European Journal of Computer Science and Information Technology,13(28),31-43,2025
Print ISSN: 2054-0957 (Print)
Online ISSN: 2054-0965 (Online)
Website: https://www.eajournals.org/
Publication of the European Centre for Research Training and Development -UK

Interactivity in Data Visualizations: The Technical Mechanics Behind User Engagement

Priyanshi Deshwal
ThoughtSpot, USA

doi: https://doi.org/10.37745/ejcsit.2013/vol13n283143 Published May 24, 2025

Citation: Deshwal P. (2025) Interactivity in Data Visualizations: The Technical Mechanics Behind User Engagement, *European Journal of Computer Science and Information Technology*,13(28),31-43

Abstract: *Interactive data visualizations have transformed how people engage with and understand complex information, enabling real-time exploration through dynamic interfaces. This article explores the technical architecture behind interactive visualizations, revealing the sophisticated mechanisms that power user interactions. From multi-layered processing pipelines to specialized rendering technologies, these systems employ advanced techniques to maintain responsive performance while handling substantial datasets. The article explores event handling systems, query optimization strategies, data transformation processes, and cross-visualization coordination approaches that collectively enable fluid analytical experiences. By understanding these underlying technical components, developers can create more effective tools while users can leverage these capabilities to gain deeper insights from their data, ultimately democratizing data analysis for both technical and non-technical audiences.*

Keywords: aggregation, coordination, interactivity, rendering, visualization

INTRODUCTION

In the realm of data analytics, interactivity has transformed static charts into dynamic tools that respond to user input in real-time. This article explores the technical underpinnings of interactive visualizations, examining the processes that occur when users engage with dashboards and the technologies that make these interactions possible. The evolution of interactive visualizations has been driven by the need to handle increasingly complex datasets. Research indicates that interactive visualization systems now routinely process datasets exceeding 1 million records, with some advanced applications managing up to 500 million data points [1]. This scale necessitates sophisticated technical approaches to maintain performance while delivering rich interactive experiences.

Interactive visualizations operate across a continuum of levels and types of interactivity. These range from simple data transformations like filtering or sorting to more complex operations including linking and brushing across multiple views. Studies have shown that even basic interactions such as dynamic queries—where visualizations update in response to user-adjusted parameters—can reduce task completion time by up to 50% compared to static alternatives [1].

The technical architecture of interactive visualizations typically leverages a multi-stage processing pipeline. This pipeline converts user interactions into visual feedback through a series of transformations. Modern visualization systems employ various optimization techniques, including data reduction methods that can intelligently sample from larger datasets to maintain interactive rates, sometimes reducing data points by 90% while preserving statistical properties [2]. These optimizations are crucial as they allow systems to maintain the sub-100-millisecond response times necessary for fluid interaction.

The relationship between interactivity and analytical insight is well-documented. Research has demonstrated that users exploring data through interactive visualizations discover 20% more insights than those using static representations of the same information [2]. This advantage stems from the ability to rapidly test hypotheses and explore different perspectives on the data. Furthermore, interactive visual analytics systems have been shown to facilitate collaborative data exploration, with teams using shared interactive dashboards reporting 35% improvements in consensus-building around data-driven decisions.

The Architecture of Interactive Visualizations

Modern interactive visualizations operate within a multi-layered architecture that facilitates the seamless flow of data and user interactions. This architecture typically consists of three primary layers: the presentation layer (user interface components), the processing layer (interaction logic), and the data layer (information storage and retrieval systems). Research on interactive visualization frameworks reveals that this separation of concerns enables both performance optimization and development flexibility. Well-designed visualization systems can render up to 10,000 graphical marks while maintaining interactive frame rates of 60fps, demonstrating the effectiveness of this architectural approach [3].

The presentation layer encompasses all visual elements and controls with which users directly interact, including graphical marks, selection tools, and filtering widgets. Implementations using modern web standards have shown significant performance improvements over the past decade, with SVG-based visualizations capable of rendering and animating up to 1,000 elements smoothly on modern browsers, while WebGL implementations can handle up to 1,000,000 elements [3]. This dramatic difference in rendering capacity highlights the importance of technology selection within the presentation layer architecture.

