https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169

Vol. 4, Núm. 2. abril-junio 2026, págs. 65 - 78

Modernización y estandarización en el desarrollo de complementos para Moodle

Modernization and Standardization in Moodle Plugin Development

Información del reporte:

Para citar este reporte técnico:

Zenil Rivas, P.E. (2026). Modernización y estandarización en el desarrollo de complementos para Moodle. *Cuadernos Técnicos Universitarios de la DGTIC, 4* (2), páginas (65 - 78). https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169

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

Resumen

El TICómetro de la Universidad Nacional Autónoma de México enfrentaba un doble desafío: la obsolescencia técnica de sus simuladores, la cual impidió su operación en versiones actualizadas de la plataforma, y la falta de un entorno de desarrollo estandarizado a nivel institucional. Para resolver esta problemática, se conformó un grupo multidisciplinario que definió una metodología ágil de desarrollo. El trabajo incluyó la comparación de diversas infraestructuras, la cual respaldó la decisión de elegir una solución basada en contenedores debido a su flexibilidad y reproducibilidad. Paralelamente, se diseñó una plantilla de complemento reutilizable que integró tecnologías web modernas dentro de la arquitectura de Moodle, lo que superó las limitaciones de su sistema de módulos nativo.

Como resultado, se implementó una versión moderna del simulador de hoja de cálculo que incorporó un sistema de evaluación granular, enfocado en analizar las características de la respuesta del estudiante, resolviendo así la ambigüedad de versiones previas. Se concluyó que el proyecto logró establecer un ambiente de desarrollo estandarizado, el cual ya ha sido utilizado en la creación e integración de nuevos simuladores.

Palabras clave: Contenedores en Docker, Node.js, interfaces con React, simuladores para TICómetro.

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 66 - 78

Abstract

The TICómetro project of the Universidad Nacional Autónoma de México *faced a dual challenge: the technical obsolescence of its simulators, which prevented their operation on updated platform versions, and the lack of a standardized institutional development environment. To solve this problem, a multidisciplinary group was formed to define an agile development methodology. The work included comparing various infrastructures, ultimately opting for a container-based solution due to its flexibility and reproducibility. In parallel, a reusable plugin template was designed that integrated modern web technologies into the Moodle architecture, overcoming the limitations of the native module system.*

As a result, a modern version of the spreadsheet simulator was implemented, incorporating a granular evaluation system focused on analyzing the characteristics of the student's response, thus resolving the ambiguity of previous versions. It was concluded that the project successfully established a standardized development environment, which has already been used in the creation and integration of new simulators.

Keywords: *Docker containers, Node.js, React interface, simulators for TICómetro.*

1. INTRODUCCIÓN

La evaluación de habilidades en tecnologías de información y comunicación (TIC) es fundamental en la educación (Juca-Maldonado *et al*., 2025). En la Universidad Nacional Autónoma de México (UNAM), el TICómetro (s. f.) es un instrumento de evaluación diagnóstica diseñado con el propósito de contar con información sobre el nivel de habilidades en el uso de las TIC de los estudiantes de bachillerato y licenciatura de nuevo ingreso a la UNAM. El instrumento, diseñado por la Dirección General de Cómputo y de Tecnologías de Información y Comunicación, proporciona datos empíricos a las entidades académicas. Estos datos permiten caracterizar el perfil de los estudiantes de primer ingreso en relación con sus habilidades en el manejo de las TIC; además, ofrece información relevante para la toma de decisiones en cuanto a la enseñanza y el uso de las TIC en los programas, actividades y necesidades de infraestructura. El instrumento se valida estadísticamente cada año. Su implementación inicial se realizó sobre la plataforma Moodle, de la cual se aprovechó el módulo de cuestionario para automatizar la calificación e integrar simuladores que evalúan habilidades prácticas.

