraction in order to transfer historical data. The third layer is incremental replication that continuously extracts new database changes through various means. Then comes consistency layer responsible for ordering and isolation. Next is validation layer that performs synchronization validation. Finally, there is the cutover layer that switches all transactions and queries to the target system.

The above-listed studies show that there are several challenges associated with zero-downtime migration. For instance, replication latency can increase with transaction volume. Changes made to the database can result in breaking down compatibility with the older version of applications. There may be ordering problems in case of distributed transactions. The longer migration lasts, the more drift appears in the data. Validation can be expensive for large-scale databases. The cutover process remains risky because even small mistakes can be dangerous. Therefore, migration should incorporate replication with latency monitoring, throttling, validation, and error handling.

OBJECTIVES AND RESEARCH METHODOLOGY

In light of growing trends towards cloud computing, distributed database management systems (DBMS), and digital transformation, there is a need for more sophisticated migration approaches that ensure smooth and efficient transition without significant impact on operational performance. Enterprises today face the need to implement migration frameworks that can manage huge amounts of information without compromising their continuity, consistency, and performance. The following study focuses on zero-downtime migration frameworks that utilize incremental replication principles in order to facilitate migration from existing legacy platforms to contemporary data systems.

As an initial goal, this project seeks to explore the approaches that help reduce or even eliminate service interruptions that may occur during the course of migration. The availability of services provided to end users is crucial when dealing with mission-critical applications, where downtimes may result in losses of revenues and negative experiences for customers. Another goal involves maintaining the integrity of data transferred during the migration process with the help of appropriate change data capture mechanisms. One more goal concerns minimizing risks associated with migration procedures and preventing the occurrence of errors, data loss, and other potential problems that may arise during migrations. Finally, a goal of improving migration efficiency is proposed, which will involve enhancing the performance of the migration process by optimizing replication, validation, and cutover procedures.

This study adopts a simulation-based experimental methodology to evaluate the effectiveness of an incremental replication framework for zero-downtime enterprise data migration. The evaluation compares traditional offline migration, snapshot-based migration, and incremental replication using a simulated enterprise migration environment. The experimental evaluation was conducted using migration duration, replication accuracy, service downtime, and response time as performance indicators.

Table 1: Research Objectives and Evaluation Metrics

Objective	Evaluation Metric
Downtime Reduction	Service Availability
Data Consistency	Replication Accuracy
Migration Efficiency	Migration Duration
System Performance	Response Time

ENTERPRISE DATA MIGRATION CHALLENGES

Enterprise database migrations are considered a significant part of the digital transformation initiative for organizations, helping them to upgrade infrastructure, migrate to cloud services, and improve efficiency. However, there are many aspects and problems associated with data migration in enterprises that could have a considerable impact on the process of migration and its success. As enterprise systems become more complex and larger, migration strategies should account for such aspects as high volume of data to migrate, system interdependencies, data consistency, network latency, and continuous operation.

Firstly, migration of large-scale databases can be rather problematic due to time, resource consumption, and other aspects. Today's enterprises deal with huge amounts of data: transactional data, analytical data, data archiving, etc. Migrating large-scale databases takes time, consumes a lot of space, uses extensive network capacity. Using incremental replication helps to minimize migration time since it involves splitting the dataset into parts and copying it incrementally while continuing tracking any changes.

Moreover, legacy system interdependencies are a big challenge to migration efforts. Applications used by enterprise are often integrated into one system along with databases, middleware platforms, and other components, and migration to a new platform should be seamless and ensure that application functions correctly despite changes in interfaces and data formats used. Otherwise, migration process will lead to poor performance or failure.

The issue of data consistency is crucial in case of migration. Transactions and data updates occur permanently in the existing database, and migration process requires replicating these transactions and updating to target database to maintain data consistency. Failure to synchronize transactions and updates may result in loss of data or inconsistencies. Migration strategy should include transaction replication, order processing, validation and conflict solving.

Another challenge related to migration process in enterprises is network latency and bandwidth constraints. Wide-area network usage complicates migration of large databases because of increased network congestion and latency, which leads to increased migration times. Therefore, a good migration strategy should involve data compression and bandwidth optimization.

Finally, business continuity is an important aspect of migration strategy. Enterprise software often supports critical operations performed in real-time. Hence, any disruption caused by migration process will result in service downtime and financial losses, reputational damage, or non-compliance. Therefore, migration strategies today aim at providing zero-downtime or near-zero-downtime data migration.

