https://doi.org/10.22201/dgtic.ctud.2023.1.1.22

Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 25 - 34

Interfaces web dinámicas: solución a las limitaciones que presentan *Vue* y *React*

Pablo Enrique Zenil Rivas

Dirección General de Cómputo y de Tecnologías de Información y Comunicación

Universidad Nacional Autónoma de México

pablozr@unam.mx

ORCID: 0009-0007-3956-0194

Información del reporte:

Para citar este reporte técnico:

Zenil Rivas, P. E. (2023). Interfaces web dinámicas: solución a las limitaciones que presentan Vue y React. *Cuadernos Técnicos Universitarios de la DGTIC, 1* (1), páginas (25 - 34). https://doi.org/10.22201/dgtic.ctud.2023.1.1.22

Resumen:

Las herramientas como *Vue* y *React* son muy populares en el desarrollo eficiente y escalable de Interfaces de Usuario; son utilizadas para implementar aplicaciones de gran tamaño y complejidad. Sin embargo, cuando se requiere de una experiencia de usuario altamente dinámica y responsiva, estas herramientas presentan algunas limitaciones como: la comunicación entre componentes, el registro de cambios en tiempo real y la manipulación directa del *DOM*. En este reporte técnico se propone e implementa la solución a las limitaciones que se presentaron en el desarrollo del proyecto *Laboratorio Virtual de Química*. La solución conjunta, además del despliegue declarativo de componentes reactivos, los paradigmas de desarrollo web: comunicación vía emisor-suscriptor de eventos y manipulación del Modelo de Objetos del Documento *HTML* (*DOM*). Se implementó una interfaz web intuitiva, altamente dinámica y responsiva a las interacciones del usuario, que permite la comunicación, interacción y seguimiento en tiempo real de los componentes que representan los distintos materiales y sustancias del laboratorio.

Palabras clave:

Interfaces web dinámicas, Vue, React.

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 26 - 34

1. INTRODUCCIÓN

Las herramientas como *Vue* (Vue team, 2022) y *React* (React team, s./f.) cubren la mayoría de las necesidades de un sistema o aplicación en el desarrollo de interfaces gráficas web. Estas herramientas se basan en los estándares de *HTML*, *CSS* y *JavaScript* y proveen un modelo de programación declarativo (Wikipedia contributors, 2023a) basado en componentes reactivos, que permite el desarrollo eficiente y escalable de interfaces de usuario (W3Schools, s.f). Lo anterior las hace ideales para implementar aplicaciones de gran tamaño y complejidad. Por esta razón, cada vez más desarrolladores y equipos de desarrollo utilizan este tipo de herramientas.

Hay dos conceptos centrales que se deben tomar en cuenta al utilizar estas herramientas:

- **Despliegue declarativo.** Describe el estado final de los componentes basados en *HTML* para cualquier estado de *JavaScript*.
- **Reactividad.** Cada componente realiza el seguimiento de los cambios en sus dependencias reactivas (el estado de *JavaScript*), y cuando suceden, se actualiza de forma eficiente. Estos cambios se ven reflejados en el Modelo de Objetos del Documento (MDN Web Docs, 2019). A este tipo de componentes se les llama componentes reactivos.

La interfaz de usuario se compone de unidades pequeñas como botones, texto e imágenes. Esto permite combinarlas en componentes anidables y reutilizables. Desde sitios web hasta aplicaciones de celular, toda interfaz puede descomponerse en componentes reactivos. Cada componente encapsula en un bloque su comportamiento, sus datos, su lógica y su estado, lo que lo hace reutilizable y fácil de mantener.