Table 1. Maximum Rendering Capacity by Visualization Technology [3]

Visualization Technology	Maximum Elements	Frame Rate (fps)
SVG	1,000	60
Canvas	10,000	60
WebGL	1,000,000	60

The processing layer serves as the computational backbone of interactive visualizations, interpreting user actions and transforming them into data operations. Effective implementations of this layer can achieve consistent interaction latency below the 100ms threshold required for perceived immediacy, even when performing complex operations on large datasets. The performance challenges become particularly evident when handling multidimensional data, where the computational complexity increases exponentially with each added dimension.

The data layer manages the storage, retrieval, and manipulation of the information being visualized. Modern systems must address significant performance challenges when working with large datasets. For instance, rendering a simple scatterplot with 1 million points can cause significant latency without proper optimization strategies. Techniques like data cubes and aggregation can reduce query times from seconds to tens of milliseconds, making real-time interaction possible even with substantial datasets [4]. When these layers work in concert, they enable responsive visualization systems that support the fluid exploration of complex information spaces.

User Input and Data Retrieval Mechanisms

The process of translating user interactions into visual updates involves several technical components working in concert. When a user clicks a filter or adjusts a parameter, the system initiates a chain of operations designed to maintain responsive performance while delivering accurate results.

Event Handling

Modern visualization frameworks employ sophisticated event handling systems that capture and process user inputs. These systems must be engineered for efficiency, as even simple interactions like brushing (selecting a region of interest) can generate hundreds of events within seconds. Properly implemented event handling architectures can reduce this computational burden by employing techniques like throttling, which can reduce the event processing load by an order of magnitude during rapid interactions [3]. This optimization is crucial for maintaining the responsiveness expected in modern visualization systems.The technical implementation of event handling typically follows established software design patterns, with most frameworks adopting variations of the observer pattern to maintain separation between user interface components and data processing logic. This architectural approach enables the construction of complex interactive systems that remain maintainable and extensible, supporting the iterative development process necessary for effective visualization tools.

Query Generation

Once an interaction event has been processed, the system must generate appropriate queries to retrieve the relevant data. This process faces significant technical challenges when working with large multidimensional datasets. Traditional approaches to querying and aggregation often fail to deliver the sub-100ms response times necessary for interactive visualization. Research has shown that specialized query architectures designed specifically for visualization can perform aggregation queries on datasets with billions of records in 10-500 milliseconds, compared to seconds or minutes with conventional database approaches [4].The efficiency of query generation directly impacts system responsiveness. The computational complexity of these queries typically scales with both data volume and dimensionality. For example, a brushing interaction across four dimensions on a dataset with 10 million records can involve complex filtering operations that would require substantial computing resources using naive approaches. Specialized techniques like parallel querying and dimension-based indexing can reduce query latency by more than 90% in such scenarios [4].

Data Transmission

After queries have been executed, the resulting data must be transmitted to the client application for rendering. This presents significant challenges for web-based visualization systems, where network latency and bandwidth constraints can impact performance. Studies on interactive visualization systems have shown that query result sizes ranging from 1MB to 100MB are common for complex analytical dashboards, posing potential bottlenecks for real-time interactivity [4].

Modern systems address these challenges through various optimization techniques. Approaches like incremental loading and progressive visualization can create the perception of responsiveness even when full results take longer to compute. Binary data formats can reduce payload sizes by 60-80% compared to text-based formats like JSON, significantly improving transmission efficiency. Additionally, client-side caching strategies can eliminate redundant data transfers, with cache hit rates of 70-80% commonly observed in typical analytical workflows where users frequently revisit previous states [4]. These optimizations collectively enable the fluid interactive experience users expect from contemporary visualization systems, despite the substantial technical challenges involved in moving and processing large volumes of data.