El avance tecnológico y la migración de la plataforma a la versión 4.3 requirieron una actualización integral del *software*. Esto propició la actualización tecnológica de tres simuladores clave del TICómetro (hoja de cálculo, procesador de texto y búsqueda en Internet) y evidenció la necesidad de un entorno de desarrollo unificado en el área responsable del servicio, para facilitar el mantenimiento y el desarrollo futuro.

Para solventar esto y seguir las mejores prácticas en metodologías ágiles (López-Mora *et al*., 2019), se conformó un grupo multidisciplinario. El equipo definió como premisas principales evitar complementos de paga, garantizar la compatibilidad con el núcleo de Moodle, facilitar el mantenimiento, fomentar la colaboración y publicar los resultados en la comunidad oficial de Moodle (Moodle, s.f.).

El objetivo general fue modernizar el desarrollo de complementos de simulación para el TICómetro mediante la creación de una plantilla que integra tecnologías web modernas (Lucas *et al*., 2025) como React y Vite, superando las limitaciones del sistema de módulos nativo. Para asegurar la reproducibilidad del entorno, se optó por una arquitectura basada en contenedores (Wang, 2022). Esta metodología,

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 67 - 78

desarrollada entre el segundo semestre de 2024 y el primero de 2025, se aplicó en la actualización del simulador de hoja de cálculo y facilita la creación de nuevas herramientas que evalúen habilidades digitales emergentes.

2. DESARROLLO TÉCNICO

Para alcanzar el objetivo propuesto, el desarrollo se dividió en dos áreas principales: primero, se definió la metodología de trabajo y se implementaron los entornos de desarrollo; y segundo, se avanzó con el desarrollo del simulador de hoja de cálculo y, de forma paralela, con el de la plantilla de complemento tipo pregunta reutilizable, que sirvió como base para los nuevos simuladores.

2.1 METODOLOGÍA

El trabajo se organizó bajo un marco ágil (López-Mora *et al.*, 2019), mediante sesiones quincenales para la revisión de avances, discusión de soluciones y redefinición de prioridades. Se utilizó el GitLab institucional para el control de versiones y la gestión de la comunicación entre los académicos.

