Journal homepage: https://itce.vn.ua/en

Information Technologies and Computer Engineering **Vol. 22, No. 3. 2025**

Article's History: Received: 10.08.2025 Revised: 22.10.2025 Accepted: 23.12.2025

UDC 004.65 DOI: 10.31649/vitce/3.2025.101

Comparison of data consistency models in distributed database management systems

Andrii Myrhorodskyi*

Postgraduate Student
Vinnytsia National Technical University
21021, 95 Khmelnytske Shose Str., Vinnytsia, Ukraine
https://orcid.org/0009-0007-6764-0060

Oksana Romaniuk

PhD in Technical Science, Associate Professor
Vinnytsia National Technical University
21021, 95 Khmelnytske Shose Str., Vinnytsia, Ukraine
https://orcid.org/0000-0003-0235-8615

Abstract. The use of distributed infrastructure to ensure scalability and high availability creates new challenges for maintaining data consistency between rapidly growing information system nodes that require reliable data management for correct operation. The aim of the study was to comprehensively systematise and comparatively analyse methods for ensuring data consistency in distributed database management systems, taking into account the fundamental trade-offs between consistency, availability and update delays described by the CAP and PACELC theorems. To achieve this goal, methods of theoretical analysis, formal modelling of system behaviour, and comparative expert evaluation were used. As a result of the study, consistency models were systematised according to two main approaches: data-centric and client-centric. The first approach analyses models that determine the global behaviour of the system: linearity, sequential, causal and eventual consistency. The advantages, disadvantages and typical application scenarios are identified for each model. The second approach considers client-oriented models that provide guarantees within a single user session: read and write consistency, monotonic read, monotonic write, and session causality. A generalised classification is proposed that visualises the relationship between the degree of consistency, delays, flexibility, fault tolerance and potential performance for each model. All considered data consistency models are compared using a number of selected essential characteristics (PACELC class, consistency, fault tolerance, potential performance, etc.) and diagrams based on their parameters. The practical value of the work lies in the formulation of clear recommendations for selecting the optimal consistency model depending on the requirements for reliability, performance, and architectural features of the information system. The results can be used to improve the efficiency of designing distributed databases in high-load systems, such as financial services, Internet of Things platforms, and cloud applications

Keywords: distributed systems; Data-Centric consistency; Client-Centric consistency; CAP theorem; PACELC; system availability; fault tolerance

Introduction

In modern conditions of rapid development in distributed computing systems and increasing demands for their speed and efficiency, the issue of ensuring data consistency in Distributed Database Management Systems (DDBMS) is becoming particularly relevant. Most contemporary information services utilise distributed infrastructure to achieve more flexible scalability, high availability, and improved fault tolerance. However, the use of these advantages is accompanied by a number of additional problems and limitations related to data consistency among the system's nodes, particularly during asynchronous replication, network delays, or the inoperability of individual components.

Scientific research in recent years has significantly expanded the theoretical and practical understanding of

Suggested Citation:
Myrhorodskyi, A., & Romaniuk, O. (2025). Comparison of data consistency models in distributed database management systems. *Information Technologies and Computer Engineering*, 22(3), 101-112. doi: 10.31649/vitce/3.2025.101

*Corresponding author

consistency models. For example, the work by P.S. Almeida (2024) proposed an axiomatic model for describing consistency in asynchronous distributed systems, which unites and generalises existing approaches. It also formulates the CLAM theorem for classifying system behaviour based on selected consistency guarantees: Closed past (finalised operations without the possibility of re-execution), Local visibility, Arbitration, and Monotonic visibility. According to the research, maintaining all four criteria for the complete set of data abstractions in systems without additional waiting is impossible. This offers an alternative perspective to the CAP (Consistency, Availability, Partition tolerance) and PACELC (Partition tolerance, Availability, Consistency, Else, Latency, Consistency) theorems, which can be used to derive and compare different consistency models.

A detailed overview of challenges regarding consistency and data integrity in DDBMS is presented in the work by H. Mahmoud & H. Yasin (2025), which focused on classic consistency protocols (such as 2PC) and modern replication methods. The Database Integrity (DDBI) project proposed by the authors offers an innovative solution through active triggers, rule-oriented integrity enforcement, and multi-version object tracking, which, according to the results obtained, is a better alternative to traditional approaches. Additionally, D. Russel *et al*. (2025) proposed an innovative approach to consistency verification in the context of serverless and edge architectures, which is extremely relevant in the transition to hybrid cloud environments. The authors used a comprehensive framework combining formal specifications, static analysis, real-time monitoring, and adaptive assurance, which also opens up prospects for using machine learning to predict consistency violations. Separate attention is warranted for the research by S. Ghasemirad *et al*. (2025), which implemented the Eiger-PORT+ protocol to ensure Transactional Causal Consistency with convergence (TCCv), confirmed by formal verification with TCCv isolation guarantees in Isabelle/HOL. The results refuted the previous hypothesis about incompatibility with transactional writes in the presence of performance-optimised read-only transactions. This represented the first full formal verification of a complex distributed database protocol and opened up the possibility of developing new abstract and practical models and protocols.

Concurrently, the article by Y. Chen *et al*. (2024) considered specific implementations of systems supporting various consistency models in cloud NoSQL platforms – from strong to bounded staleness. The work focused on supporting a wide range of load balancing, latency reduction, and resource management mechanisms popular in cloud environments for multi-model systems. In the work by N. Faria & J. Pereira (2025), the use of Conflict-Resistant SQL Views (CRDV) is proposed to ensure consistency in hybrid transactional and analytical workloads. An important distinction of this work is its focus on supporting CRDV in existing popular relational SQL databases, which allows for improved performance, greater flexibility in data merging strategies, and expands their use as heterogeneous DDBMS in large-scale information systems. Researchers also payed special attention to consistency algorithms in replicated systems, particularly CRDT (Conflict-free Replicated Data Types) approaches, which are reviewed in the works of Yu. Rabeshko & Yu. Turbal (2023), confirming the relevance of the topic in modern scientific discourse.

Thus, despite noticeable achievements in the field of consistency models, a need remains for a unified classification of existing approaches, a systematisation of their application conditions, and a determination of the algorithmic and engineering constraints imposed when implementing the respective models in DDBMS. Understanding these features can provide a better foundation for the further development of this area and the design of new data consistency models. The aim of this research was the formalisation and systematisation of methods for ensuring data consistency in DDBMS, taking into account the requirements of the CAP and PACELC theorems, as well as modern approaches to the implementation of Data-Centric and Client-Centric consistency models. Achieving this goal involved the execution of the following tasks: analysis and classification of modern consistency models; identification of the limitations and advantages of each approach in view of the system type; and determination of practical recommendations for model selection depending on usage scenarios.