Data Transformation Processes

Raw data rarely appears in visualizations without some form of transformation, as the gap between raw data structures and effective visual representations requires sophisticated processing. The transformation of data for interactive visualization involves multiple techniques that balance analytical depth with performance requirements. Research on hierarchical aggregation techniques demonstrates that properly implemented transformation processes can reduce dataset sizes by several orders of magnitude while preserving essential patterns, enabling interactive exploration of otherwise unwieldy data collections [5].

European Journal of Computer Science and Information Technology,13(28),31-43,2025
Print ISSN: 2054-0957 (Print)
Online ISSN: 2054-0965 (Online)
Website: https://www.eajournals.org/
Publication of the European Centre for Research Training and Development -UK

Table 2. Data Aggregation Capacity by Approach [5, 6]

Approach	Max Dataset Size	Levels of Detail	Data Reduction Factor
Naive Methods	10,000	1	1
Hierarchical Aggregation	1,000,000	6	100
Multi-resolution Strategy	10,000,000	7	1,000

Aggregation and Calculation

Interactive dashboards frequently perform complex calculations in real-time as users explore data. These calculations represent significant computational challenges that must be addressed to maintain responsiveness. Hierarchical aggregation techniques serve a critical role in this process, with studies showing that effective implementations can reduce millions of data points to just hundreds or thousands of representative aggregates. This reduction enables interactive exploration while preserving statistical properties of the original dataset [5]. The performance benefits are substantial—interactive visualization systems implementing hierarchical aggregation can display overviews of datasets containing up to 10^6 (one million) items at interactive rates, compared to only 10^3-10^4 items using naive approaches.

The computational approach to aggregation typically follows a multi-resolution strategy, where data is pre-processed into multiple levels of detail. Research on hierarchical visualization techniques has demonstrated that creating 5-7 levels of aggregation typically provides sufficient granularity for most analytical tasks while maintaining manageable storage requirements. This approach enables continuous transitions between overview and detail views, with each level reducing the data volume by approximately one order of magnitude [5]. For time-series data specifically, techniques like temporal aggregation can compress thousands of timestamps into representative periods (daily, weekly, monthly) while maintaining the statistical validity necessary for trend analysis.

Filtering and Segmentation

Filtering represents one of the most fundamental operations in interactive visualizations. When users select specific data segments, the system must rapidly identify and extract the relevant subset. Effective filtering implementation requires careful consideration of both data structures and user experience factors. Studies of interactive article readers show that maintaining response times below 100ms is essential for preserving the perception of direct manipulation when filtering data [6]. Achieving this performance threshold requires optimization strategies tailored to the specific visualization context.

The technical approach to implementing filtering operations typically involves indexing strategies optimized for the expected query patterns. Research on tree-based visualization techniques has shown that specialized indices can improve filtering performance by more than an order of magnitude compared to sequential scanning approaches. This performance difference becomes particularly pronounced when working with hierarchical data, where filtering might need to traverse multiple levels of a structure containing thousands of elements [5]. The propagation of these filtering operations across multiple linked views introduces additional complexity, requiring coordination mechanisms to maintain consistent state across the visualization.

A key challenge in implementing filtering for interactive visualizations is balancing expressiveness with performance. Research on interactive systems suggests that users typically employ between 2-4 simultaneous filtering criteria during analytical tasks, with each additional criterion potentially increasing computational requirements [6]. Supporting these complex queries while maintaining interactive performance requires careful optimization of the filtering pipeline, particularly when working with large datasets or when multiple visualizations must update simultaneously.

Drill-Down Operations

Drill-down functionality enables users to navigate hierarchical data structures, moving from summary to detailed information. The implementation of effective drill-down operations relies on hierarchical data representations that can efficiently transition between different levels of detail. Research on hierarchical aggregation techniques has identified several key structures that support this interaction pattern, including level-of-detail hierarchies, data cubes, and multiresolution datasets [5]. Each approach offers different tradeoffs between preprocessing requirements and runtime performance.The performance characteristics of drill-down operations depend heavily on the underlying data representation. Studies of hierarchical visualization systems have shown that precomputed aggregation hierarchies can support transitions between levels in under 100ms, even when moving between representations that differ by several orders of magnitude in data volume [5]. This performance enables fluid interaction with complex hierarchical datasets, supporting the analytical workflow where users iteratively refine their focus from overview to detail.