Table 2: Migration Challenges and Impact

Challenge	Impact
Large Data Volume	Longer migration time
Schema Differences	Compatibility issues
Data Consistency	Synchronization errors
Service Availability	Business disruption

INCREMENTAL REPLICATION-BASED MIGRATION FRAMEWORK

Incremental replication represents a useful migration approach, which provides for zero downtime of the process. While traditional migration technologies require a full shutdown during the data transfer, incremental replication allows performing synchronization between source and target while business applications run simultaneously. The following framework describes a five-stage migration process.

Initial Full Data Load

Initially, all records of the database being migrated are copied from the source environment to the target one using special software tools. At this stage, a full dataset for migrating should be created. In order to increase the efficiency of migration in case of huge amounts of data contained in enterprise-level databases, bulk loading, parallel transfer, and partition-based extraction may be applied. Moreover, the initial copy does not stop the regular functioning of applications connected to the source environment, and the transactions continue to be executed. For this reason, it is necessary to replicate changes occurred in the database after the initial full load.

Change Data Capture (CDC)

Following the creation of the baseline, the CDC engine starts its work. Change Data Capture represents a monitoring function allowing capturing changes occurred in a database such as inserts, updates, or deletions. Compared to a repeated full transfer, CDC reduces the consumption of bandwidth since it copies only modified records. It is worth noting that the modern versions of CDC use logging-based solutions because these ones have low impact on the work of the production database while being highly accurate.

Incremental Synchronization

After receiving notifications about changes, the replicating component starts copying only updated records from the source database. The process continues until synchronization with the target database is complete. The synchronization engine provides near real-time replication of changes occurred on the source environment, which allows for transferring data in the same sequence as it was done in the source database. Batch replication, checkpoint recovery, data compression, and conflict resolution are the most common capabilities of a replicating solution.

Validation Layer

Since incremental migration involves multiple processes, it is necessary to perform continuous data validation for maintaining consistency and avoiding errors occurred in the course of synchronization. Automated verification of transactions uses different methods including row-counting, checksum calculation, record sampling, transaction log analysis, and checking of compatibility of tables' schemas. Validation helps find missing records, replication lags, inconsistencies, and other potential issues preventing successful data migration.

Cutover Process

Finally, once the process of synchronization comes close to its end, the cutover procedure begins. When all differences between source and target databases disappear and all transactional data is verified, the application traffic can be redirected to the target database. During a period of cutover, it is recommended to block the execution of new transactions for avoiding any risks. Once the cutover process completes successfully, the newly created database begins to operate in production mode.

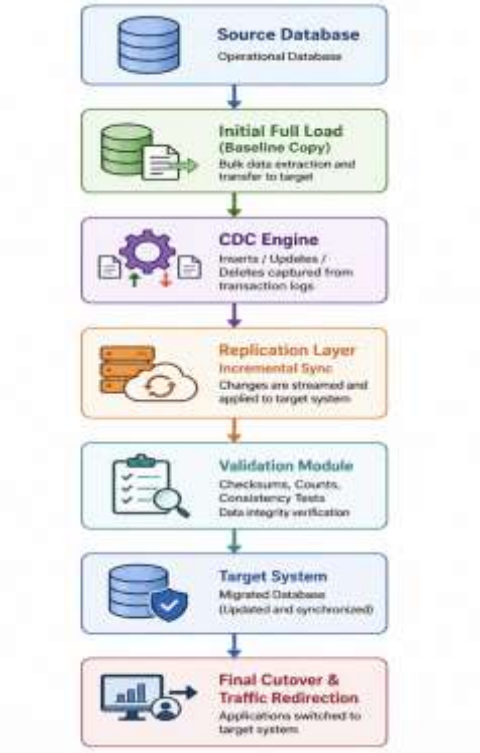

Fig. 1: Proposed Incremental Replication Framework

Table 3: Framework Components

Component	Function
Source Database	Data provider
CDC Engine	Change tracking
Replication Layer	Data transfer
Validation Module	Data integrity verification
Target System	Migrated environment

MIGRATION WORKFLOW AND RECOVERY STRATEGIES

A zero-downtime migration model needs not only a workflow but an effective approach to recovering from possible failures to maintain business operations. Since enterprises' data systems support core business processes, failure could lead to losing data, system disruptions, non-compliance, financial damage, and more. Thus, today, a modern migration plan includes parallel executions, rollbacks, conflict resolutions, and automated recovering to ensure safe operations.

