International Journal of Computer Trends and Technology
ISSN: 2231–2803 / https://doi.org/10.14445/22312803/IJCTT-V72I10P113

Volume 72 Issue 10, 79-85, October 2024

Research Paper

Multi-Screen Customer Relationship Management (CRM) Systems: Analysis of Considerations and Complexity

Tanmaya Gaur[1], Krishna Chandra[2]

Bachelor of Engineering (Electronics and Telecommunication), Birla Institute of Applied Sciences
Masters in computer science, Harrisburg University of Science & Technology

[1]*Corresponding Author : tanmay.gaur@gmail.com*

Received: 29 August 2024 Revised: 02 October 2024 Accepted: 17 October 2024 Published: 30 October 2024

Abstract *- Customer Relationship Management (CRM) systems are essential for enterprises seeking to improve customer satisfaction and optimize business operations. These solutions provide technological capabilities that empower employees to assist customers who reach out to the organization. Modern CRMs often need to handle complex products or services, requiring multi-step processes and the ability to display or analyze large sets of data and visuals. This paper explores the potential of using multiple screens side by side to manage these intricate CRM scenarios.*

Keywords *- CRM, Web applications, Multi-screen, Productivity, User experience.*

1. Introduction

Traditional CRM applications are typically web-based, displaying enterprise data on a single screen. Employees use this data to handle customer queries that span various functions such as sales, servicing, analysis, and troubleshooting. Although single-monitor setups are common, there are strong arguments for companies to adopt CRM solutions that function across multiple monitors. These arguments mainly arise from the increasing complexity of modern products and CRMs, where multi-screen experiences can potentially enhance productivity. However, merely offering a web application that operates across two monitors does not automatically enhance productivity. Without a well-thought-out user experience strategy that tackles the right issues, this approach could result in confusion and cognitive overload. This, in turn, may have adverse impacts like increased contact resolution times and negative impacts on end Customer Satisfaction (CSAT).

There are also several application considerations to ensure a cohesive and seamless experience. Key aspects include session management, state preservation, and real-time interactivity across multiple screens. Additionally, it is essential to consider performance, security, and cross-device compatibility. While numerous studies and research papers that focus on dual screens exist, where the completed studies suggest that using multiple monitors enhances productivity, most of these studies have a slight bias, as monitor hardware companies often conduct them. This manuscript will investigate the application considerations required to deliver an effective CRM solution across two monitors. Although the examples will primarily focus on web applications, the same principles can be applied to native applications.

2. Understanding CRMs and the Importance of Multiple Screens

CRM systems are important for modern businesses for several reasons. Let us review what benefits a CRM provides and why multi-screen CRMs can be useful.

2.1. Why Do Organizations need a CRM

Some common CRM benefits are listed below.

- Enhanced Customer Insights: CRMs collate several customer data points from multiple data sources and touchpoints, which provide valuable insights into customer behavior, preferences, and purchase history. This holistic view allows businesses to customize their marketing efforts and enhance customer engagement to be more relevant.
- Enhanced Customer Service: By offering a 360 view of customer interactions, CRMs empower organizations to offer personalized and efficient customer service. Customer service representatives use these CRMs to access comprehensive customer history, enabling them to resolve and understand customer concerns more effectively and build stronger customer relationships.

- Streamlined Operations: CRM systems or complementary software are often used to automate routines around user tasks. This could include steps like data entry, follow-up reminders, and email communications, minimizing manual effort and operational errors. This automation boosts productivity and enables employees to concentrate on strategic activities and customer conversation.
- Increased Sales and Revenue: CRMs are crucial in boosting conversion rates and revenue by identifying prospect opportunities and helping manage sales pipelines. They enable sales teams to track leads, monitor progress, and close deals, thereby driving overall business growth efficiently.

2.2. Advantages of Multi-Screen CRM

Considering these CRM use-cases, multi-screen CRMs can offer significant advantages to employees who need to assist customers through intricate workflows. Some potential benefits of multi-screen experiences include enhanced productivity, improved task management, and a more streamlined customer service process.