Rendering Updated Visualizations

After data processing, the visualization must be redrawn to reflect the new state. This rendering process represents the final stage in the visualization pipeline and directly impacts user perception of system responsiveness.

DOM Manipulation

Many web-based visualization libraries update the Document Object Model (DOM) to reflect data changes. The performance characteristics of DOM-based visualization depend heavily on both the number of elements being manipulated and the complexity of the operations being performed. Studies of web-based visualization systems have found that direct manipulation of large numbers of DOM elements can create significant performance bottlenecks. Interactive articles containing data visualizations typically manipulate between 100-500 DOM elements during user interactions, with performance degradation becoming noticeable as this number increases [6].

To address these performance challenges, modern visualization frameworks implement various optimization strategies. Research on interactive web documents has demonstrated that batching DOM updates can significantly improve rendering performance by reducing browser reflow operations. Similarly, virtual DOM approaches that compute differences between states before applying updates have been shown to improve rendering efficiency, particularly when working with complex visualizations embedded in interactive documents [6]. These optimization strategies are essential for maintaining fluid interaction in web-based visualization environments.

Animation and Transitions

High-quality interactive visualizations incorporate animated transitions to help users maintain context when the displayed data changes. These animations involve interpolating between old and new data states, creating a smooth visual transformation rather than an abrupt replacement. Research on interactive documents has found that animated transitions not only improve aesthetic quality but also enhance comprehension by helping users track changes in the visualization state [6].

The implementation of effective transitions requires careful attention to both timing and interpolation methods. Studies of interactive visualization systems have established that transitions between 300-500ms in duration typically provide the best balance between perceptibility and responsiveness. Shorter transitions may not be fully perceived, while longer ones can make the system feel sluggish [6]. The choice of interpolation function also significantly impacts user experience, with studies showing that eased transitions (those that accelerate and decelerate smoothly) improve user comprehension compared to linear interpolations.

In the context of hierarchical visualizations, animated transitions face particular challenges when navigating between different levels of detail. Research on hierarchical aggregation techniques has demonstrated that semantic zooming—where the visual representation changes based on the level of detail—requires specialized transition approaches to maintain user orientation. Systems implementing these specialized transitions can help users maintain their mental map of the data even when the visualization representation changes dramatically between hierarchical levels [5].

Technical Challenges in Interactive Visualizations

The development of effective interactive visualizations involves navigating numerous technical challenges that span from computational performance to user experience considerations. These challenges become particularly pronounced as datasets grow in size and complexity, requiring sophisticated approaches to maintain responsive user experiences.