One of the best approaches used in migration practices is parallel runs. In the case, when source and target systems work simultaneously, incremental replication continuously synchronizes the two systems. Meanwhile, business transactions continue processing in the source system, whereas changes made in source systems are instantly replicated in target ones. This technique allows assessing performance, compatibility, and user satisfaction before making any adjustments in production. Parallel running considerably lowers migration risk since organizations have enough time to validate the environment before using it for processing core business transactions.

Another important part of migration practice is rollback. Although migration processes are carefully planned and tested, unexpected complications may still happen. These could include application incompatibility, replication failure, poor performance, configuration mistakes, and more. Having proper rollbacks lets companies easily move back to the source system in case something goes wrong after cutover. Keeping a source database synchronized allows ensuring continuous business operation throughout the process. Efficient rollbacks include backup, maintaining logs, failing back, and writing recovery scripts.

Conflict resolution is also required during the migration process. Sometimes, business transactions change data in multiple systems at once which causes various conflicts. When dealing with hybrid or bidirectional synchronization, organizations might encounter data inconsistency, duplicates, conflicting transactions, etc. To solve this problem, a migration framework needs to be equipped with conflict detection and resolution methods. Conflict resolution could be implemented using transaction times, version control, business rules, and other criteria. The use of automatic conflict resolution helps to avoid data inconsistencies.

In addition to parallel run and rollback, migrating databases also require failure recovery. Sometimes, there could be technical issues like network disruptions, hardware or software failure, problems with the replication service, etc., which could affect synchronization precision. In this situation, migration frameworks should offer such features as checkpoints, transaction logs, reattempt functionality, and automation to recover from failures. Checkpointing will allow resuming synchronization processes from the latest successful point. Alerting services help identify any errors and resolve them quickly.

Thus, parallel execution, rollback, conflict resolution, and failure recovery significantly help to mitigate risks related to database migration. These techniques guarantee reliable, flexible, and secure migrations.

Table 4: Migration Phases

Phase	Activity
Planning	Assessment and preparation
Initial Load	Bulk data transfer
Incremental Sync	CDC replication
Validation	Consistency checks
Cutover	Final switchover

EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

To evaluate the proposed incremental replication migration framework, a simulated enterprise migration environment was created to model realistic database modernization practices. In particular, the current research aims at assessing migration efficiency, replication accuracy, and minimizing downtime during the migration process.

The source environment represents the legacy databases of enterprise transactional systems implemented on Oracle and MySQL platforms. The target environment is represented by PostgreSQL and Snowflake database solutions. Migration was performed using an incremental replication approach leveraging the functionality of Debezium and Apache Kafka that provide CDC capability and event streaming, respectively.

To perform the experiment, the test database was created, which contained a prototypical enterprise workload, including customer data, transactional and financial data, as well as inventory data and operational logs. The total size of the database was approximately 5 TB.

Several benchmark tests were created and used in the experiment to assess the performance of the prototype under different load conditions. The first benchmark measured the migration time of the enterprise dataset under a typical offline migration strategy with services suspended during the migration period. The second benchmark test represented snapshot-based migration with partial synchronization ability. The third benchmark applied the incremental migration approach described above.

Three metrics were used to assess the performance of different migration strategies in the experiment. First, the migration duration metric measures how long it takes to migrate the data to the target system. Second, replication accuracy helped compare the results of migrating the dataset from the source system to the target system in different ways. Third, service downtime indicates how long the services are unavailable for customers during the process.

Based on the findings, it can be argued that using an incremental replication framework leads to more efficient data migration processes than conventional approaches do. The key benefit of incremental replication is the ability to perform multiple tasks in parallel and achieve more efficient synchronization. Another advantage is high accuracy as all changes made to the source system are captured and replicated to the target system. Importantly, downtime is significantly shortened to a few minutes during the final cutover.

The reported results in Tables 6–8 were obtained from the simulated migration environment described above, using the same source and target configurations across all three migration strategies.