Sin embargo, cuando se requiere de una experiencia de usuario altamente dinámica, estas herramientas presentan las siguientes limitaciones:

- **Comunicación entre componentes.** El flujo de la comunicación entre componentes solamente se da entre padre e hijo. Los componentes anidados que no guardan esta relación, no pueden comunicarse directamente entre sí.
- **Registro de cambios continuos en los componentes.** La reactividad de cada componente lleva el seguimiento de cambios discretos, es decir, cambios que se derivan de eventos aislados como las interacciones del usuario (presionar un botón, teclear texto, entre otros). Los cambios continuos en el tiempo, como posiciones derivadas de una transición animada o la variación de temperatura en una simulación, requieren de un tratamiento especial.
- **Manipulación directa del Modelo de Objetos del Documento *HTML* (*DOM*).** El despliegue declarativo y la reactividad de los componentes se encargan de la manipulación del *DOM*. Sin embargo, algunas interacciones de usuario como arrastrar y soltar, implican una manipulación directa del *DOM* que debe ser implementada de manera independiente.

El presente reporte técnico describe el proceso de solución de las limitaciones anteriormente mencionadas, que se implementó en el desarrollo del proyecto *Laboratorio Virtual de Química:* un ambiente web de simulación que contribuye a promover la experimentación para la resolución de problemas de diluciones en agua destilada y titulación de ácidos/bases fuertes (De Levie, 2001; McCord y Stanton, 2005).

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 27 - 34

2. OBJETIVO

Implementar una interfaz web intuitiva, altamente dinámica y responsiva a las interacciones del usuario que permita la comunicación, interacción y seguimiento en tiempo real de los componentes que representan los distintos materiales y sustancias del Laboratorio Virtual de Química y proponer una solución a las limitaciones entre componentes reactivos que presentan las herramientas como *Vue* y *React*.

3. DESARROLLO

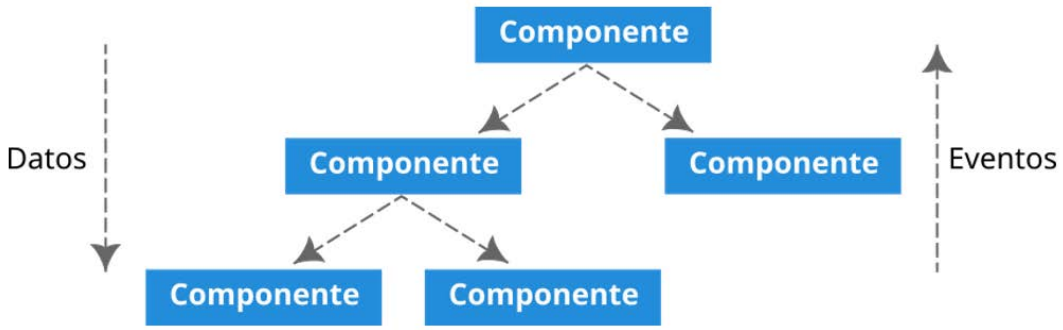
El tipo de interacción que se necesitó implementar implica que el usuario pueda posicionar distintos materiales en la mesa de trabajo, arrastrar y soltar componentes unos sobre otros, para activar interacciones entre ellos, por ejemplo, el vaciado de sustancias y la medición de sus propiedades fisicoquímicas.

3.1 COMUNICACIÓN ENTRE COMPONENTES

En los componentes anidados, la comunicación se da entre padre e hijos, como se muestra en la figura 1.

Figura 1

Flujo de comunicación entre componentes



Nota. La comunicación entre componentes, con este paradigma, se da de manera directa

Cada hijo recibe datos directamente de su padre, y le responde mediante eventos. Esto limita la interactividad a los elementos que tienen una relación de padre e hijo. Si bien la reactividad y el despliegue declarativo de los componentes facilitan el desarrollo de sistemas complejos, al integrar al árbol nuevos componentes, muy probablemente romperán con la relación padre-hijo de otros ya establecidos y, con esto, su comunicación. Una solución a este problema es proveer un mecanismo de comunicación directa entre cualquier componente. Esto se logra mediante la implementación del paradigma de emisor-suscriptor de eventos (Wikipedia contributors, 2023b). En este paradigma, un componente, a través de un objeto global Emisor, emite eventos con un nombre definido, lo que ocasiona que los componentes receptores previamente suscritos a ese evento dado ejecuten una acción pertinente.