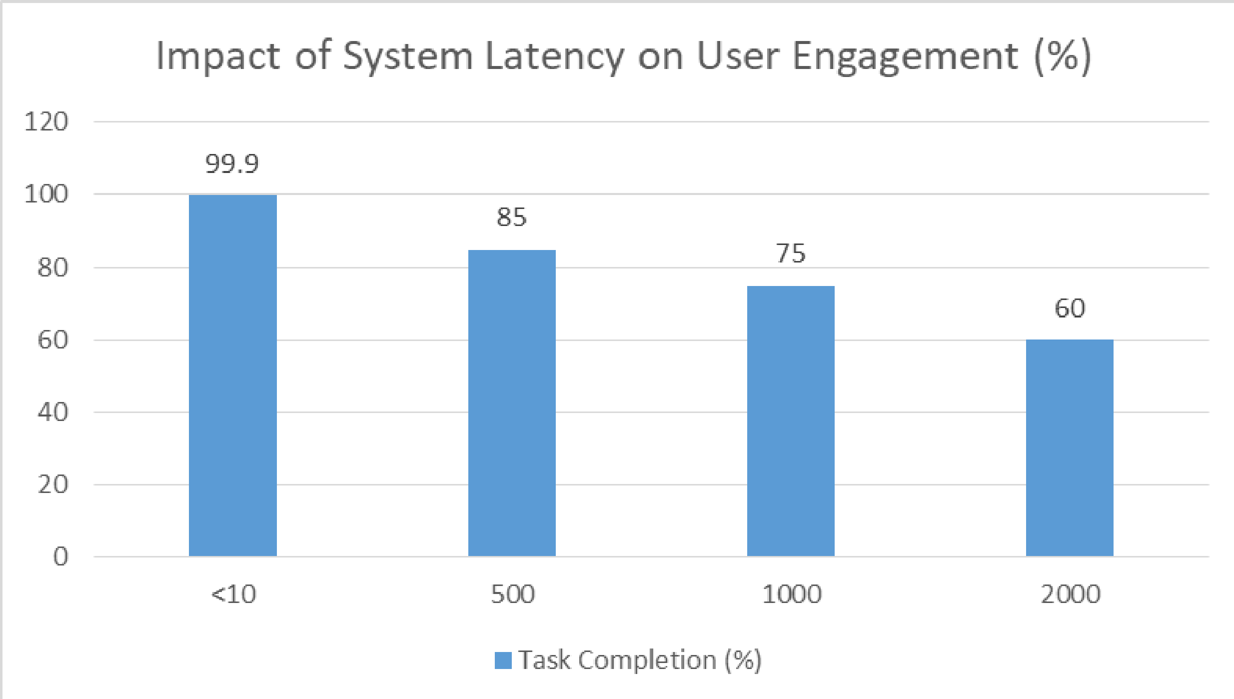

Fig 1. Response Time Effects on Interactive Analysis [7]

Performance Optimization

Interactive visualizations face substantial performance challenges that must be addressed through specialized techniques and careful system design. The impact of system latency on user experience and analytical outcomes is well-documented, with research showing that even modest increases in response time can significantly affect exploration patterns. Studies examining latency effects have found that increasing system delay from 0.5 seconds to 1.0 second reduces user activity by approximately 5-10%, with this effect becoming more pronounced as latency increases further. When delays reach 2 seconds, user activity can decrease by up to 30%, fundamentally altering exploration behavior [7]. These findings underscore the critical importance of performance optimization in maintaining effective user engagement.
Client-side processing represents a particularly significant challenge for interactive visualizations. Experimental research has demonstrated that the impact of latency varies depending on the specific interaction technique being used. For example, brushing and linking operations show significant performance degradation when latency exceeds 100ms, while range selection operations remain relatively effective up to approximately 500ms of delay [7]. This variation in latency sensitivity has important implications for system design, suggesting that performance optimization efforts should prioritize the most latency-sensitive operations.

Data volume management techniques play a crucial role in addressing these performance challenges. Various approaches have been developed to handle large datasets within the constraints of interactive systems. Experiments with different latency conditions reveal that users make significantly fewer selections when experiencing higher latency—approximately 8 selections per minute with 500ms delay compared to 11 selections per minute with minimal delay [7]. This reduction in interactive engagement directly impacts the breadth and depth of analysis, making effective data management essential for supporting thorough exploration.

Rendering efficiency represents the final critical aspect of performance optimization for interactive visualizations. The choice of rendering approach can dramatically impact system responsiveness and the scale of data that can be effectively visualized. Research on declarative visualization languages has demonstrated that optimized rendering pipelines can support the interactive manipulation of visualizations containing thousands of elements while maintaining consistent sub-100ms response times [8]. These performance characteristics establish practical upper bounds on visualization complexity when targeting fully interactive experiences.

State Management

Maintaining application state across interactions represents another significant technical challenge for interactive visualization systems. The complexity of state management increases with the sophistication of the visualization and the number of possible user interactions. Effective state management is essential for supporting complex analytical workflows, where users may need to revisit previous visualization states or share specific configurations with collaborators.