2.1.1 ANÁLISIS Y SELECCIÓN DEL ENTORNO DE DESARROLLO

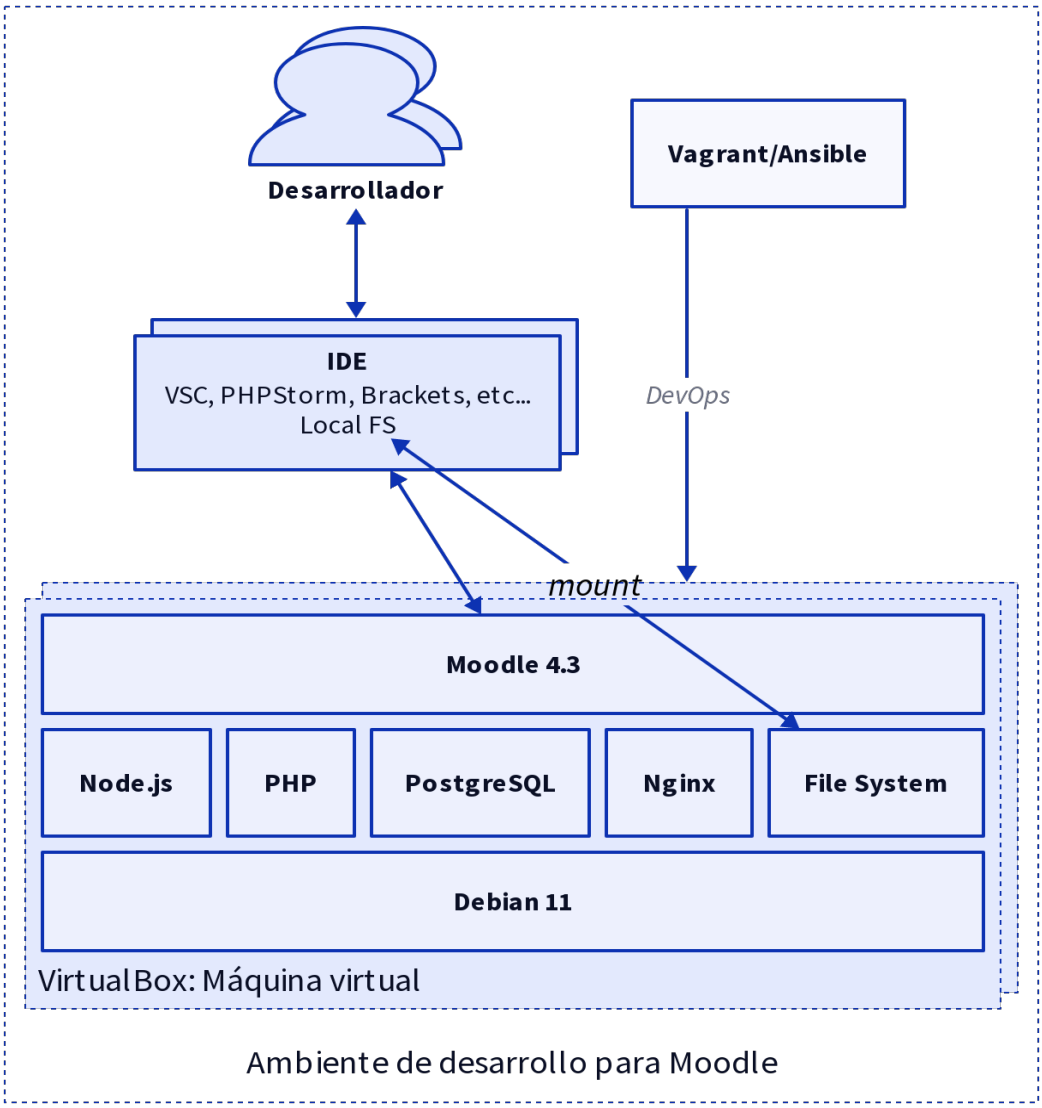
El proyecto requería un entorno de desarrollo reproducible con dependencias específicas para Moodle 4.3, incluyendo tecnologías de servidor y bases de datos como PHP, PostgreSQL y Nginx, así como las herramientas necesarias para la nueva plantilla web, tales como Node.js, React y Vite.

La primera solución utilizó máquinas virtuales en VirtualBox, gestionadas con Vagrant, para la reproducibilidad, y Ansible, para la automatización de la instalación sobre el sistema operativo Debian. Este enfoque empaquetaba el entorno en archivos de configuración, asegurando la replicabilidad. En la Figura 1 se presenta el ambiente de desarrollo utilizado.

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 68 - 78

Figura 1

Ambiente de desarrollo para Moodle con Vagrant



Sin embargo, esta solución presentó dos deficiencias:

- Manejo del sistema de archivos: obligaba al desarrollador a instalar sus herramientas de desarrollo (IDE) dentro de la máquina virtual, generando un cuello de botella en el flujo de trabajo.
- Gestión de versiones: para soportar otra versión de Moodle, se debía crear una máquina virtual completamente nueva, creando redundancias innecesarias.

En una segunda instancia, se adoptó el uso de contenedores con Docker y Docker Compose. Esta alternativa conservó las ventajas de las máquinas virtuales (replicando la arquitectura de la Figura 1) y

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 69 - 78

solucionó sus deficiencias, lo cual permitió trabajar sobre el sistema de archivos local y agilizó el flujo de trabajo (Wang, 2022).