- Productivity: With the use of multiple screens, employees can handle several application interactions at the same time, enhancing both response times and service quality. This setup allows them to compare data, cross-reference information, and access various CRM modules seamlessly without switching between tabs.
- User Experience flexibility: Multi-screen setups provide a more holistic view of customer data and interactions, facilitating more informed decision-making. This allows employees to better visualize customer journeys, identify trends, and proactively address issues.
- Improved Workflow: Isolated yet interconnected CRM instances can enable independent navigation and efficient task management. This setup allows employees to handle various aspects of customer relationships simultaneously, boosting efficiency and collaboration.

2.3. Multi-Screen CRM scenarios

Dual screens can be useful for multitasking and managing complex tasks. However, they can also hinder service if agents only need to look at one screen at a time. This is why it is important to understand the scenarios and use-cases where multi-screen CRMs can help.

- When a large amount of data needs to be displayed and assessed side by side, such as examining logs and alerts to understand a customer's concern.
- When displaying two different views of the same data can help the experience. Users can simultaneously see the same information in two different ways, like a list of restaurants on one screen and a map on the other.
- Instances where an employee has two instances of an application open: one for a transaction in progress, and the other may be used for looking up relevant data. If supported by the web application, this can help maintain the state of the transaction window, which the employee may not want to lose, such as data entered on a large form.
- In complex enterprise use-cases, a well-designed multi-screen strategy can reduce cognitive load. Employees often need to use multiple applications to solve customer concerns. For certain call types, agents might need multiple windows open to assist a customer and switch between them constantly. It may be better for them to launch these experiences side by side.

3. Current Multi-Screen Experience Technology and Trends

Let us explore scenarios and experiences where multi-screen setups can be beneficial. The following section will discuss some trending multi-screen experiences.

3.1. Existing Literature

Some existing studies go into solutions built to use multiple screens, and there are white papers from multiple monitor hardware companies like Dell, NEC, and Fujitsu. While none of this focuses specifically on an enterprise CRM use-case, they discuss productivity and efficiency. Some good studies focus on display space usage from Dugald Ralph Hutchings and a paper that dives into productivity and efficiency gains when running a task across multiple monitors from Youn-ah Kang and John Stasko, where they ran a controlled test group and proved multiple monitors did indeed boost productivity. However, they also found that as employees get more experienced with the functions, the productivity gains begin to diminish, which likely has a high bearing on the CRM use-case that an employee will use consistently.

3.2. Scenarios Where Multi-Screen Experiences Add Value

When discussing multi-screen experiences, below are common scenarios that may come up.

- Editing tools: allow users to view before and after states side by side. Having two screens can also provide a more natural opportunity to compare two versions of the same type of content. These can be used to compare two images, lists, or documents side-by-side. While this is possible within single browser experiences, the additional real estate becomes a huge factor in such scenarios.
- Scenarios: need to show the same information in two different ways simultaneously. In this scenario, each screen works seamlessly to provide more information to the user. For example, show a list of restaurants on one screen and a map with locations on the other.

3.3. Real World Multi-Screen Experience Implementations at Scale

There are few real-world scenarios where cohesive multi-screen experiences are evangelized and effectively utilized.

- Casting: Screencasting is an approach that allows users to share their device screen with other platforms, whether for collaborative tasks, presentations or media consumption. With this capability, there are often avenues for users to interact by highlighting key information, providing instructions, etc. A good use-cases would be Google's Cast SDK, which enables users to extend their Android, iOS, or Web app to direct their streaming video and audio to a TV or sound system. The app casting often also becomes the remote control to play, pause, seek, rewind, stop, and otherwise control the media. This capability was primarily built for media consumption, though there have been some JS capabilities that allowed web applications running in browsers to be able to cast as well.
- Multi-screen devices: With the advent of foldable phones and other form factors from Samsung, Microsoft and other hardware companies, there have been significant software enhancements to the underlying stack to support building and solving multi-screen experiences. Microsoft even proposed a dual-screen standard to help the developer community build solutions for their foldable multi-screen devices. These protocols propose standardizing capabilities and providing better developer experiences to build and test such solutions.