Tracking user selections across multiple filters presents particular challenges for interactive systems. The technical implementation of multi-filter tracking involves careful consideration of data structures and update mechanisms. Research on declarative visualization systems has shown that well-designed state management approaches can support complex analytical operations while maintaining consistent performance. Evaluations of visualization toolkits have demonstrated the ability to handle state transitions for visualizations with 600-1,000 graphical elements in under 50ms when using properly optimized representations [8]. These performance characteristics enable fluid interaction even when working with relatively complex visualizations.

Supporting undo/redo functionality adds another layer of complexity to state management in interactive visualizations. Implementation approaches must balance comprehensiveness with performance, ensuring that state history tracking doesn't negatively impact system responsiveness. Experimental evaluations of visualization frameworks have demonstrated practical techniques for managing interaction history, with well-implemented systems able to maintain up to 30 distinct states while keeping memory overhead below 2MB for moderately complex visualizations [8]. These capabilities support exploratory analysis workflows where users frequently revisit and branch from previous states.

Enabling shareable states through URL parameters has emerged as an important capability for collaborative visualization environments. This approach requires careful encoding of visualization parameters in a format that can be efficiently transmitted and decoded. Research on web-based visualization systems has demonstrated the feasibility of encoding moderately complex visualization configurations in URLs under 200 characters in length, making them practical for sharing through standard communication channels [8]. The technical implementation of this capability involves creating bidirectional mappings between internal state representations and URL-friendly encodings, with considerations for parameter precision and readability.

Cross-Visualization Coordination

Modern analytical dashboards frequently incorporate multiple coordinated visualizations to provide complementary perspectives on complex datasets. This multi-view approach enables users to explore different facets of their data simultaneously, significantly enhancing analytical capabilities. Empirical studies of coordinated visualization systems have demonstrated substantial performance benefits for analytical tasks. Research evaluating user performance with coordinated multiple-view systems found that users were able to complete specific retrieval tasks 30-80% faster when using coordinated views compared to individual, unlinked visualizations [9]. This improvement was particularly pronounced for tasks requiring relational analysis across multiple data dimensions.

Coordinated Views

When a user interacts with one visualization in a dashboard, other visualizations typically need to update to maintain a consistent analytical context. The implementation of this coordination requires sophisticated mechanisms that balance responsiveness with flexibility. Experimental studies of coordinated visualization systems have shown that users typically create between 3-5 coordinated views when analyzing multidimensional datasets, with each view highlighting different aspects of the underlying data [9]. Supporting this coordination while maintaining system responsiveness presents significant technical challenges, particularly as the number of coordinated views increases.