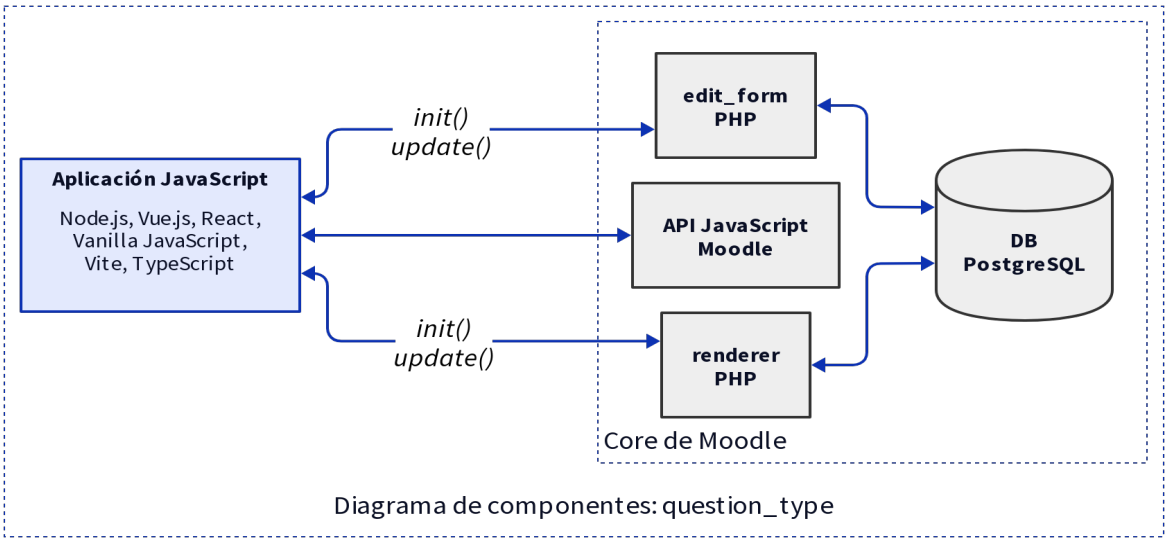
2.1.2 DISEÑO DE LA PLANTILLA DE COMPLEMENTO

Un reto clave fue diseñar una plantilla que permitiera el desarrollo sobre un entorno moderno de JavaScript (Lucas *et al.*, 2025), ya que Moodle utiliza nativamente un *stack* obsoleto basado en AMD (Wikipedia contributors, 2025), RequireJS y Grunt. Dicho *stack* presenta compilaciones lentas, requiere recarga manual y dependencias antiguas, tales como Node.js 14 o 16, (la versión actual, al momento del desarrollo, era la 22).

Por lo tanto, el diseño de la nueva plantilla se enfocó en el módulo App, el cual se presenta en la Figura 2 como una aplicación JavaScript general que se comunica con los componentes de Moodle.

Figura 2

Diagrama de componentes plantilla de complemento tipo pregunta



Dicha plantilla puede también desarrollarse en un entorno moderno de Node.js.

2.1.3 DISEÑO DEL SIMULADOR DE HOJA DE CÁLCULO

Para el simulador de hoja de cálculo, se analizaron varios paquetes de código abierto y se eligió FortuneSheet (Ruilisi, s. f.). La elección se fundamentó en su compatibilidad nativa con React, su facilidad de configuración y una interfaz de programación de aplicaciones (API) que provee información relevante sobre el contenido de cada celda (fórmulas y referencias, entre otros).

Además, se rediseñó el sistema de evaluación. La versión anterior obligaba a configurar múltiples "respuestas esperadas" por reactivo, ya que era imposible analizar las características de la celda. Esto generaba un problema crítico en la evaluación (Pordanjani & Salehi, 2025): si una respuesta correcta no estaba preconfigurada, el sistema la calificaba como incorrecta, invalidando la automatización. La nueva

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 70 - 78

versión subsanó esto al permitir la especificación de las características de cada celda (por ejemplo: uso de fórmula, función o rango) en una única respuesta esperada, lo que garantizó una evaluación objetiva.