Materials and Methods

The research is grounded in a theoretical and analytical method, which was used to study and compare data consistency models, as well as to analyse and interpret them within the context of the CAP and PACELC theorems. The initial stage of the study involved the collection and analysis of information about consistency models from relevant literature and available technical (or research) documentation. The models selected for the research were those that are widespread or have had a significant impact on the development of the DDBMS field, such as linearisability, which significantly influenced subsequent data-centric models. This stage also included the analysis of existing classification methods, such as the division into Data-Centric and Client-Centric models (proposed in the work by H.N.S. Aldin *et al*. (2019)), and methods for formally modelling the behaviour of distributed systems under various network conditions. This analysis was carried out based on the categorisation of systems according to the criteria of consistency (C), availability (A), partition tolerance (P), and update latency (L). For systematisation of the results, system types were categorised: CP, AP, CA (according to CAP), and PA/EL, PA/EC, PC/EL, PC/EC (according to PACELC (Abadi, 2012)).

Based on the obtained data, an analytical review of the consistency models was conducted, highlighting their core architectural principles, advantages, disadvantages, and typical application scenarios in distributed systems. This stage involved theoretical modelling of the Data-Centric and Client-Centric consistency models. A comparative method was used to contrast the characteristics and properties of the models, along with expert evaluation of their

parameters, including the level of consistency, flexibility, latency, fault tolerance, and potential performance. System-structural analysis was applied to build an independent classification of characteristics such as those primarily dependent on theoretical principles and those primarily dependent on implementation and to define the interrelationships between consistency models.

The next stage of the research was performed by visualising the acquired data in MS Excel to display the classification results and facilitate further analysis. To visualise the relationships between the models' properties, a generalised classification table of characteristics was used, which compares the model type, its correspondence to the PACELC theorem, and the evaluated expert parameters. The table allowed for the conceptualisation of the trade-offs between performance and consistency across different classes of DDBMS models, as well as trends in more modern models, such as the prioritisation of availability and latency according to the PACELC theorem. The use of radar charts (or spider diagrams) allowed for a graphical comparison of the characteristics' assessments obtained through theoretical analysis and enabled the tracking of their change and dependence on typical usage scenarios of modern information systems. The results of the analysis served as the basis for substantiating the choice of an appropriate consistency model type depending on the system's target application scenario, which allows for the adaptation of the DDBMS architecture to the specified requirements for availability, performance, or data integrity. The final stage of the research involved using the comparative method and content analysis to correlate the obtained data and methods for classifying characteristics with existing academic works in the field of DDBMS and distributed systems.

Results and Discussion

An information system consisting of only a single node is responsible for processing all possible read and write operations, whereas a distributed system has a defined set of nodes that do this concurrently. If a single-node system becomes inoperable, all operations become impossible to execute. One of the properties of distributed systems is the ability to continue functioning even when one or more nodes are inoperable. On the other hand, this creates the potential for node desynchronisation: network problems can interrupt the data replication process, and simultaneous write operations on different nodes may conflict with each other.

Data consistency is the property of a distributed system to maintain all nodes in a uniform and predictable state. Data consistency is closely related to and balances against other system properties and characteristics, such as performance and availability - ensuring consistency requires additional logic and can limit processing speed, but it provides specific guarantees regarding data synchronisation between nodes (Ahmed *et al.*, 2023). Data consistency mechanisms help order the execution of concurrent requests and provide options for system recovery from failures without data corruption.

The CAP and PACELC theorems are frequently used to describe the fundamental limitations in data consistency models in distributed systems. The CAP theorem (or Brewer's theorem) defines three key characteristics of distributed systems (Muñoz-Escoí *et al.*, 2019):

- Consistency: following any write operation, the distributed system must return the latest version of the data upon a read request, regardless of which node the request was made to (Lourenço *et al.*, 2015).
- Availability: the distributed system must provide a response to any incoming request, even if a portion of the nodes are inoperable at the moment the request is being processed.
- Partition tolerance: in the event of a loss of connection (or other network problems) between several nodes (splitting the cluster into several parts with limited communication), the distributed system must continue to operate and process incoming requests.

The CAP theorem states that a distributed system can only guarantee the simultaneous assurance of two out of the three described characteristics/guarantees:

- CP (Consistency & Partition Tolerance) systems will maintain data consistency in the event of network problems by limiting availability, i.e., the ability to process incoming requests. Without limiting availability, nodes with disrupted communication would be unable to synchronise completed write operations, which would lead to server desynchronisation and a violation of consistency.
- AP (Availability & Partition Tolerance) systems operate according to an algorithm opposite to CP systems nodes continue to process requests, ensuring availability even in the case of network problems, but the results of these requests may be inconsistent between nodes. AP systems often either limit the number of nodes accepting write requests or switch entirely to processing only read requests.
- CA (Consistency & Availability) systems always ensure consistency and availability and, therefore, cannot continue to operate if network problems occur either all nodes process requests and synchronise with each other, or continued operation is impossible. DDBMSs that support ACID transactions are typically included in this category of distributed systems.

The CAP theorem focuses its attention on the behaviour of a distributed system during cluster partitioning (when communication problems between nodes occur). Under normal network conditions, a system can maintain both consistency and availability simultaneously, regardless of its conditional type (CP, AP, or CA). Changes in behaviour and state occur only at the moment network problems arise. The PACELC theorem extends the CAP theorem by adding a new classification of behaviour in the absence of system partitioning due to communication problems. If there are network problems in the distributed system and the system can potentially be partitioned or partially inoperable, the choice described in the CAP theorem (between data consistency and availability) is applied. However, if

the system is operating under normal network conditions, the choice is applied between consistency (C) and update latency (L) (Golab, 2018). These relationships are visually represented in the Figure 1.