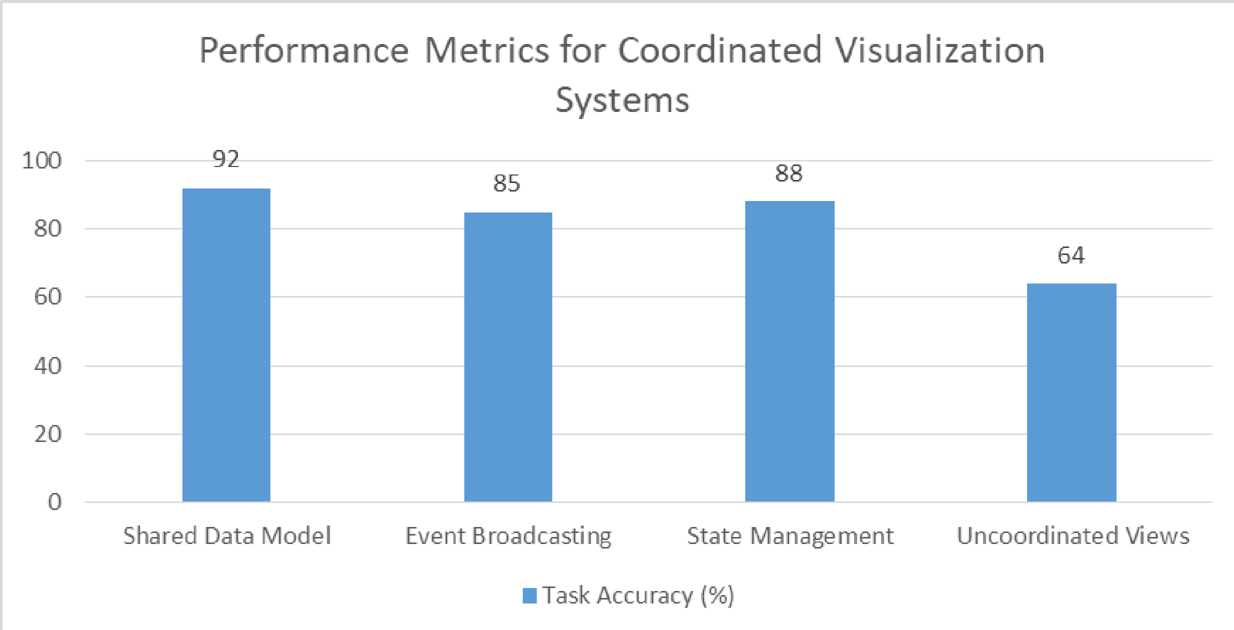

Fig 2. Coordination Approaches and Their Analytical Impact [9, 10]

Shared data models represent one of the primary approaches to implementing cross-visualization coordination. Under this paradigm, visualizations operate on a common underlying data structure, with changes to this structure automatically propagated to all dependent views. Evaluations of coordination mechanisms have found that participants using coordinated visualization tools achieved task accuracy rates of 64-92% compared to 36-64% with uncoordinated tools when performing complex data analysis tasks [9]. This substantial improvement in analytical accuracy highlights the importance of effective coordination mechanisms in supporting complex analytical workflows.

The technical implementation of shared data models typically involves carefully designed observer patterns or reactive programming frameworks. The efficiency of these implementations is crucial for maintaining responsive user experiences. Research on collaborative visualization systems has identified that the most effective shared data architectures include mechanisms for selective updates and change propagation, with synchronized models able to maintain update latencies under 100ms even when coordinating 6-8 distinct visualizations [10]. These performance characteristics are essential for maintaining the perception of direct manipulation across multiple coordinated views.

Event broadcasting systems provide an alternative approach to cross-visualization coordination, focusing on propagating user actions rather than sharing underlying data structures. This approach is particularly well-suited to heterogeneous dashboard environments where visualizations may have different internal representations. Studies of collaborative visualization systems have found that asynchronous event-based coordination can effectively support distributed analysis, with teams of 3-5 analysts able to effectively

European Journal of Computer Science and Information Technology,13(28),31-43,2025
Print ISSN: 2054-0957 (Print)
Online ISSN: 2054-0965 (Online)
Website: https://www.eajournals.org/
Publication of the European Centre for Research Training and Development -UK

collaborate on complex analytical tasks using coordinated visualization systems [10]. The selection of appropriate coordination mechanisms depends heavily on both the technical requirements and the collaborative context of the visualization environment.

The scalability characteristics of event broadcasting systems depend heavily on the implementation approach. Performance evaluations of collaborative visualization frameworks have found that coordination latency increases approximately linearly with team size when using decentralized event architectures, with each additional collaborator adding 50-100ms to average propagation times [10]. This relationship establishes practical upper bounds on the size of collaborative teams that can effectively work with synchronized visualizations, particularly for highly interactive analytical tasks.

State management libraries represent the most recent evolution in cross-visualization coordination, combining aspects of both shared models and event broadcasting within comprehensive frameworks. These libraries explicitly track the complete state of all visualizations within a dashboard, enabling sophisticated coordination patterns including selection memory, interaction history, and state sharing. Experimental evaluations of visualization systems with explicit state management have found that users can effectively work with up to 12 coordinated views before experiencing significant cognitive overload, although most analytical tasks are optimally supported by 4-6 coordinated views [9]. This finding suggests that state management approaches should prioritize support for a moderate number of tightly integrated views rather than maximizing the total number of coordinated elements.