3. RESULTADOS

A continuación, se presentan los resultados obtenidos tras aplicar la metodología propuesta. Primero, explicaremos cómo armamos un entorno de desarrollo unificado que el equipo puede replicar fácilmente mediante contenedores, y después presentaremos el desarrollo técnico del nuevo complemento y cómo funciona en la práctica el simulador de hoja de cálculo ya integrado en Moodle.

3.1 ENTORNO DE DESARROLLO ESTANDARIZADO

Se adaptó el repositorio público *moodle-docker* (Moodle HQ, s. f.), el cual replica la arquitectura mostrada en la Figura 1 sin las desventajas de las máquinas virtuales. La adaptación se centró en la carga de parámetros mediante variables de entorno en un archivo .env, siguiendo un enfoque de cero configuración (*zero-configuration*). Esta solución facilita probar los complementos contra múltiples versiones de bases de datos, PHP y Moodle, proveyendo al equipo de un entorno ágil (Wang, 2022).

3.2 PLANTILLA DE COMPLEMENTO MODERNO

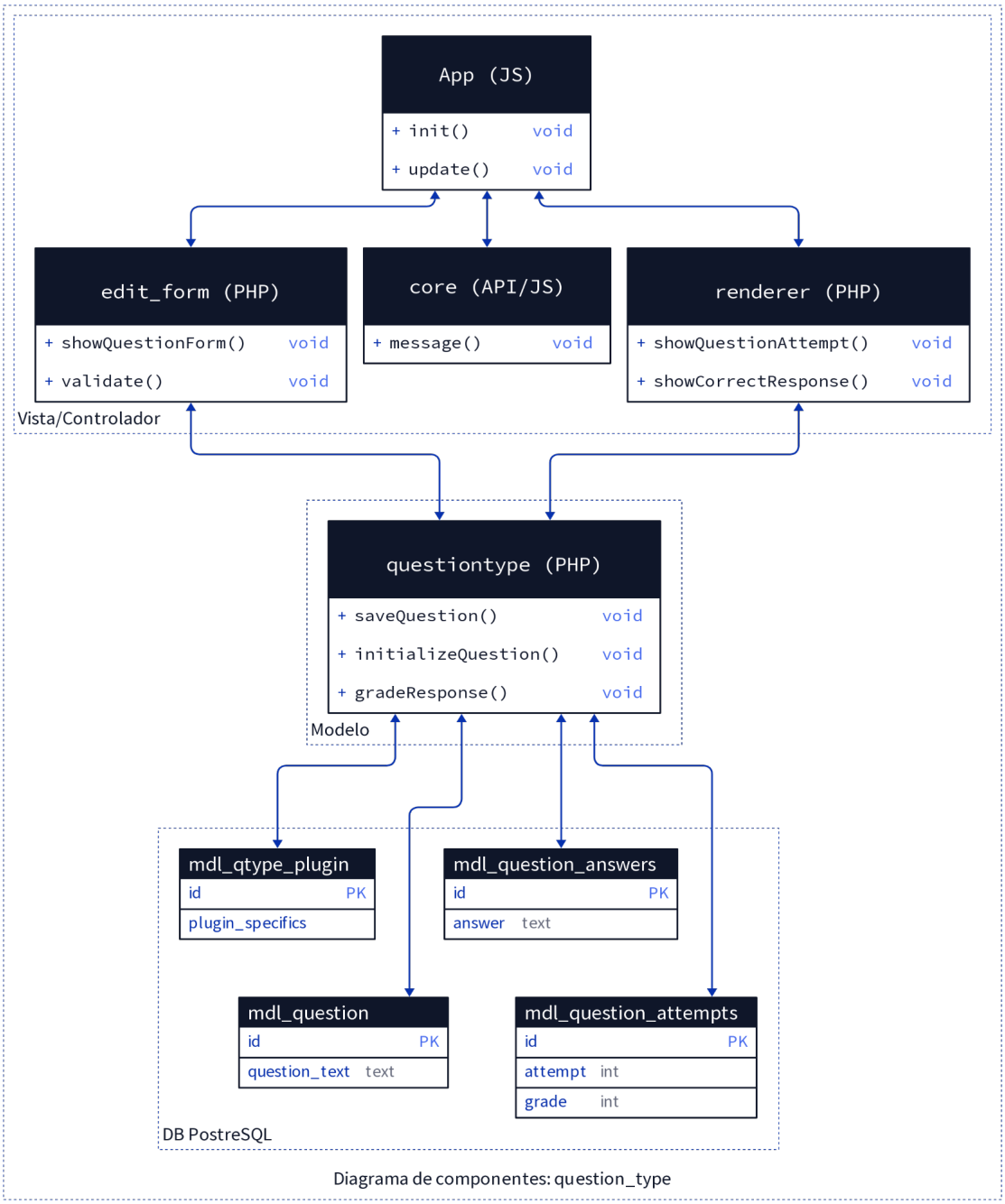
Se desarrolló la plantilla de complemento de tipo “pregunta”, ilustrada en la Figura 3, la cual integra aplicaciones JavaScript de propósito general. Sus componentes principales son:

- *App*: encapsula la aplicación JavaScript de propósito general. Se comunica con la API de JavaScript de Moodle y los componentes de vista, manteniendo la coherencia de la información.
- *edit_form*: componente PHP (vista/controlador) para crear, guardar y editar las preguntas en el banco de Moodle.
- *renderer*: componente PHP (vista/controlador) que presenta la pregunta al estudiante, gestiona el intento de respuesta y muestra la retroalimentación.
- *questiontype*: componente PHP del modelo que sirve como interfaz con la capa de persistencia.
- *DB*: Capa de persistencia donde se almacena la información de preguntas, respuestas e intentos.

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 71 - 78

Figura 3

Componentes principales de la plantilla del complemento de tipo pregunta



La relevancia de esta plantilla radica en que su componente App admite módulos de Node.js y se compila con Vite, lo que optimiza el flujo de desarrollo (Lucas *et al.*, 2025). Esta plantilla, compatible con React, se utilizó como base para los simuladores de hoja de cálculo y búsqueda en Internet.

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 72 - 78

3.3 SIMULADOR DE HOJA DE CÁLCULO

Con la plantilla descrita, se implementó la nueva versión del simulador de hoja de cálculo, cuya arquitectura se presenta en la Figura 4. Este complemento integra React para la interfaz, utiliza FortuneSheet para la manipulación interactiva de la hoja y accede a la API de Moodle para gestionar la internacionalización.