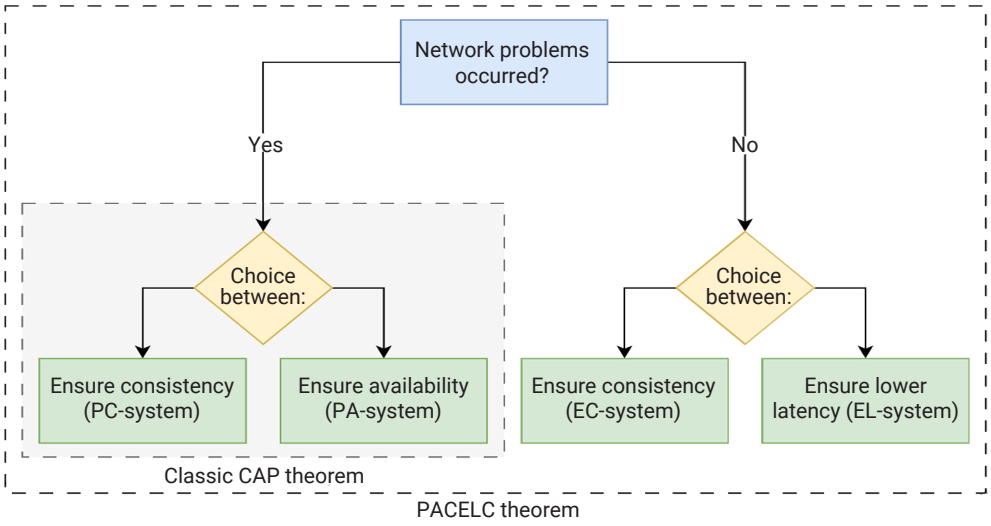

Figure 1. Choice in the PACELC theorem
Source: developed by the authors based on research by D. Abadi (2012)

If a system prioritises consistency, any data replication must include a guarantee that all nodes in the distributed system are updated before request processing continues. That is, a data write operation is considered fully complete when all nodes have received/replicated that operation. This guarantees the strict consistency required for ACID transactions but can decrease the speed (increase latency) of query execution due to the need for communication among all system nodes.

Conversely, if the distributed system prioritises lower latency and greater speed, an operation is considered complete after being written to just one or a few nodes (depending on the system's implementation). Node synchronisation in this scenario is considered a separate, asynchronous operation that can occur independently of subsequent write operations. Accordingly, some nodes in such a distributed system may be inconsistent with the rest of the cluster at certain times and return stale data.

Combining the behaviour options with and without network problems, the PACELC theorem divides distributed systems into four types (Gorbenko *et al.*, 2020):

- PA/EL: the system prioritises Availability during communication problems, and Lower Latency during normal functioning.
- PA/EC: the system prioritises Availability during network partitioning, but otherwise prioritises Consistency over lower latency.
- PC/EL: the system prioritises Consistency during network partitioning, but favours Lower Latency under normal conditions.
- PC/EC: the system prioritises Consistency both during network problems and under normal functioning.

It should be noted that the classification of both theorems is not rigid – some systems support several types of behaviour depending on the choice of the user or administrator of the DDBMS. For example, Amazon DynamoDB can be classified as a PA/EL system, i.e., availability and lower latency are prioritised over consistency, but the user may require strict data consistency when executing queries – in such cases, DynamoDB will function as a PA/EC system. In addition to classification according to CAP and PACELC theorems, there is also a division into data-centric and client-centric data consistency models (Aldin, 2019). Data-centric consistency models are built around the rules for replicating data in a distributed system as a whole. Such models describe the required behaviour of the entire system and its states. For example, an data-centric model may include rules for replicating operations, their ordering, and the logic for simultaneously processing read or write requests.

Client-centric consistency models describe the rules of operation within a single client or session, rather than the system as a whole. Such models typically provide consistency guarantees for operations from a single user, but the global state of the distributed system or the same operations for another user may be temporarily inconsistent. For example, a distributed system that prioritises lower latency can guarantee the consistency of all operations for a single client – if the same client changes the data and then reads it immediately, it will always receive the latest version. However, its operations will not be immediately replicated to other nodes.

Here are some examples of well-known methods for ensuring data consistency and their classification. Linearizability is an data-centric model that guarantees strict data consistency. This model is based on the global ordering of all operations of a distributed system into a single list. The actual process of system operation may involve several clients working in parallel with the same data, but

the model assumes that between receiving a request and responding to it, there is a moment when the operation is considered to be performed instantaneously or atomically (Park *et al*., 2024). Also, the time of sending the response to the request is used to order events, rather than the time of starting its processing. Using these two concepts, it is possible to transform any history of requests of a real distributed system into a single ordered list of events, which acts as a source of truth for all nodes and thus ensures strict data consistency. Implementing linearizability in an DDBMS will require the use of a data locking mechanism or another system for synchronous replication of operations between nodes, such as a consensus algorithm. Accordingly, such a system will belong to the PC/EC category according to the PACELC theorem.

Sequential consistency is another data-centric model that is similar to linearisability and orders operations into a single sequence. The main difference lies in the absence of ordering based on the completion time of the request execution. The global history must include operations from all nodes in the order they were executed on the respective server, but the ordering of operations from different nodes can be arbitrary and does not necessarily reflect their true execution order (Perrin *et al*., 2016). Thus, the sequential model provides less strict data consistency compared to linearisability. This simplification potentially offers higher performance for the distributed system, which is well-suited for distributed NoSQL databases used as a distributed cache. Accordingly, distributed systems based on the sequential model can be classified as PC/EL models under PACELC.

Causal consistency is a data-centric consistency model that only guarantees the preservation of the sequence of interrelated operations. For example, write and read requests for two different, unrelated table rows do not affect each other. Therefore, a database management system based on causal consistency may not maintain a strict order of their execution this will not affect the final state of the database or change the result of the user's operation (Junfeng *et al*., 2022). However, if the system has information about a potential link between operations, such as reading and updating the same value, their sequence must be preserved and correctly replicated to all system nodes to maintain consistency. In such a case, this model will function similarly to linearisability or sequential consistency.

This flexible approach gives causal consistency even greater potential operational performance, making it well-suited for DDBMSs with less strict consistency requirements. This model is more challenging to classify under PACELC, as its flexibility allows for more freedom in software implementation. Generally, a system based on causal consistency is less vulnerable to network problems and does not require forced node synchronisation unless the operations are related, so it can be classified as PA/EL.

Another well-known data-centric model is eventual consistency. It offers the weakest data consistency guarantees of all the models discussed so far, as it does not inherently require operations to be stored in a specific sequence or ordered on individual system nodes (Xu *et al*., 2024). Eventual consistency establishes a synchronisation rule for the data: in the absence of new updates, the distributed system will eventually reach the same state on all nodes, meaning it will become consistent. During request processing and until the synchronisation of all nodes is complete, a distributed database using eventual consistency may return stale data. Typically, such systems also have additional mechanisms for writing/updating data to avoid potential conflicts between nodes during concurrent operations.

Thus, eventual consistency only guarantees that the distributed system will be consistent at some point in time but does not impose stricter conditions or limitations on when this will occur. This simplification provides the highest potential request processing speed among all the data-centric models reviewed. Furthermore, eventual consistency is well-suited for information systems with high availability requirements, which is why it is used in NoSQL DDBMSs, such as Amazon DynamoDB. Accordingly, distributed systems based on eventual consistency can also be classified into the PA/EL class according to the PACELC theorem. A generalised comparison of the properties and application of the reviewed data-centric models is provided in Table 1.