3.4. Current Web Application multi-screen scenarios

While using multiple web pages across different screens has been around for some time, there is very little available out of the box through out-of-box HTML5 or third-party solutions. The available browser technology allows applications to exist on multiple screens. However, very few web developers leverage this setup to create cohesive multi-screen application experiences, likely due to gaps in the current ability to control and coordinate the user experience or the unknowns around topics like how to manage session, state, caching, events, etc. across multiple screens. In some instances, web applications are intentionally or unintentionally designed to allow and be useful when used in a side-by-side user experience.

- There are scenarios where a web application opens other applications in a new window or tab and leaves it to the user to arrange them and make the best use of the available screen space.
- There are instances where application developers allow for a dual view-like experience without designing two specific views but by supporting multi-instances of their apps. This approach can often take advantage of the built-in support the OS or Browsers provide. In such cases, application users either open multiple instances of these web applications by design or manually open them to view them side by side, benefiting from their functionality organically or by design.

However, both scenarios are largely manual and rely on the users effectively utilizing these capabilities rather than being systematically designed to operate this way.

4. Application Considerations When Solving Multi-Screen Apps

4.1. User Experience and Cognitive Overload

While multi-screen CRM solutions offer numerous advantages, they can also lead to a decline in overall user experience and increase employee cognitive overload. Navigating and processing information across multiple screens can be challenging, potentially reducing efficiency and accuracy. Multi-tasking, especially with CRM use-cases, may not be beneficial and can also increase confusion, errors and mistakes as agents are already dealing with customer conversations. Therefore, user experience is crucial in designing and delivering effective multi-screen experiences. Emphasis should be placed on intuitive user interfaces, comprehensive training, and task automation to mitigate cognitive overload. Some user-experience strategies to address cognitive overload include

- Automating workflows, particularly repetitive tasks, can significantly reduce cognitive overload. This not only makes tasks more efficient and streamlined but also positively impacts other crucial CRM and contact center KPIs
- Training programs that factor in the multi-screen experience usage guidelines and preparation are essential. These programs should not only guide employees on the functional aspects of the application but also provide recommendations on setting up applications and monitors for optimal use. The focus should be on enhancing overall efficiency in how employees' access and process information.
- Embracing AI and self-service options is crucial. Eliminating repetitive tasks that employees must perform can significantly reduce cognitive overload. This approach allows employees to concentrate on more urgent and engaging issues, ultimately enhancing their job satisfaction.

Integrating AI, automating workflows, and offering top-tier training can greatly reduce agent burnout and boost productivity.

4.2. Ergonomics

Various factors must be considered when designing a workspace tailored to employees' workdays. The following considerations are recommended if dealing with multiple monitors.

- Monitor Size and Hardware. For example, choosing monitors that are the same or similar in size is recommended.
- Positioning It may be recommended to position the primary monitor in front and the secondary monitor at an angle to one side. This is primarily dependent on the intended user experience. Also, both monitors should be positioned equidistantly, normally at the employee's arm's length and eye level.

- Viewing angle. It is recommended that monitors be positioned to have an optimal viewing angle of 15–30 degrees.
- Mouse should be positioned towards the primary monitor.
- Monitor arms are recommended to allow employees flexibility to adjust variables like height and tilt to suit their personal preferences.

Advice on body posture may also change when two monitors are part of the workspace. For example, it is recommended for employees to swivel in their chairs when looking between the two screens rather than twisting to avoid musculoskeletal risks.