Figura 4

Arquitectura del simulador de hoja de cálculo

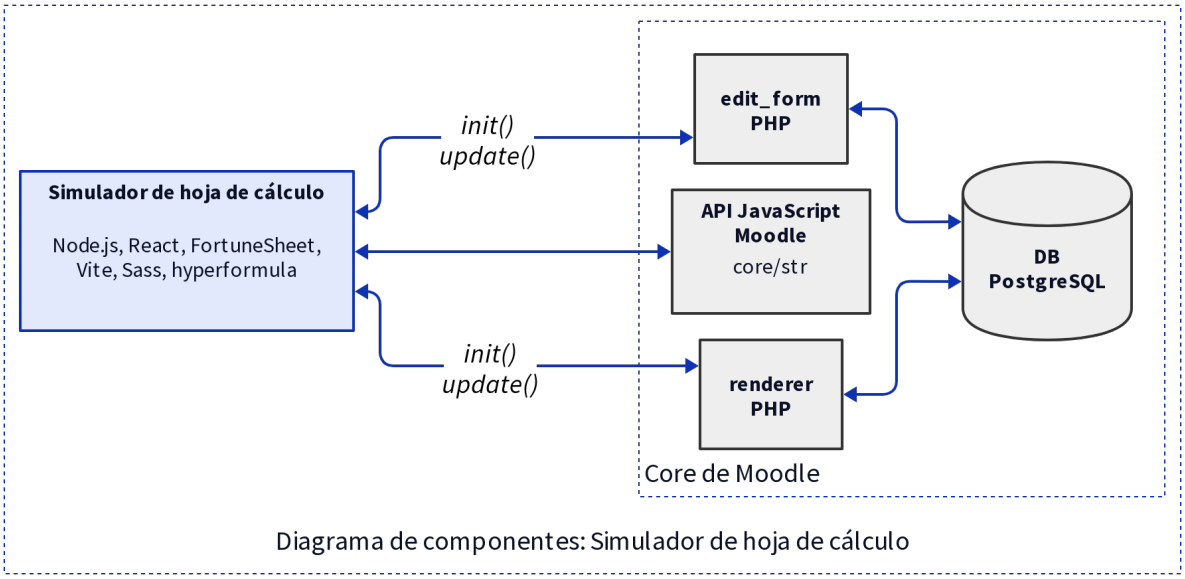


Diagrama de componentes: Simulador de hoja de cálculo

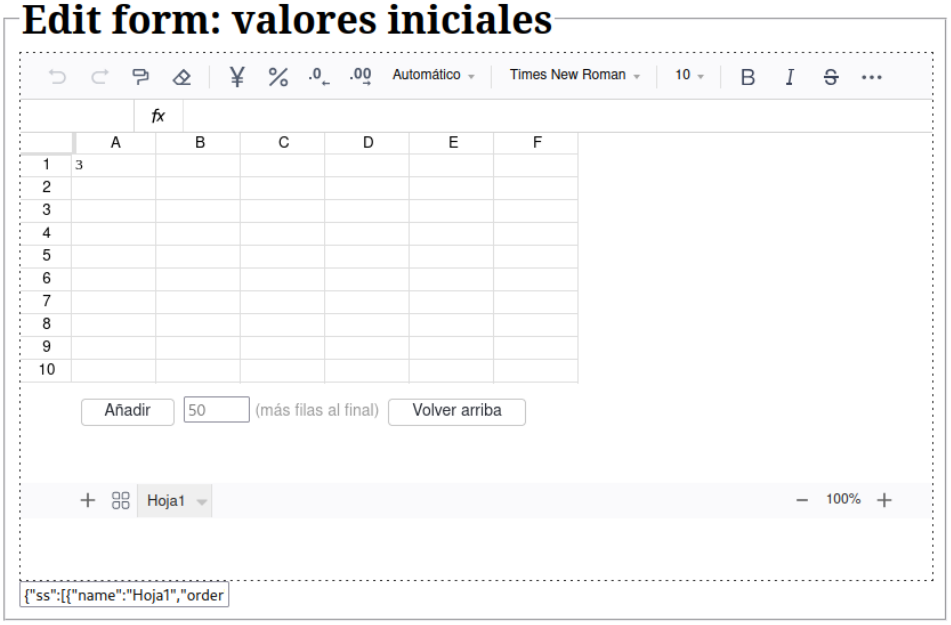
Los componentes *edit_form* y *renderer* de Moodle gestionan las instancias de la hoja de cálculo y almacenan los datos en la base de datos. Se definieron cuatro vistas (instancias) específicas:

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 73 - 78

1. Vista de valores iniciales (*edit_form*): ilustrada en la Figura 5, permite configurar el estado inicial de la hoja que verá el estudiante.