Table 1. Comparison of data-centric consistency models

No.	Model name	Description	Advantages	Disadvantages	Application
1	Linearisability	Ensures a strict global ordering of all operations based on the time of the request response	The strictest guarantees of data consistency	High latency, complex implementation, low performance	Banking systems, critical transaction management systems, and systems with mission-critical data
2	Sequential consistency	Ensures a strict ordering of requests within a single server, but without coordinating the order between nodes	Simpler implementation than linearisability; strict data consistency	Does not guarantee the real-time order of operations in a distributed cluster	NoSQL databases, caching systems with low consistency requirements
3	Causal consistency	Guarantees the ordering of only causally related operations. Independent operations can be executed in any order	Simpler implementation; no need for global synchronisation	The need to track causal links; complexity of the logic depends on the data structure	Collaborative applications, social networks, and distributed collaborative editing environments
4	Eventual consistency	The system eventually reaches a uniform state (synchronises) across all nodes, but new updates may not be immediately reflected	Highest performance, minimal latency	Lack of strict guarantees; a possibility of receiving outdated data	Web applications, mobile applications, and caching systems.

Source: developed by the authors

Client-centric consistency models typically describe the rules for the interaction of a client (or several clients) with a single node in the system. Thus, consistency guarantees are provided only for requests sent specifically to that node, while the distributed system as a whole may be in an inconsistent state or synchronise according to other rules. The read-your-writes model guarantees that if a request to write/update data has been successfully executed, all subsequent requests to read data will return the updated value corresponding to the previous changes. The main difference between this model and others is the need to maintain the order of operations for only one client on a given node; there is no global ordering at the time of the request, and the system may appear inconsistent to other clients until synchronisation occurs (Aldin *et al.,* 2020). Due to the absence of global synchronisation rules, the read-write consistency model is somewhat similar to the eventual consistency model, but provides more stable behaviour within the operations of a single client (as long as that client is connected to the same node). Therefore, this model is well suited for DDBMSs for data caching or for information systems where a large amount of information is tied to specific users and the risks of conflicts during global synchronisation are lower. According to PACELC, this model can be classified as PA/EL.

The monotonic reads model is a client-centric model that guarantees data reads within a single session. If the client is connected to the same node, all data read operations will be consistent – the results of new queries cannot return an older version of information than that already provided in previous queries (Campêlo *et al.,* 2020). Similar to the read-write consistency model, such a distributed system does not require global ordering of operations and does not restrict the data synchronisation process – it can be asynchronous and initiated by an individual node as needed. On the other hand, the monotonic read model does not provide any guarantees for data update requests – they may appear inconsistent even within a single server. This model is well suited for implementing DDBMS for working with cache or information systems where most of the load consists of read operations. According to PACELC, this model can also be classified as PA/EL.

The monotonic write model is a data consistency model that guarantees the ordering of write operations according to the order in which the client initiated them (Taghinezhad-Niar, 2024). That is, if a client has made several writes in sequence, the order in which they will be written to the distributed system node will be exactly the same. It is important to consider some limitations of this model: first, ordering guarantees are only provided for operations of a single client, as in other client-centric models. That is, there is no global ordering, and the overall list of operations may differ on each node, but the operations of each individual node will have the correct order of execution. Second, this model does not provide consistency guarantees when reading data, so each client may receive both new and stale data in response to its query.

The monotonic record model, similar to other models of this class, can be classified as PA/EL type according to the PACELC theorem. A DDBMS with monotonic record consistency is most useful for applications with multi-user data editing, such as online document editors, so that individual users' edits are correctly ordered. However, this model can also be used in certain types of caching systems.

The Session Causality model (also known as Write Follows Reads Consistency) is a client-centric model that resembles a simplified version of the data-centric causal consistency model. While causal consistency orders all causally related operations at the level of the system as a whole, the Session Causality model tracks only the read-write operations of a single client (Viotti & Vukolić, 2016). The model makes the logical assumption that a data read operation might prompt the client to certain actions, including a data write operation. Accordingly, the write request must be processed on those distributed system nodes where the data is no older than what the client last read. This preserves the cause-and-effect relationship of operations within a single session. The session causality model can similarly be classified as PA/EL, and its application scenarios in distributed systems and DDBMSs are analogous to other client-centric models.

While data-centric models describe the logic of data consistency at the level of the entire distributed system, client-centric models give only limited guarantees concerning specific events, and in most usage scenarios, they do not conflict with each other. As a result, a distributed system may implement several different consistency guarantees at once this approach is distinguished as a separate model called Session consistency (Wang *et al*., 2024) or the Session consistency model. Typically, the session consistency model incorporates the guarantees of all four reviewed client-centric models: Read-your-writes, Monotonic Reads, Monotonic Writes, and Session Causality however, the actual implementation of the information system may include other or modified guarantees. Furthermore, it should be noted that all reviewed models only offer rules for data consistency within a session; the actual mechanisms for synchronising this data and resolving potential conflicts remain unaddressed. A generalised comparison of the properties and application of the reviewed client-centric models is provided in Table 2.

Table 2. Comparison of client-centric consistency models

No.	Model name	Description	Advantages	Disadvantages	Application
1	Read-your-writes	Guarantees local consistency within a user's session - after a write, a client only sees updated data	Guarantee of sequentiality for the user; easy to understand	Potential for conflicts between different clients	Systems with personal data, user profiles, and caching systems

Table 2. Continued

No.	Model name	Description	Advantages	Disadvantages	Application
2	Monotonic reads	Guarantees that a client will never read data older than what they were previously read	Stable consistency in reading	Does not guarantee correct writes; it is possible to get outdated information for other users	Analytical systems, reading logs, and statistical overviews
3	Monotonic writes	A single client's writes occur in the same order in which they were initiated.	Preserves the logic of changes.	No guarantees on reads; possible conflicts between clients	Online editors, version control systems
4	Session causality	Guarantees causal links between operations for a single client.	Strict consistency within a single session or user	Complex to implement, latency during relationship checks	Collaborative editing of documents, cloud platforms
5	Session consistency	A comprehensive model that combines all previous client-centric models	Highest local consistency; preserves client interaction logic	Does not guarantee global consistency; dependent on the user's session	Interactive web applications, personalised services, and mobile applications that maintain session state

Source: developed by the authors

The comparisons in Tables 2 and 3 were conducted based on the models' main advantages, disadvantages, and typical application scenarios. Such an analysis allows for the identification of development trends, particularly when comparing consistency models within the same class. For instance, in both tables, later models show not only improvements in functional characteristics, such as increased speed or local consistency, but also an orientation towards use in mobile and web applications. Since most models have varying degrees of formalisation without a golden standard implementation, it's impossible to compare them using specific numerical values. However, the existing classification systems for these consistency models can be expanded by relatively comparing individual characteristics and properties.