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 28 - 34

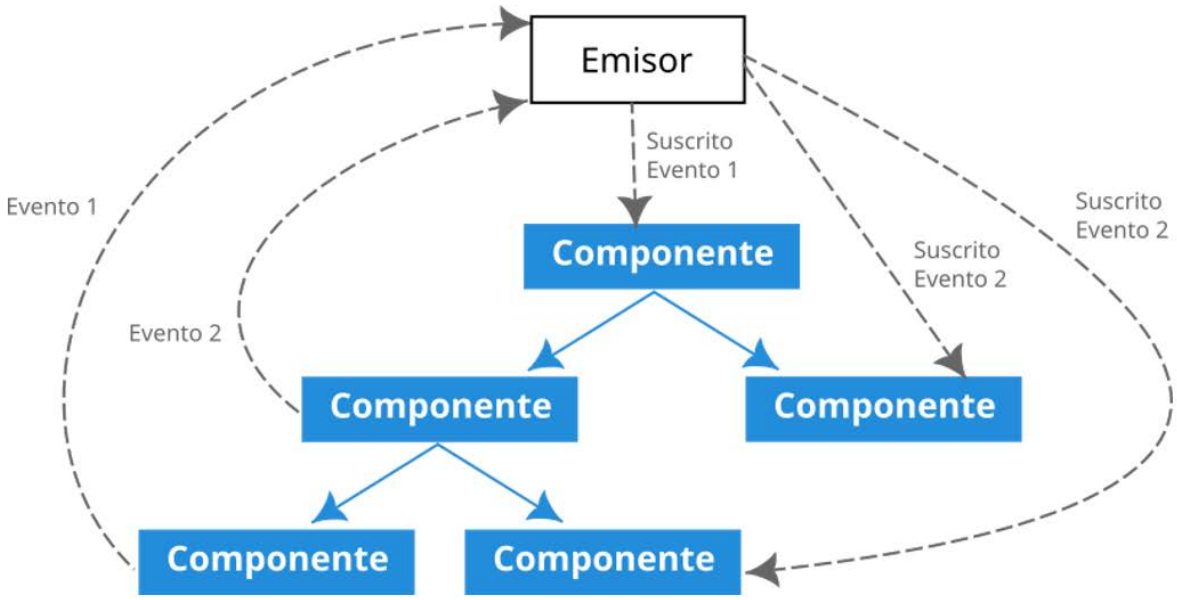
El objeto emisor cumple con dos funciones:

- Emitir eventos.
- Registrar y eliminar funciones receptoras.

Lo anterior habilita la comunicación directa entre cualquier par de componentes. Por ejemplo, como se muestra en la figura 2, un componente es capaz de emitir el Evento 1 y otro componente distinto emite el Evento 2. Los componentes que requieran reaccionar a estos eventos se suscriben al emisor para escuchar a cada uno de ellos. Los componentes se pueden suscribir y también pueden emitir cualquier número de eventos.

Figura 2

Utilizando el paradigma emisor-suscriptor



Nota. La comunicación entre componentes, con este paradigma, se da de manera directa.

La limitante de la comunicación entre componentes no anidados queda resuelta, y se eleva así el grado de interacción. Bajo este paradigma, la complejidad y el número de componentes puede escalar fácilmente, ya que pueden ser incorporados en cualquier nivel del árbol sin romper la comunicación de componentes ya establecidos. El emisor puede ser implementado de manera nativa con *JavaScript*, sin embargo, en el desarrollo del laboratorio se utilizó la herramienta de terceros *Mitt* (Miller, s/f), al ser una implementación ligera, funcional y fácil de utilizar.