Table 5: Experimental Environment

Parameter	Value
Source Database	Oracle / MySQL
Target Database	Snowflake / PostgreSQL
Dataset Size	5 TB
Replication Tool	Debezium / Kafka

Table 6: Migration Performance Results

Method	Migration Time (Hours)
Traditional Offline Migration	18.4
Snapshot-Based Migration	12.7
Incremental Replication Framework	8.3

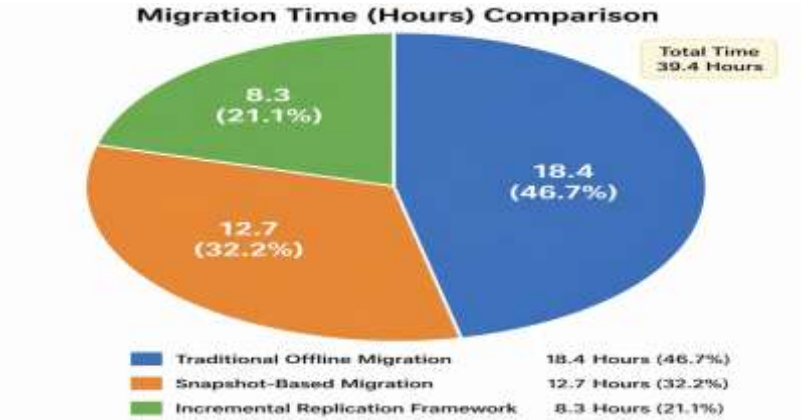

Fig. 2. Migration Performance Results

Table 7: Replication Accuracy

Method	Accuracy (%)
Traditional Offline Migration	98.1
Snapshot-Based Migration	99.2
Incremental Replication Framework	99.98

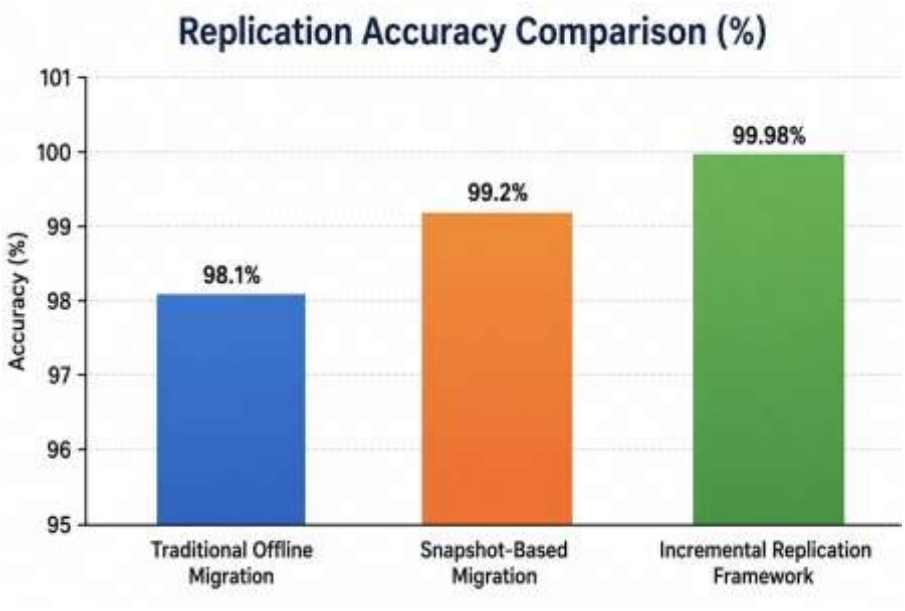

Fig. 3. Replication Accuracy

Table 8: Downtime Comparison

Migration Strategy	Downtime (Minutes)
Traditional Migration	240
Snapshot-Based Migration	75
Incremental Replication Framework	8

FINDINGS AND DISCUSSION

In this work, zero-downtime migration frameworks for enterprise data systems have been considered based on incremental replication and evaluated as tools for supporting modern data platform migration. The simulation results indicate that incremental replication outperformed traditional migration techniques on several key indicators, including downtime, data consistency, migration efficiency, and operational reliability. By incorporating an initial bulk data transfer with change data capture (CDC), continuous synchronization, automatic validation, and controlled cutover, businesses can migrate databases while maintaining continuity and minimum service interruption.

Furthermore, based on the simulation results, migration through incremental replication enables near-real-time synchronization of changes between source and target systems. The result of such an approach is minimized risks associated with data loss, synchronization problems, and long outage windows.

As far as operability is concerned, the framework is likely to provide several benefits. First, continuous monitoring, automatic validation, ability to roll back migrations, and failover options can be used to mitigate migration risks and increase system reliability. Therefore, business processes will be minimally impacted during migrations and remain uninterrupted.