The model type comparison uses the classical division into data-centric and client-centric models. This characteristic should be considered first when comparing the models against each other, as, although most client-centric models offer a high degree of consistency, this consistency is limited to the session between the system node and the client. Only a data-centric model can fully ensure data consistency purely between the nodes of a distributed system. The PACELC Class describes the models' behaviour according to the PACELC theorem, allowing them to be quickly classified based on their actions under normal operation and in the presence of network problems. Other characteristics use relative assessments based on available theoretical data and application examples. The defined characteristics and metrics of the models are presented in Table 3.

Table 3. Characteristics of data consistency models

Consistency model	Model type	PACELC class	Degree of consistency	Latency	Flexibility	Fault tolerance	Potential performance
Linearisability	Data-centric	PC/EC	High	High	Low	Medium	Low
Sequential consistency	Data-centric	PC/EL	High/ Medium	Medium	Medium	Medium	Medium
Causal consistency	Data-centric	PA/EL	Medium	Low	High	Medium	Medium
Eventual consistency	Data-centric	PA/EL	Low	Low	High	High	High
Read-your-writes	Client-centric	PA/EL	Locally medium	Low	Low	Medium	High
Monotonic reads	Client-centric	PA/EL	Locally medium	Low	Low	Medium	High
Monotonic writes	Client-centric	PA/EL	Locally medium	Low	Low	Medium	High
Session causality	Client-centric	PA/EL	Locally high	Medium	Low	Medium	High
Session consistency	Client-centric	PA/EL	Locally high	Low	Medium	Medium	High

Source: developed by the authors

According to the assessments, the data-centric models demonstrate a gradation from strict consistency guarantees with high latency to weak guarantees with improved performance, which can also be linked to the changing application scope of the respective DDBMSs in Tables 1-2. client-centric models achieve a locally high level of consistency while maintaining low global latency, making them suitable for use in information systems without high data integrity requirements. It is worth noting that the majority of the reviewed models are classified as PA/EL, which indicates a greater focus on availability and low latency in modern DDBMSs. Furthermore, the selected metrics for relative comparison can be tentatively divided into two categories – those primarily dependent on theoretical principles and those primarily dependent on implementation, as shown in Figure 2.

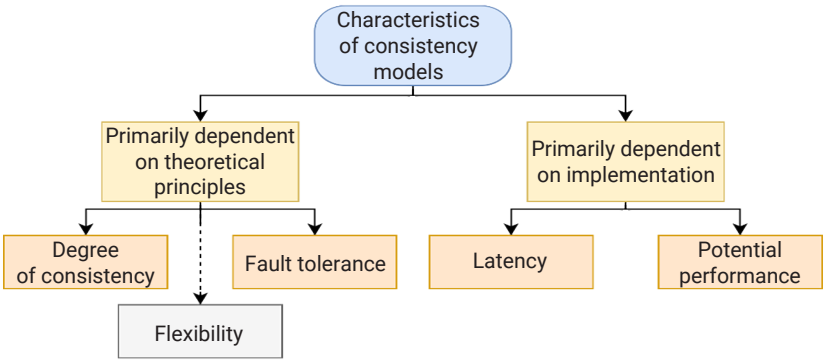

Figure 2. Classification of characteristics for comparing consistency models
Source: developed by the authors

This classification reflects only a tendency towards one side or the other. True (non-relative) assessments must be made only when a reference implementation is available for each of the data consistency models under review. For example, flexibility was classified as a characteristic primarily dependent on theoretical principles, but of all other metrics in this category, it is the most susceptible to the specifics of practical software implementation. Accordingly, it has a special designation on the diagram.

Degree of consistency and latency characterise the main parameters of the reviewed models what data consistency guarantees are provided between nodes and how strongly these guarantees affect performance. It is important to note that the assigned scores are relative and based on available theoretical knowledge, so they should only be used to compare the reviewed models with each other. Also, since client-centric models do not account for consistency with other nodes, a different measure of consistency is used for their classification.

Flexibility and fault tolerance characterise the potential ability of a distributed system with a certain type of consistency to adapt to changes (including the inoperability of individual nodes) and various usage scenarios. Automation of the recovery process, data replication frequency, and actual downtime are important factors when choosing both a disaster recovery strategy and the consistency model in which it will be used (Myrhodskyy *et al.*, 2023). Potential performance summarises all preceding characteristics for a relative comparison of the operational efficiency of different models against each other.

If scores from 1 to 3 are used instead of the "low"–"high" gradation for the reviewed characteristics, a radar chart can be constructed for each consistency model. This type of chart allows for a more visual comparison of the models a larger area on the radar chart signifies greater universality and a higher overall score relative to other consistency models. It should be noted that latency is a negative characteristic, so its correspondence between the verbal gradation and the scores is inverse – a model with high latency receives a score of 1. Due to the specifics of the degree of consistency in client-centric models, their comparison should be conducted separately. The radar charts for data-centric models are shown in Figure 3.

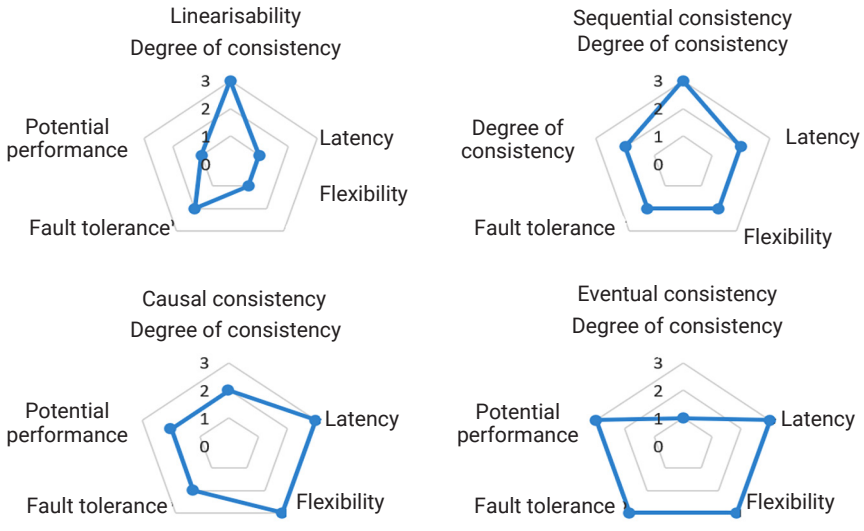

Figure 3. Comparison of data-centric consistency models
Source: developed by the authors

The charts allow to track changes in characteristics when transitioning to progressively weaker data-centric models. Linearisability, sequential consistency, and causal consistency are similar to one another, but each successive model reduces the complexity of the consistency guarantees. This, in turn, reduces implementation complexity and operation execution time, allowing for improvements in the models' latency, potential performance, and flexibility. Under these conditions, eventual consistency, with the weakest guarantees, shows the best potential performance and other characteristics provided that such a degree of consistency satisfies the requirements of the specific distributed information system. The radar charts for client-centric models, shown in Figure 4, display different relationships between characteristics.