4.3. State Management (and Session Token Storage) for Same Domain Applications

To deliver an effective multi-screen CRM experience, managing session and state sharing across instances is essential. For an integrated user experience, it is essential that the two application sessions remain consistent and reliable and do not need users to perform repetitive activities. For example, a CRM user who pulls up a customer account on one screen should see the information reflected on the other. Likewise, closing user accounts need not be performed on both screens. These actions require ensuring that the data remains consistent and up to date in real-time between the screens. The various concerns include

- Session Management: Maintaining user sessions across multiple screens ensures that users do not have to log in separately for each instance. This seamless experience is crucial for efficiency and user satisfaction. This is traditionally done by maintaining some kind of session token available to the application and transmitted in requests made to the server.
- State Synchronization: Ensuring that the state of the CRM (e.g., customer context, data, interaction history) is synchronized across all instances is vital for data consistency. Any changes made in one instance should be reflected in real-time across all other instances.

Several methods can be employed to share sessions and states between the various screens (and in real-time) running multiple CRM instances. Listed below are some common approaches to state and session management and their potential capabilities and constraints when it comes to multi-screen implementations. These assume the various instances of the application running in the same domain and will call out where the domain matters to the solution.

- Cookies: Cookies, or HTTP Cookies, are small files of information sent by web servers to web browsers over the HTTP protocol. Web browsers store these cookies for a specific period, as specified in the cookie metadata returned by the server. Under normal circumstances (e.g., no use of incognito mode, same browser), cookies returned by the server for one web application instance are available to all other instances running on the same browser. These cookies are transmitted with all subsequent requests made to the domain server for which they are set. This, along with the ability to secure cookies with HTTPS and configurations that disable client-side editing, makes cookies a great option for storing session management tokens. However, cookies have limitations regarding the amount and type of data they can store. They support a maximum of 4096 bytes of data, which is insufficient for the state management needs of modern applications. Additionally, web and app server restrictions on maximum header sizes limit applications using multiple cookies to store data.
- Storage Object Session or Local Storage: Session storage and local storage allow application developers to store name-value pairs on the client side. Unlike cookies, the data in storage objects is not sent to the server natively and needs an additional application code to be sent to server requests for session management. The key differences between the two storage types are that the data stored in local storage is available across all tabs of the same web application domain, while session storage restricts the data access to individual application tabs. This, along with local storage persisting data across the entire browsing session and its 5MB size limit, makes local storage a great option for multi-screen application state management.
- Global Variable: This is a custom variable created by the application under the global window object. This object is only available to the tab it is created in and, hence, is unsuitable for multi-screen application state management.
- Server-side sessions: This approach involves storing session data on the server, ensuring consistency across devices and sessions. While it offers reliable state management, it may introduce latency and dependency on network stability. Server-side sessions are advantageous for security and centralized control but require efficient server communication protocols.
- Service Worker: Service Workers behave like a browser proxy server that executes in an impendent context outside the application tabs, even if the web app refreshes or reloads. This is like having the server-side session, but data is stored by the service worker in the browser instead of on a server. Data stored by the service worker is accessible to other instances of multi-screen applications running in the browser. This makes service workers a good fit for state management needs. Service workers have the disadvantage of limited support, which can be complex to implement.
- IndexedDB: A more robust solution for storing and sharing state, IndexedDB allows for persistent data storage within the browser. It supports asynchronous interactions, making it suitable for complex CRM applications. IndexedDB is particularly useful for offline storage and large datasets, enabling efficient data retrieval and manipulation. The key difference from web storage API is the ability to store large amounts of data, its

asynchronous API, and its ability to survive tabs and even a browser crash. IndexedDB Storage can be accessed by all tabs of the same domain running in the browser and hence is another option for state management.