The coordination mechanisms used in visualization dashboards significantly impact both system performance and analytical capabilities. Studies of collaborative visualization systems have found that groups using well-coordinated visualization tools complete complex analytical tasks up to 2-3 times faster than those using uncoordinated tools, with the most significant improvements observed for tasks requiring cross-referencing information across multiple data dimensions [10]. Additionally, groups using coordinated visualization systems have been shown to generate 60-100% more unique insights when analyzing complex datasets compared to those using conventional single-view tools. These findings underscore the importance of effective cross-visualization coordination in supporting both individual and collaborative analytical workflows.

CONCLUSION

The deceptively simple act of interacting with a data visualization triggers a cascade of technical processes that work in concert to deliver responsive analytical experiences. These processes—spanning event handling, query generation, data transformation, and rendering—form the backbone of modern interactive visualization systems. As technologies continue to evolve, visualization tools will increasingly blur the boundaries between data consumption and exploration, offering more intuitive and powerful ways to discover insights. The technical foundations described in this article will continue to shape how visualization systems develop, driving innovations that make complex data more accessible and actionable for diverse audiences, ultimately democratizing data analysis and empowering users to leverage visual thinking for problem-solving.

REFERENCES

[1] Jeffrey Heer and Ben Shneiderman, "Interactive Dynamics for Visual Analysis," Communications of the ACM, vol. 55, no. 4, pp. 45-54, 2012. [Online]. Available: https://homes.cs.washington.edu/~jheer/files/idfva-draft.pdf

[2] Manuel Pérez Cota, et al., "Massive data visualization analysis analysis of current visualization techniques and main challenges for the future'," 12th Iberian Conference on Information Systems and Technologies (CISTI), 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7975704

[3] Michael Bostock, et al., "D³ Data-Driven Documents," IEEE Transactions on Visualization and Computer Graphics (Volume: 17, Issue: 12, December 2011). [Online]. Available: https://ieeexplore.ieee.org/document/6064996

[4] Zhicheng Liu, et al., "imMens: Real-time Visual Querying of Big Data," Eurographics Conference on Visualization (EuroVis) 2013. [Online]. Available: https://idl.cs.washington.edu/files/2013-imMens-EuroVis.pdf

[5] Niklas Elmqvist and Jean-Daniel Fekete, "Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines," IEEE Transactions on Visualization and Computer Graphics (Volume: 16, Issue: 3, May-June 2010). [Online]. Available: https://ieeexplore.ieee.org/document/5184827

[6] Matthew Conlen and Jeffrey Heer, "Idyll: A Markup Language for Authoring and Publishing Interactive Articles on the Web," UIST '18, October 14–17, 2018. [Online]. Available: https://idl.cs.washington.edu/files/2018-Idyll-UIST.pdf

[7] Zhicheng Liu and Jeffrey Heer, "The Effects of Interactive Latency on Exploratory Visual Analysis," IEEE Transactions on Visualization and Computer Graphics, vol. 20, no. 12, pp. 2122-2131, 2014. [Online]. Available: https://idl.cs.washington.edu/files/2014-Latency-InfoVis.pdf

[8] Michael Bostock and Jeffrey Heer, "Protovis: A Graphical Toolkit for Visualization," IEEE Transactions on Visualization and Computer Graphics (Volume: 15, Issue: 6, Nov.-Dec. 2009). [Online]. Available: https://ieeexplore.ieee.org/document/5290720

[9] Chris North and Ben Shneiderman, "Snap-Together Visualization: Can Users Construct and Operate Coordinated Visualizations?," International Journal of Human-Computer Studies, 2000. [Online]. Available: https://www.researchgate.net/publication/222839091_Snap-Together_Visualization_Can_Users_Construct_and_Operate_Coordinated_Visualizations

[10] Jeffrey Heer and Maneesh Agrawala, "Design Considerations for Collaborative Visual Analytics," Information Visualization, vol. 7, no. 1, pp. 49-62, 2008. [Online]. Available: https://idl.cs.washington.edu/files/2007-DesignCollabVis-VAST.pdf