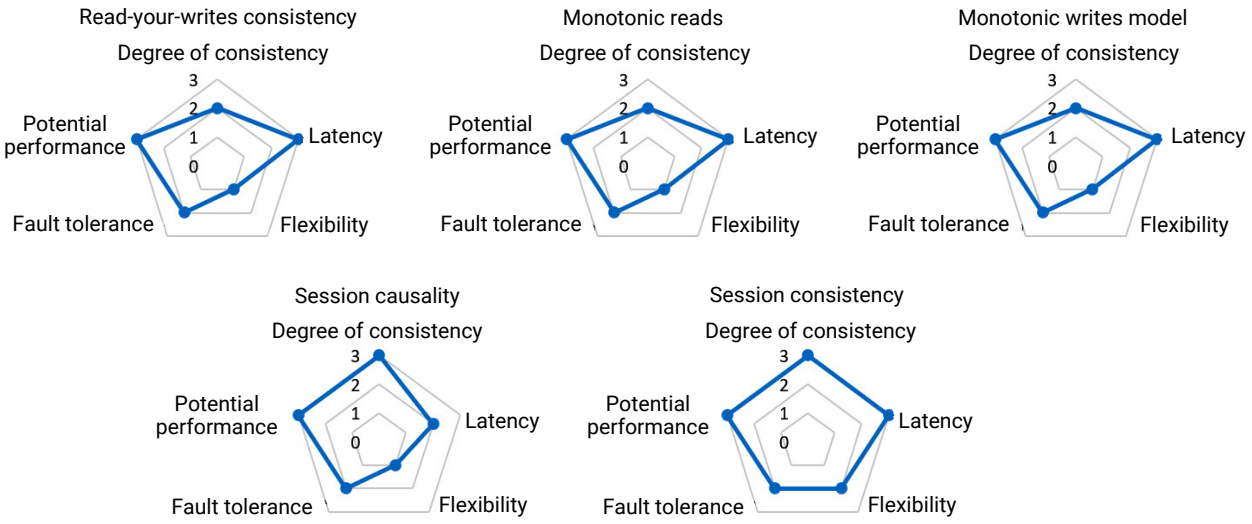

Figure 4. Comparison of data-centric consistency models

Source: developed by the authors

Read-your-writes, monotonic reads, and monotonic writes all exhibit similar metrics due to comparable constraints inherent in their architectural principles. Although they specialise in different application scenarios, their potential performance and flexibility remain similar under the best conditions for each model. The final two models, session causality and session consistency, stand out for having better characteristics due to their hybrid architecture. session causality has a higher degree of consistency due to its kinship with data-centric models, while the superior metrics of session consistency are possible only through the absence of additional constraints in client-centric models, allowing multiple approaches to be applied simultaneously to cover the widest range of application scenarios.

It is worth noting that while the comparison is theoretical, it allows to track a certain trend: older data consistency models with stricter guarantees are less flexible and less performant, whereas the best speed, even at a theoretical level, is offered by models and methods with the minimum degree of global consistency. Thus, maintaining data in a consistent state is a complex task that is directly interconnected with other properties and characteristics of the created distributed system. The researched information and theoretical foundations of existing consistency models can be used to develop a proprietary method for ensuring data consistency that can cover a broader spectrum of application scenarios.

The results obtained are particularly interesting in the context of comparison with the research by E. Brewer (2012), where the author of the original CAP theorem reviews its application in a modern context. The author's analysis determined that the balance between consistency and availability can vary depending on the DDBMS subsystem or the current operation. This suggests that classifying models according to intermediate consistency levels is appropriate, which aligns with the results obtained from the analysis of data-centric models, where the level of guarantees is reduced to speed up operations. However, the author emphasised the need to develop and integrate mechanisms for operability and recovery under network problems, which was only partially addressed in this work during the analysis of client-centric models.

A similar analysis of consistency models was performed by H.N.S. Aldin *et al*. (2019), which considers not only data-centric and client-centric models but also newer hybrid solutions. The authors examined only the general operating principles of the models and their architectural solutions, without the detailed identification and comparison of their characteristics performed in this study. However, their focus on additional DDBMS characteristics is important, such as issues of scalability, security, cost optimisation (which is particularly crucial in the context of using cloud providers), handling stale data, and the energy efficiency of modern distributed systems.

Similar results were obtained in the study by R. Cattell (2011), where the main focus was on comparing the SQL and NoSQL systems that use the consistency models, rather than the models themselves. The author conducted a broad analysis of key-value and document-oriented

DDBMSs, as well as considering the problems of classic relational systems. Based on this, the author predicted the continued proliferation of systems using a lesser degree of consistency to improve other characteristics - a similar trend was found during this research, exemplified by the development of data-centric systems. The analysis by Z. Mahfoud & N. Nouali-Taboudjemat (2019) also considers the classification of consistency models in the context of the CAP and PACELC theorems, but places a greater focus on existing distributed systems from the largest cloud providers, including Amazon Web Services, Google Cloud, and Microsoft Azure. Furthermore, the researchers analyse the possibilities of using multiple consistency models in a single DDBMS depending on the concept, which is represented in some software products, but a more detailed comparison of properties and characteristics, as performed in the current research, is absent.

The work by M. Diogo *et al.* (2019) is based on most of the reviewed consistency models, but with less focus on the data-centric sphere. Moreover, it analyses the practical implementation of these models in existing software products, including hybrid solutions like Cassandra, MongoDB, and Neo4j. This allowed the authors to gain a broader representation of consistency model application scenarios, but the resulting comparison was based solely on the CAP theorem, without a more detailed investigation of behaviour according to PACELC or other systems. An analysis and classification system similar to those used in this research can be found in the work by B. Pradeep (2023). The author investigated the features of the CAP theorem, its drawbacks, and similarly arrived at a classification using PACELC and a relative comparison of consistency models using the characteristics of latency, core operating principles, and application scenarios. A significant difference is the consideration of the coordination complexity of data-centric models when scaling DDBMSs. However, the author did not explore the application and comparison of client-centric consistency models.