- Post Message: This is a method provided in JavaScript to communicate between open windows or iframes and supports cross origin communication. This solution only works when the windows know the target application reference to communicate. For security reasons, browsers restrict communication only between parent and child window applications (i.e. scenarios where a 'parent' window opens a second 'child' window) or to iframes running within a parent window. While not a state management solution, this approach can allow parent and child apps to communicate and keep the state updated. When multiple parent child windows are involved, this approach becomes complicated.
- Broadcast channel API: Broadcast channel API is a java-script feature that enables communication between multiple tabs, windows or iframes if they are of the same domain. While this is not a state management solution, it can enable other state management options by synchronizing data by broadcasting state changes to all open instances of the application. These application instances can listen and update their state. Broadcast channel solutions are especially efficient when many application instances need to be updated.
- MessageChannel API: MessageChannel API is another modern feature that allows communication between different windows, iframes or workers, even those from a different domain. The message channel is optimized for efficient data transfer and is well suited where performance concerns are paramount. Like other messaging options like broadcast and post message, this approach can help solve updating state between various tabs via messages sent over this channel.

4.4. State Management and Session Token Storage for Different Domain Applications

The last sub-section discussed options to store session and state data accessible to multiple tabs of the same domain. What are the options if the CRM is built of multiple domain apps.

- In terms of storage solutions, cookies, storage objects, service workers, and indexed DB are only available within the same domains and cannot be used by cross-domain state sharing.
- Server-side state storage does present a solution as long as the session token can be shared between the instances. Before considering this solution, the disadvantages mentioned in the earlier sub-section must be evaluated.
- Post messages can share data between cross domain applications if the parent-child relationship is maintained. Post messages are not a good option if any application sessions are involved or a parent-child relationship is not present.
- MessageChannel API provides an efficient way of sharing data/state across cross domain apps and is more maintainable, efficient, and manageable than post messages.
- There is a 'hacky' approach (refer to figure 1) where multiple cross domain parent applications can each open a child iFrame of the same domain. This child iFrame can then share state via local storage, making it available to parent applications via post messages. While this works, it is not recommended.

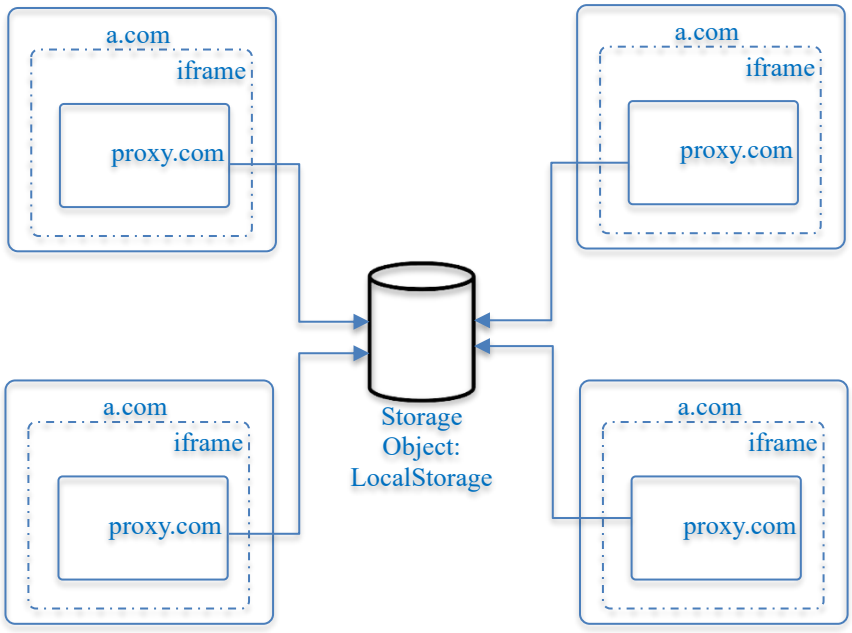

Fig. 1 Using local storage cross-domain

4.5. Performance Concerns and Resource Constraints

Modern development tools enable the creation of complex applications through modular development. An emerging trend, micro-frontends, allows developers to build large applications as small, independent components that come together at runtime for a cohesive experience. Whether the application is a monolith or a micro-frontend, it runs in a browser with regulated CPU, memory, and I/O thresholds. Running multiple instances of the application can often strain these constrained resources. Another performance concern is the impact of long-running web applications. Unlike public-facing sites end customers use, CRM sessions run longer as employees assist multiple customers. This often leads to a gradual buildup of resource usage over time, which can cause issues. These issues are exacerbated when multiple application instances are used simultaneously. The complexity of multi-session applications, such as the need to share state, can also cause problems. In a prior implementation, the data sent over post messages started hitting browser thresholds, causing the applications to freeze.