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 29 - 34

3.2 REGISTRO DE CAMBIOS CONTINUOS EN LOS COMPONENTES

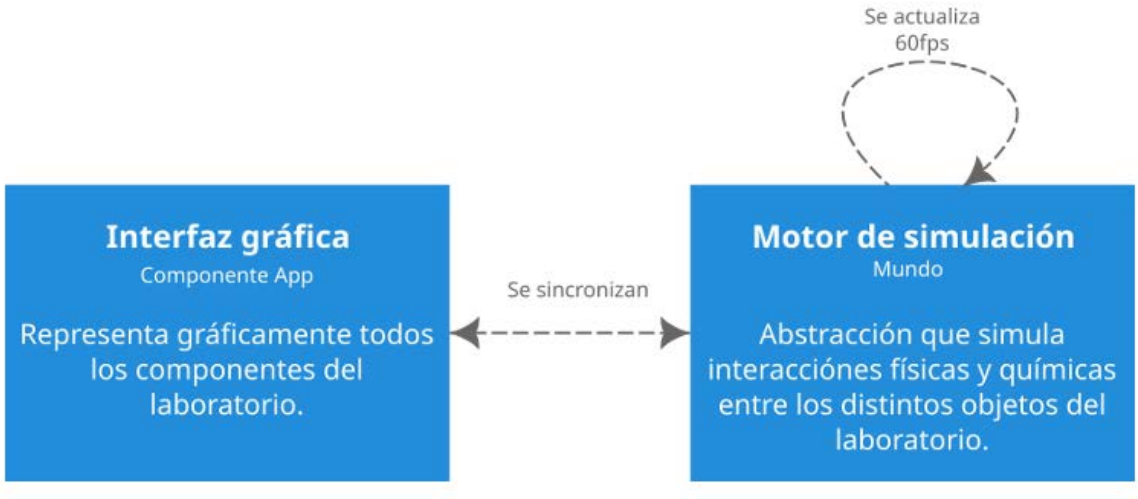
La aplicación consta de dos elementos principales, como se muestra en la figura 3:

1. **Motor de simulación de reacciones químicas.** Abstrae al mundo, el cual modela a las distintas sustancias en términos de su composición química mediante la *Molaridad* (De Levie, 2001; McCord & Stanton, 2005), temperatura y volumen. El simulador depende del bucle de ejecución del navegador y 60 veces por segundo, calcula en tiempo real los avances de reacción (De Levie, 2001; McCord & Stanton, 2005; Saker Neto, 2018), entalpía de mezclado/reacción (De Levie, 2001; McCord & Stanton, 2005) y enfriamiento (Downey, 2022), para poder representar de forma dinámica y realista las distintas propiedades fisicoquímicas, como pH y temperatura, de las sustancias y contenedores que interactúan entre sí.

2. **Interfaz gráfica.** Tiene como raíz al componente *App* y representa gráficamente, mediante componentes, a los elementos del mundo, así como sus propiedades fisicoquímicas. Además, permite que el usuario los manipule e interactúe con ellos.