The trade-off between the level of data consistency and system speed and availability identified in this research is also tracked in the work by D. Nguyen *et al*. (2019). The authors detail the operating principles and limitations of eventual and sequential consistency models, modelling various implementations of DDBMSs and their specific mechanisms in the context of performance improvement, which provides extensive practical data for further research. However, the analysis and comparison of other models with a weaker degree of consistency, as well as the possibilities of applying client-centric models which are particularly important in the context of key-value DDBMSs on which the work is focused remains unresolved.

Conclusions

In this article, a comprehensive analysis of methods for ensuring data consistency in distributed database management systems was carried out. Consistency serves as a key factor in ensuring the correct operation of such systems in a multi-node environment where both network failures and conflicting write operations are possible. It has been established that consistency mechanisms play a critical role in maintaining the reliability and predictability of data, especially with concurrent access and asynchronous replication.

As a result of the research, the main approaches to classifying consistency models were systematised based on the CAP and PACELC theorems, with an emphasis on practical trade-offs between consistency, availability, latency, and resistance to cluster partitioning. Data-Centric models (linearisability, sequential consistency, causal consistency, eventual consistency) and Client-Centric models (read-your-writes, monotonic reads, monotonic writes, session causality, session consistency) were analysed. Based on the analysis, their advantages, disadvantages, and application scenarios were determined, and a comparison was performed across a range of selected characteristics, such as the degree of consistency, fault tolerance, flexibility, latency, and potential performance. The comparison allowed to establish a relationship between the degree of consistency and performance among the data-centric models: the principles of linearisability and sequential consistency provide better consistency guarantees but suffer from higher latency and lower potential performance. At the same time, client-centric models have a narrower scope of application as they only provide data consistency guarantees within a session or within a single node and a few clients, but in such usage scenarios, they show good performance and performance.

The causal and eventual consistency models can be highlighted as the most promising for further research and development. The causal model shows a good balance between the selected characteristics while maintaining a sufficiently high degree of data consistency, and the eventual model offers potentially the best performance among data-centric models. Among the client-centric models, session consistency is worth noting, as it combines most of the characteristics of the other models in this category and is the most universal and effective. Due to a different synchronisation mechanism, this model is not suitable for resolving conflicts between nodes and cannot be directly compared with data-centric models, but its principles can be used to develop specific methods within dynamic or adaptive data consistency models. Further research should focus on finding ways to improve the performance and fault tolerance of the causal consistency model, enhancing the degree of consistency in the eventual model while preserving other metrics, and researching and developing adaptive and dynamic consistency models.

Acknowledgements

None.

Funding

The study was not funded.

Conflict of Interest

None.

References

[1] Abadi, D. (2012). Consistency tradeoffs in modern distributed database system design: CAP is only part of the story. *Computer*, 45(2), 37-42. doi: 10.1109/mc.2012.33.

[2] Ahmed, J., Karpenko, A., Tarasyuk, O., Gorbenko, A., & Sheikh-Akbari, A. (2023). Consistency issue and related trade-offs in distributed replicated systems and databases: A review. *Radioelectronic and Computer Systems*, 2(106), 171-179. doi: 10.32620/reks.2023.2.14.

[3] Aldin, H.N.S., Deldari, H., Moattar, M.H., & Ghods, M.R. (2019). Consistency models in distributed systems: A survey on definitions, disciplines, challenges and applications. *ArXiv*. doi: 10.48550/arXiv.1902.03305.

[4] Aldin, H.N.S., Deldari, H., Moattar, M.H., & Ghods, M.R. (2020). Strict timed causal consistency as a hybrid consistency model in the cloud environment. *Future Generation Computer Systems*, 105(C), 259-274. doi: 10.1016/j.future.2019.11.038.

[5] Almeida, P.S. (2024). A framework for consistency models in distributed systems. *ArXiv*. doi: 10.48550/arXiv.2411.16355.

[6] Brewer, E. (2012). CAP twelve years later: How the "rules" have changed. *Computer*, 45(2), 23-29. doi: 10.1109/mc.2012.37.

[7] Campêlo, R.A., Casanova, M.A., Guedes, D.O., & Laender, A.H.F. (2020). A brief survey on replica consistency in cloud environments. *Journal of Internet Services and Applications*, 11(1), article number 1. doi: 10.1186/s13174-020-0122-y.

[8] Cattell, R. (2011). Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*, 39(4), 12-27. doi: 10.1145/1978915.1978919.

[9] Chen, Y., Pan, A., Lei, H., Ye, A., Han, S., Tang, Y., Lu, W., Chai, Y., Zhang, F., & Du, X. (2024). TDSQL: Tencent distributed database system. *Proceedings of the VLDB Endowment*, 17(12), 3869-3882. doi: 10.14778/3685800.3685812.

[10] Diogo, M., Cabral, B., & Bernardino, J. (2019). Consistency models of NoSQL databases. *Future Internet*, 11(2), article number 43. doi: 10.3390/fi11020043.

[11] Faria, N., & Pereira, J. (2025). CRDV: Conflict-free replicated data views. *Proceedings of the ACM on Management of Data*, 3(1), 1-27. doi: 10.1145/3709675.

[12] Ghasemirad, S., Sprenger, C., Liu, S., Multazzu, L., & Basin, D. (2025). Pushing the limit: Verified performance-optimal causally-consistent database transactions. In A. Gurfinkel & M. Heule (Eds.), *Tools and algorithms for the construction and analysis of systems. TACAS 2025. Lecture notes in computer science* (Vol. 15698, pp. 43-62). Cham: Springer. doi: 10.1007/978-3-031-90660-2_3.

[13] Golab, W. (2018). Proving PACELC. *SIGACT News*, 49(1), 73-81. doi: 10.1145/3197406.3197420.

[14] Gorbenko, A., Karpenko, A., & Tarasyuk, O. (2020). Analysis of trade-offs in fault-tolerant distributed computing and replicated databases. In *2020 IEEE 11th international conference on dependable systems, services and technologies (DESSERT)* (pp. 1-6). Kyiv: IEEE. doi: 10.1109/DESSERT50317.2020.9125078.

[15] Junfeng, T., Wenqing, B., & Haoyi, J. (2022). PGCE: A distributed storage causal consistency model based on partial geo-replication and cloud-edge collaboration architecture. *Computer Networks*, 212(C), article number 109065. doi: 10.1016/j.comnet.2022.109065.

[16] Lourenço, J.R., Cabral, B., Carreiro, P., Vieira, M., & Bernardino, J. (2015). Choosing the right NoSQL database for the job: A quality attribute evaluation. *Journal of Big Data*, 2(1), article number 18. doi: 10.1186/s40537-015-0025-0.