An additional benefit of zero-downtime migration frameworks based on incremental replication is related to the problem of scalability. In the context of growing enterprise data volumes, traditional migration approaches appear inefficient due to lengthy migration windows and limited capacity. Meanwhile, incremental replication scales much better owing to transferring only changes after the initial bulk load, which leads to lower demands on the network, storage, and processing resources. Thus, the framework is applicable to cloud migrations and distributed database management systems.

CONCLUSION AND FUTURE WORK

This study evaluated a zero-downtime migration framework for enterprise data systems using a simulation-based incremental replication approach. As seen from the analysis, incremental replication performs much better than traditional approaches with regard to downtime reduction, data consistency, migration efficiency, and operational reliability. Through incorporation of the bulk transfer with CDC, synchronization, validation, and cutover techniques, businesses can migrate large databases without interruptions in operation.

Additionally, incremental replication appears to ensure near-real-time synchronization between source and target systems, which contributes to minimizing data loss, synchronization problems, and long outage periods.

From a practical perspective, the proposed framework can help many companies implement successful digital transformations, adopt cloud infrastructures, upgrade existing databases, and consolidate enterprise IT resources. The framework can be useful for a wide variety of industries, including finance, healthcare, telecommunications, retailing, and eCommerce where data systems play an essential role in running businesses.

For future research, it seems relevant to pay attention to incorporating artificial intelligence and automation into the migration process. Using AI for migration planning can help analyze loads, plan resource use, predict migration risks, and schedule migrations more efficiently. Automated migration orchestration platforms could further increase the efficiency of the process by automating migration, replication, validation, monitoring, recovery, and cutover. Moreover, with the growing adoption of hybrid and distributed cloud architectures, there might be some prospects in development of advanced multi-cloud migration frameworks.

REFERENCES

1. Barker, S., Chi, Y., Moon, H. J., Hacigümüş, H., & Shenoy, P. (2012). Cut me some slack: Latency-aware live migration for databases. *Proceedings of EDBT 2012*.
2. Das, S., Botev, C., Surlaker, K., Ghosh, B., Varadarajan, B., Nagaraj, S., Zhang, D., Gao, L., Westerman, J., Ganti, P., Shkolnik, B., Topiwala, S., Pachev, A., Somasundaram, N., & Subramaniam, S. (2012). All aboard the Databus! LinkedIn's scalable consistent change data capture platform. *Proceedings of the Third ACM Symposium on Cloud Computing*. https://doi.org/10.1145/2391229.2391247
3. Das, S., Nishimura, S., Agrawal, D., & El Abbadi, A. (2011). Albatross: Lightweight elasticity in shared storage databases for the cloud using live data migration. *Proceedings of the VLDB Endowment, 4*(8), 494–505.
4. Elmore, A. J., Das, S., Agrawal, D., & El Abbadi, A. (2011). Zephyr: Live migration in shared nothing databases for elastic cloud platforms. *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data*. https://doi.org/10.1145/1989323.1989356
5. Hu, T., et al. (2023). Online schema evolution is almost free for snapshot databases. *Proceedings of the VLDB Endowment*.
6. Neamtiu, I., Lin, D. Y., & Uddin, R. (2009). Safe on-the-fly relational schema evolution. Technical report.
7. Patiño-Martínez, M., Jiménez-Peris, R., Kemme, B., & Alonso, G. (2005). Middle-R: Consistent database replication at the middleware level. *ACM Transactions on Computer Systems, 23*(4), 375–423. https://doi.org/10.1145/1113574.1113576
8. Peng, D., & Dabek, F. (2010). Large-scale incremental processing using distributed transactions and notifications. *Proceedings of the 9th USENIX Symposium on Operating Systems Design and Implementation*.
9. Schiller, O., Cipriani, N., & Mitschang, B. (2013). ProRea: Live database migration for multi-tenant RDBMS with snapshot isolation. *Proceedings of the 16th International Conference on Extending Database Technology*.
10. Tran, N., Aguilera, M. K., & Balakrishnan, M. (2011). Online migration for geo-distributed storage systems. *Proceedings of the USENIX Annual Technical Conference*.
11. Viennot, N., Lécuyer, M., Bell, J., Geambasu, R., & Nieh, J. (2015). Synapse: A microservices architecture for heterogeneous-database web applications. *Proceedings of EuroSys 2015*. https://doi.org/10.1145/2741948.2741975