Figura 3

Elementos principales de la aplicación

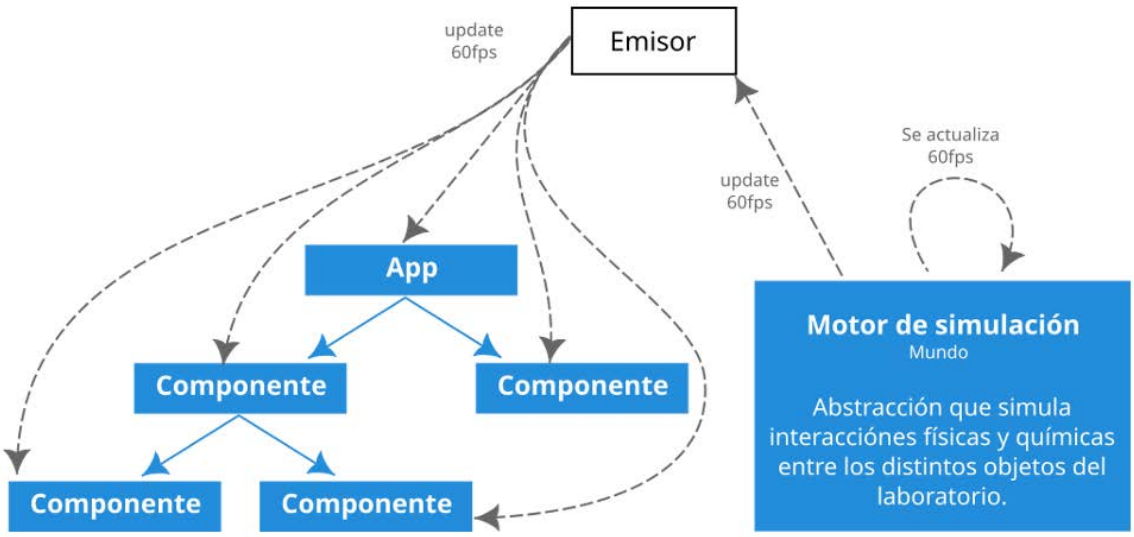


Tras haber implementado el emisor, resolver la sincronización es relativamente fácil, como se ilustra en la figura 4, pues basta con suscribir a todos los componentes al evento *update*, emitido por el mundo (60 veces por segundo), que es el encargado de notificar que se ha ejecutado un cuadro de tiempo del *bucle* de ejecución. Aquí cabe resaltar que no solo los componentes reactivos pueden emitir y escuchar eventos, sino que también otros elementos (como el mundo) de la aplicación pueden comunicarse con los componentes.

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 30 - 34

Figura 4

Sincronización entre el motor de simulación y los componentes



Nota. El emisor se encarga de notificar a todos los componentes para que se actualicen en tiempo real.

3.3 INTERACCIONES ENTRE COMPONENTES

Los componentes principales representan materiales de laboratorio como el vaso de precipitado, matraz, pipeta, entre otros. El usuario puede arrastrarlos y colocarlos en cualquier parte de la mesa de trabajo que es el componente padre y representa casi toda el área visible de la ventana del navegador. Cada envase puede contener un volumen determinado de alguna sustancia y al ser arrastrado y soltado sobre otro envase, se puede verter parte de esa sustancia. Parte importante de la interacción del usuario está en arrastrar y soltar los componentes de la mesa de trabajo. Cada componente notifica los siguientes eventos a cada suscriptor, los que se derivan de la interacción del usuario:

- Se está arrastrando.
- Ha sido soltado.
- Le ha sido soltado otro componente.
- Ha sido seleccionado (se presionó sobre él sin ser arrastrado).

Además, también notifica en tiempo real los siguientes eventos del mundo:

- Cuánto volumen de sustancia contiene (si es envase).
- La temperatura de la sustancia que contiene (si es envase).
- La molaridad de la sustancia que contiene (si es envase).

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 31 - 34

- Los valores de sensor (pH en el caso del pH-metro).
- El ángulo de inclinación (si aplica).
- Cuánto volumen de sustancia está vertiendo (si es envase).
- En qué zona de la mesa de trabajo se encuentra.

Cada componente suscriptor reacciona al evento que le corresponde. Por ejemplo, si un vaso de precipitado es arrastrado y soltado en un matraz, el matraz sabe que puede recibir la sustancia del vaso y a su vez, el vaso sabe a quién verter la sustancia.

El arrastrado y soltado de componentes es una interacción especial que requiere del paradigma de manipulación del *DOM*, que nos permite: agregar y eliminar elementos al documento *HTML*, modificar atributos de elementos y ejecutar funciones que reaccionan a los distintos eventos, entre otros. En la implementación de esta interacción en el laboratorio, utilizamos la herramienta de terceros *jQuery* (jQuery team, s/f), una herramienta eficiente y ligera que facilita la manipulación del *DOM*.

Con todo lo anterior, solo falta asegurar que todos los componentes representen gráficamente su estado en tiempo real (volumen de líquido, inclinación, temperatura, color), como se muestra en la figura 5. Lo anterior se logra al actualizar el estado visual de cada componente en el evento update, manipulando la parte gráfica vía *CSS* y con la herramienta de terceros *Paper.js* (Lehni & Puckey, s/f), la cual provee una funcionalidad ligera y muy poderosa en la creación y manipulación de gráficos.