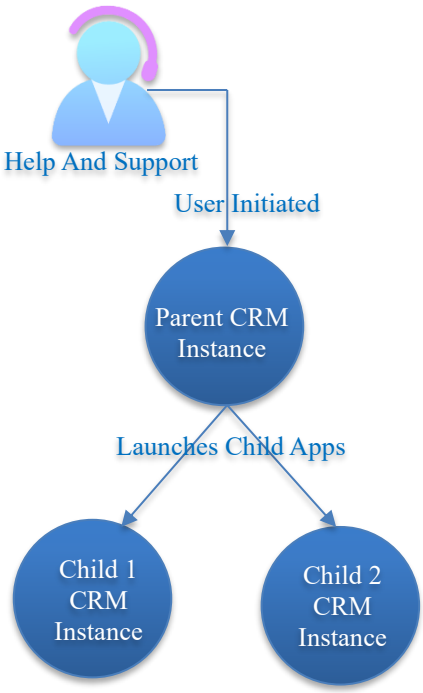

Fig. 1 Parent app managing windows

4.6. Screen/Window Management

To manage the multiple instances of the web application CRM systematically, it is recommended to have one instance play the role of a 'parent' app, which opens the other 'child' applications. This allows the parent application to manage the child applications in a co-ordinated way. This also allows the parent app to keep child window references, enabling capabilities like posting messages to share state and events and closing child windows if functionally desired. There are other ways to programmatically launch web apps that need the delivery of additional software to the user browser, like an electron or a native process that can launch browser instances. These add significant additional complexity to the solution.

4.7. Miscellaneous Concerns

Implementing multi-screen CRM solutions across various device types presents additional challenges like compatibility, form factors, etc.

- Device Compatibility: Ensuring consistent performance across desktops, tablets, and mobile devices requires careful optimization and testing. Different devices have varying screen sizes, processing power, and input methods, which must be accommodated in the CRM design. Also, browsers differ across devices and sometimes lack certain capabilities. For example, indexedDB has limited support for IE and Opera.
- User Experience: Adapting CRM interfaces for different screen sizes and resolutions can be complex, impacting the overall user experience. Also, solving for all such variations adds additional development complexity. Practices like Responsive design principles, adaptive layouts, and customizable interface options can help maintain usability across devices.
- Accessibility: A key aspect of UX design is accessibility, which requires that CRMs and other enterprise CRM solutions be inclusive and usable by users of various abilities. Multi-screen CRMs bring additional complexity to accessibility solutions.
- Security: A unique effect of managing session and state across two screens often drives the development of custom interaction methods outlined in earlier sections. This custom development can lead to potential security vulnerabilities unless properly designed.
- Testing: Most testing and synthetics solutions, including automation capabilities, are not built to solve multiple screen experiences out of the box and may need customization or in-house development.

5. Conclusion

CRM systems are critical for enterprise success, offering enhanced customer insights, improved service, and streamlined operations. However, their complexity necessitates advanced capabilities such as multi-screen and multi-monitor setups. While these solutions offer significant benefits, they also require careful consideration of implementation aspects, performance and non-functional implications, and overall usability and user experience. By addressing these challenges proactively, enterprises can leverage CRM systems to their full potential, driving growth and customer satisfaction.