[17] Mahfoud, Z., & Nouali-Taboudjemat, N. (2019). Consistency in cloud-based database systems. *Informatica*, 43(3), 313-319 . doi: 10.31449/inf.v43i3.2650.

[18] Mahmoud, H.A., & Yasin, H.M. (2025). Data integrity and consistency challenges in distributed database systems. *Engineering and Technology Journal*, 10(5), 5077-5086. doi: 10.47191/etj/v10i05.36.

[19] Muñoz-Escoí, F.D., de Juan-Marín, R., García-Escrivá, J.-R., González de Mendívil, J.R., & Bernabéu-Aubán, J.M. (2019). CAP theorem: Revision of its related consistency models. *The Computer Journal*, 62(6), 943-960. doi: 10.1093/comjnl/bxy142.

[20] Myrhodskyy, A.V., Romanyuk, O.V., Romanyuk, O.N., & Titova, N.V. (2023). Development of a high availability method for configuration management software. *Optoelectronic Information-Power Technologies*, 46(2), 64-75. doi: 10.31649/1681-7893-2023-46-2-64-75.

[21] Nguyen, D., Charapko, A., Kulkarni, S.S., & Demirbas, M. (2019). Using weaker consistency models with monitoring and recovery for improving performance of key-value stores. *Journal of the Brazilian Computer Society*, 25(1), article number 10. doi: 10.1186/s13173-019-0091-9.

[22] Park, S., Kim, J., Mulder, I., Jung, J., Lee, J., Krebbers, R., & Kang, J. (2024). A proof recipe for linearizability in relaxed memory separation logic. *Proceedings of the ACM on Programming Languages*, 8(PLDI), 175-198. doi: 10.1145/3656384.

[23] Perrin, M., Petrolia, M., Mostéfaoui, A., Jard, C. (2016). On composition and implementation of sequential consistency. In C. Gavoille & D. Ilcinkas (Eds.), *Distributed computing. DISC 2016. Lecture notes in computer science* (Vol. 9888, pp 284-297). Berlin: Springer. doi: 10.1007/978-3-662-53426-7_21.

[24] Pradeep, B. (2023). Data consistency models in distributed systems: CAP theorem revisited. *International Journal on Science and Technology*, 14(3). doi: 10.5281/zenodo.14631471.

[25] Rabeshko, Yu. , & Turbal , Yu. (2023). Review of joint text editing algorithms Conflict-free Replicated Data Types (CRDT). *Bulletin of Cherkasy State Technological University*, 28(4), 10-18. doi: 10.62660/2306-4412.4.2023.10-18.

[26] Russel, D., Dawson, R., Chen, N., & Chambers, A. (2025). *Consistency models and verification in modern distributed systems*. Retrieved from https://www.researchgate.net/publication/389598133_Consistency_Models_and_Verification_in_Modern_Distributed_Systems.
[27] Taghinezhad-Niar, A. (2024). A client-centric consistency model for distributed data stores using colored petri nets. In *10th international conference on web research (ICWR)* (pp. 309-314). Tehran: IEEE. doi: 10.1109/ICWR61162.2024.10533365.
[28] Viotti, P., & Vukolić, M. (2016). Consistency in non-transactional distributed storage systems. *ACM Computing Surveys (CSUR)*, 49(1), article number 19. doi: 10.1145/2926965.
[29] Wang, C., Mohror, K., & Snir, M. (2024). Formal definitions and performance comparison of consistency models for parallel file systems. *IEEE Transactions on Parallel and Distributed Systems*, 35, 1092-1106. doi: 10.1109/TPDS.2024.3391058.
[30] Xu, Q., Yang, C., & Zhou, A. (2024). Native distributed databases: Problems, challenges and opportunities. *Proceedings of the VLDB Endowment*, 17(12), 4217-4220. doi: 10.14778/3685800.3685839.

Порівняння моделей забезпечення узгодженості даних у розподілених системах керування базами даних

Андрій Миргородський

Аспірант
Вінницький національний технічний університет
21021, вул. Хмельницьке шосе, 95, м. Вінниця, Україна
https://orcid.org/0009-0007-6764-0060

Оксана Романюк

Кандидат технічних наук, доцент
Вінницький національний технічний університет
21021, вул. Хмельницьке шосе, 95, м. Вінниця, Україна
https://orcid.org/0000-0003-0235-8615

Анотація. Використання розподіленої інфраструктури для забезпечення масштабованості та високої доступності створює нові виклики для підтримки узгодженості даних між вузлами інформаційних систем, що стрімко зростають та вимагають надійного управління даними для коректної роботи. Метою дослідження була комплексна систематизація та порівняльний аналіз методів забезпечення узгодженості даних у розподілених системах керування базами даних, враховуючи фундаментальні компроміси між узгодженістю, доступністю та затримками оновлення, що описуються теоремами CAP та PACELC. Для досягнення мети було використано методи теоретичного аналізу, формального моделювання поведінки систем та порівняльного експертного оцінювання. В результаті дослідження було систематизовано моделі узгодженості за двома основними підходами: інформаційно-орієнтованим та клієнтоорієнтованим. В рамках першого підходу проаналізовано моделі, що визначають глобальну поведінку системи: лінеаризовність, послідовну, причинну та кінцеву узгодженість. Для кожної моделі визначено переваги, недоліки та типові сценарії застосування. У рамках другого підходу розглянуто клієнтоорієнтовані моделі, що надають гарантії в межах сесії одного користувача: узгодженість читання і запису, монотонне зчитування, монотонний запис та сесійну причинність. Запропоновано узагальнену класифікацію, яка візуалізує співвідношення між ступенем узгодженості, затримками, гнучкістю, стійкістю до збоїв та потенційною продуктивністю для кожної моделі. Проведено порівняння усіх розглянутих моделей узгодженості даних за допомогою ряду відібраних істотних характеристик (клас за PACELC, узгодженість, стійкість до збоїв, потенційна продуктивність тощо) та діаграм на основі їх параметрів. Практична цінність роботи полягає у формулюванні чітких рекомендацій щодо вибору оптимальної моделі узгодженості залежно від вимог до надійності, продуктивності та архітектурних особливостей інформаційної системи. Результати можуть бути використані для підвищення ефективності проектування розподілених баз даних у високонавантажених системах, таких як фінансові сервіси, платформи Інтернету речей та хмарні застосунки

Ключові слова: розподілені системи; інформаційно-орієнтована узгодженість; клієнтоорієнтована узгодженість; CAP-теорема; PACELC; системна доступність; відмовостійкість