Figura 5

Representación gráfica en tiempo real, del estado de algunos componentes

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 32 - 34

4. RESULTADOS

Se implementó el proyecto Laboratorio Virtual de Química donde se usó *Vue* para el manejo de componentes reactivos. El sistema cuenta con una interfaz web intuitiva, altamente dinámica y responsiva a las interacciones del usuario, que permite la comunicación, interacción y seguimiento en tiempo real de los componentes que representan los distintos materiales y sustancias del laboratorio; es el resultado de la conjunción de tres principales paradigmas de programación web:

1. **Despliegue declarativo de componentes reactivos:** permite el desarrollo de una interfaz intuitiva y responsiva a los eventos de usuario, mediante el encapsulamiento de cada componente y su funcionalidad.
2. **Comunicación mediante el emisor-suscriptor de eventos:** permite una comunicación directa entre componentes y elementos del sistema, lo que eleva el nivel de interacción y escalabilidad del mismo.
3. **Manipulación directa de los atributos y propiedades del *DOM*:** permite el desarrollo de interacciones que el uso de componentes reactivos no contempla, como el arrastrado y soltado de componentes.

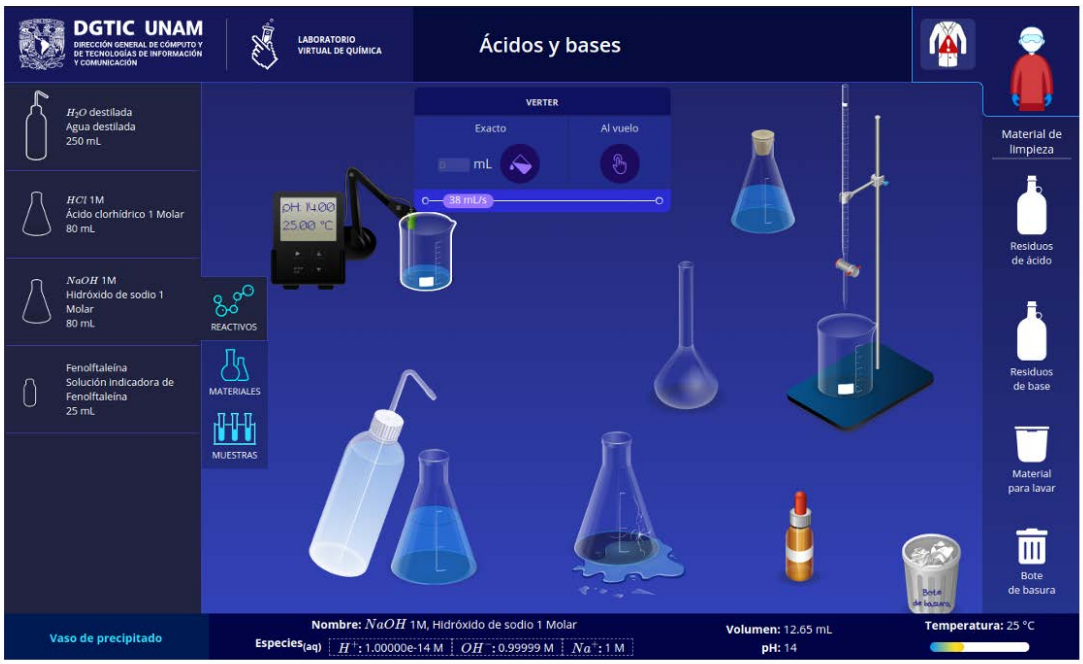
La experiencia en el desarrollo de sistemas web permitió proponer una solución sencilla y escalable en la que:

- Nuevos componentes reactivos como materiales, menús o ventanas emergentes, pueden ser incorporados fácilmente en cualquier nivel del árbol de componentes, sin interferir en lo que ya es funcional.
- Nuevos elementos que representen objetos del mundo como sustancias, sensores o materiales, pueden ser incorporados con la misma facilidad.
- Se pueden definir nuevos eventos e interacciones sin interferir con lo ya implementado.
- Cualquier nuevo componente o elemento solo tiene que definir y/o suscribirse a los eventos de las interacciones que le corresponden e implementarlas según sea el caso.
- Cabe destacar que esta solución es aplicable en cualquier escenario en el que se requiera usar cualquier tecnología basada en componentes reactivos. Una captura de pantalla del producto final se puede apreciar en la figura 6.