References

[1] Dual View Design Patterns, Microsoft Ignite, 2022. [Online]. Available: https://learn.microsoft.com/en-us/dual-screen/design/dual-view

[2] Can I use?, Caniuse. [Online]. Available: https://caniuse.com/?search=indexed%20db

[3] Dual Screen Web Experiences, Microsoft Ignite, 2022. [Online]. Available: https://learn.microsoft.com/en-us/dual-screen/web/

[4] Developing Multi-screen Web Interface -- Such as the Bloomberg Terminal, Stack Overflow. [Online]. Available: https://stackoverflow.com/questions/13910421/developing-a-multi-screen-web-interface-such-as-the-bloomberg-terminal

[5] Tobias Uhlig, Web based Multi-Screen Apps Including Drag and Drop, Medium, 2021. [Online]. Available: https://medium.com/geekculture/web-based-multi-screen-apps-including-drag-drop-5e161da6507b

[6] How to Build Dual Screen Experiences for the Websites and Web Apps, Microsoft 365 Developer. [Online]. Available: https://www.youtube.com/watch?v=DXrZWsqXPVc

[7] Code – Internet of Things: Web Apps with Multiple Screens. [Online]. Available: https://docs.idew.org/code-internet-of-things/references/web-app-multiple-screens

[8] Jeff Toister, Are Dual Monitors Bad for Customer Service Agents?, Toister Solutions, 2016. [Online]. Available: https://www.toistersolutions.com/blog/2016/2/8/are-dual-monitors-bad-for-customer-service-agents#:~:text=It%20seems%20there%20are%20plusses,one%20screen%20to%20the%20next

[9] Boost Productivity with a Dual Monitor Desktop, Dell Small Business. [Online]. Available: https://i.dell.com/sites/csdocuments/Shared-Content_data-Sheets_Documents/en/au/Boost_Productivity_with_a_Dual_Monitor_Desktop.pdf

[10] Jeremy Watkin, Multi-tasking and Dual Monitors in the Contact Center, Customer Service Life, 2015. [Online]. Available: https://customerservicelife.com/multitasking-and-dual-monitors-in-the-contact-center/

[11] Google Cast SDK, Google Cast. [Online]. Available: https://developers.google.com/cast

[12] How to use Multiple Screens, Web-dev. [Online]. Available: https://web.dev/patterns/web-apps/multiple-screens

[13] Investigating Multi-window Browser Apps, Scott Logic, 2020. [Online]. Available: https://blog.scottlogic.com/2020/03/18/Investigating-Multi-Windowed-Apps.html

[14] Cognitive Overload in the Contact Center is Costing You More Than You Think, ICMI, 2024. [Online]. Available: https://www.icmi.com/resources/2024/ccaas-cognitive-overload

[15] Second Screen Working Group, Discover W3C Groups. [Online]. Available: https://www.w3.org/groups/wg/secondscreen/

[16] Window Management, W3C Working Draft, 2024. [Online]. Available: https://www.w3.org/TR/window-management/

[17] Louay Bassbouss et al., "Towards a Multi-screen Application Model for the Web," *2013 IEEE 37th Annual Computer Software and Applications Conference Workshops*, Japan, pp. 528-533, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[18] Dual Screen Do's and Don'ts, System Concept. [Online]. Available: https://www.system-concepts.com/insights/dual-screen/

[19] Whitepaper: Optimal Performance with Two or More Computer Screens, Bakker Elkhuizen. [Online]. Available: https://www.bakkerelkhuizen.com/en-us/knowledge/whitepaper-optimal-performance-with-two-or-more-computer-screens/

[20] Multi Monitor Setup and Productivity, Medium, 2021. [Online]. Available: https://zetablogs.medium.com/multi-monitor-setup-and-productivity-48f98e36cd1

[21] Second Screen working Group, Discover W3C Groups. [Online]. Available: https://www.w3.org/groups/wg/secondscreen/

[22] Youn-ah Kang, and John Stasko, "Lightweight Task/Application Performance using Single Versus Multiple Monitors: A Comparative Study," Graphics Interface Conference, pp. 17-24, 2008. [Google Scholar] [Publisher Link]

[23] Dugald Ralph Hutchings et al., "Display Space Usage and Window Management Operation Comparisons between Single Monitor and Multiple Monitor Users," *Proceedings of the Working Conference on Advanced Visual Interfaces*, pp. 32-39, 2004. [CrossRef] [Google Scholar] [Publisher Link]