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 33 - 34

Figura 6

Captura de pantalla del Laboratorio Virtual de Química



5. CONCLUSIONES

Herramientas basadas en componentes reactivos como *Vue* y *React*, cubren la mayoría de las necesidades de un sistema o aplicación en el desarrollo de interfaces gráficas web. Sin embargo, cuando se requiere de una experiencia de usuario altamente dinámica y responsiva, estas herramientas presentan las siguientes limitaciones:

- **Comunicación entre componentes:** los componentes anidados que no guardan una relación de padre e hijo no pueden comunicarse directamente entre sí.
- **Registro de cambios continuos en los componentes:** los cambios continuos en el tiempo, como la variación de temperatura en una simulación, requieren de un tratamiento especial.
- **Manipulación directa del *DOM*:** las interacciones de usuario como el arrastrado y soltado, implican una manipulación directa del DOM que debe ser implementada de manera independiente.

En el proyecto Laboratorio Virtual de Química, la experiencia en el desarrollo de sistemas web permitió proponer e implementar una solución a dichas limitaciones, sencilla, escalable y que conjunta los siguientes paradigmas de desarrollo web:

https://doi.org/10.22201/dgtic.ctud.2023.1.1.12 Vol. 1, núm. 1. Octubre-diciembre 2023, págs. 34 - 34

1. **Despliegue declarativo de componentes reactivos:** permite el desarrollo de una interfaz intuitiva y responsiva a los eventos de usuario.
2. **Comunicación mediante el emisor-suscriptor de eventos:** permite una comunicación directa entre componentes y elementos del sistema.
3. **Manipulación directa de los atributos y propiedades del *DOM*:** permite el desarrollo de interacciones que no contemplan el uso de componentes reactivos.

Lo anterior permitió implementar una interfaz web intuitiva, altamente dinámica y responsiva a las interacciones del usuario, que permite la comunicación, interacción y seguimiento en tiempo real de los componentes que representan los distintos materiales y sustancias del laboratorio. Esta solución es aplicable en cualquier escenario en el que se requiera usar una tecnología similar basada en componentes reactivos y que presente las mismas limitaciones.

REFERENCIAS BIBLIOGRÁFICAS

De Levie, R. (2001). *How to use Excel in analytical chemistry and in general scientific data analysis*. Cambridge University Press.

Downey, A. (2022). *Modeling And Simulation In Python*. O'reilly Media.

jQuery team. (s.f.). *jQuery*. Recuperado el 7 de septiembre de 2023, de https://jquery.com

Lehni, J., & Puckey, J. (s.f.). *Paper.js*. paperjs.org. Recuperado el 7 de septiembre de 2023, de http://paperjs.org

McCord, P., & Stanton, E. (2005). *pH, Titrations, and Dilutions. Course: General Chemistry I; The University of Texas at Austin*. http://mccord.cm.utexas.edu/courses/spring2005/ch301/concentrations.html

MDN Web Docs. (2019, agosto 16). *Introduction to the DOM*. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction

Miller, J. (s.f). *Mitt*. GitHub. Recuperado el 7 de septiembre de 2023. https://github.com/developit/mitt

React team. (s.f.). *Describing the UI – React*. react.dev. Recuperado el 7 de septiembre de 2023. https://react.dev/learn/describing-the-ui

Saker Neto, N. (2018). *The reason behind the steep rise in pH in the acid base titration curve*. Chemistry Stack Exchange. https://chemistry.stackexchange.com/q/8074

Vue team. (2022). *Introduction* | Vue.js. vuejs.org. https://vuejs.org/guide/introduction.html

W3Schools. (s.f.). *What is a Front-End Developer*. www.w3schools.com. Recuperado el 7 de septiembre de 2023. https://www.w3schools.com/whatis/whatis_frontenddev.asp

Wikipedia contributors. (2023a, septiembre 7). *Declarative programming*. Wikipedia. https://en.wikipedia.org/w/index.php?title=Declarative_programming&oldid=1174296946

Wikipedia contributors. (2023b, septiembre 7). *Event-driven programming*. Wikipedia. https://en.wikipedia.org/w/index.php?title=Event-driven_programming&oldid=1174274696