Figura 5

Vista de valores iniciales en la configuración del componente

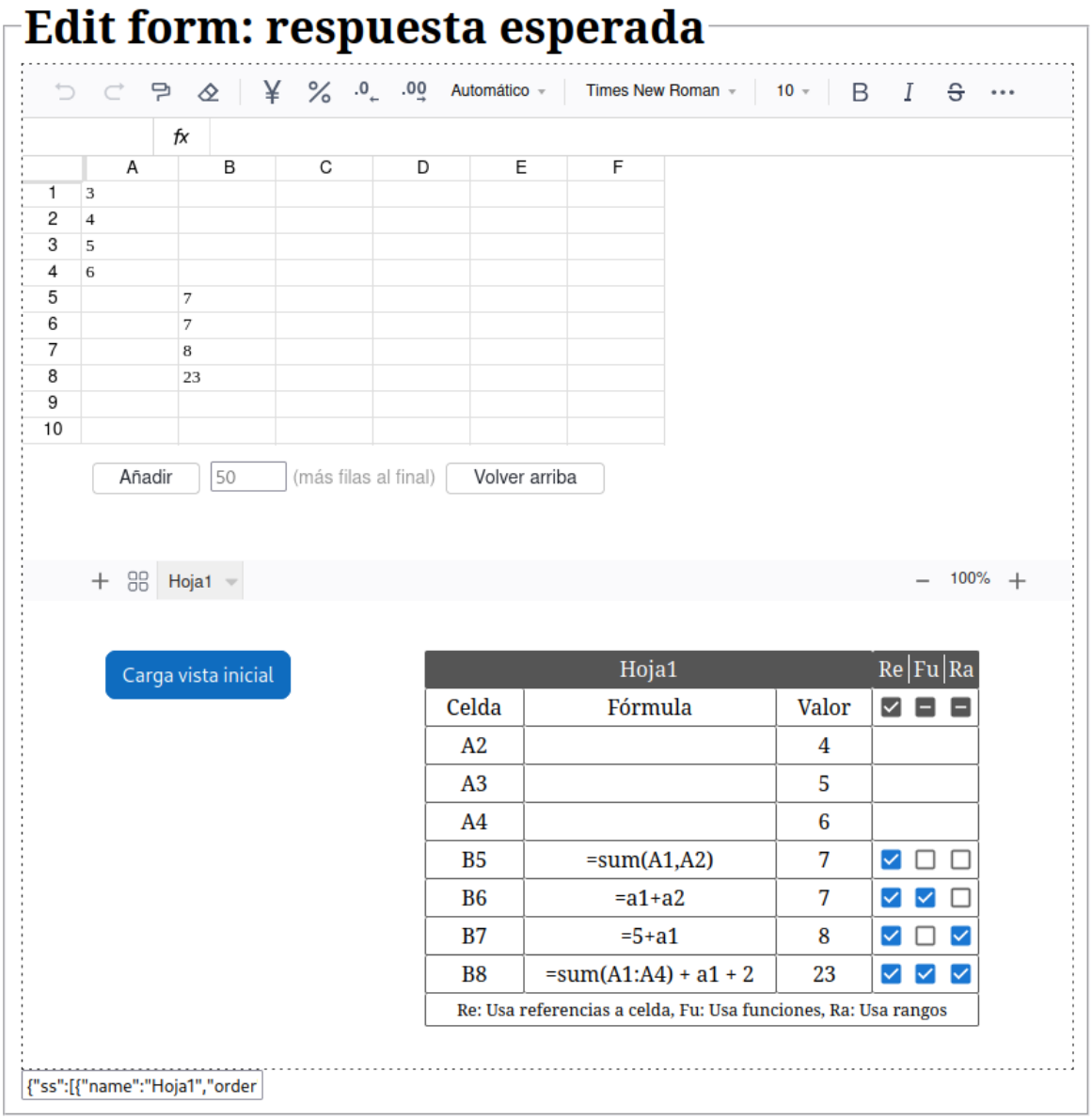


2. Vista de respuesta esperada (*edit_form*): como se detalla en la Figura 6, configura la respuesta correcta e incluye opciones para definir qué elementos se evaluarán específicamente por cada celda (referencias, rangos o funciones).

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 74 - 78

Figura 6

Vista de respuesta esperada en la configuración del componente

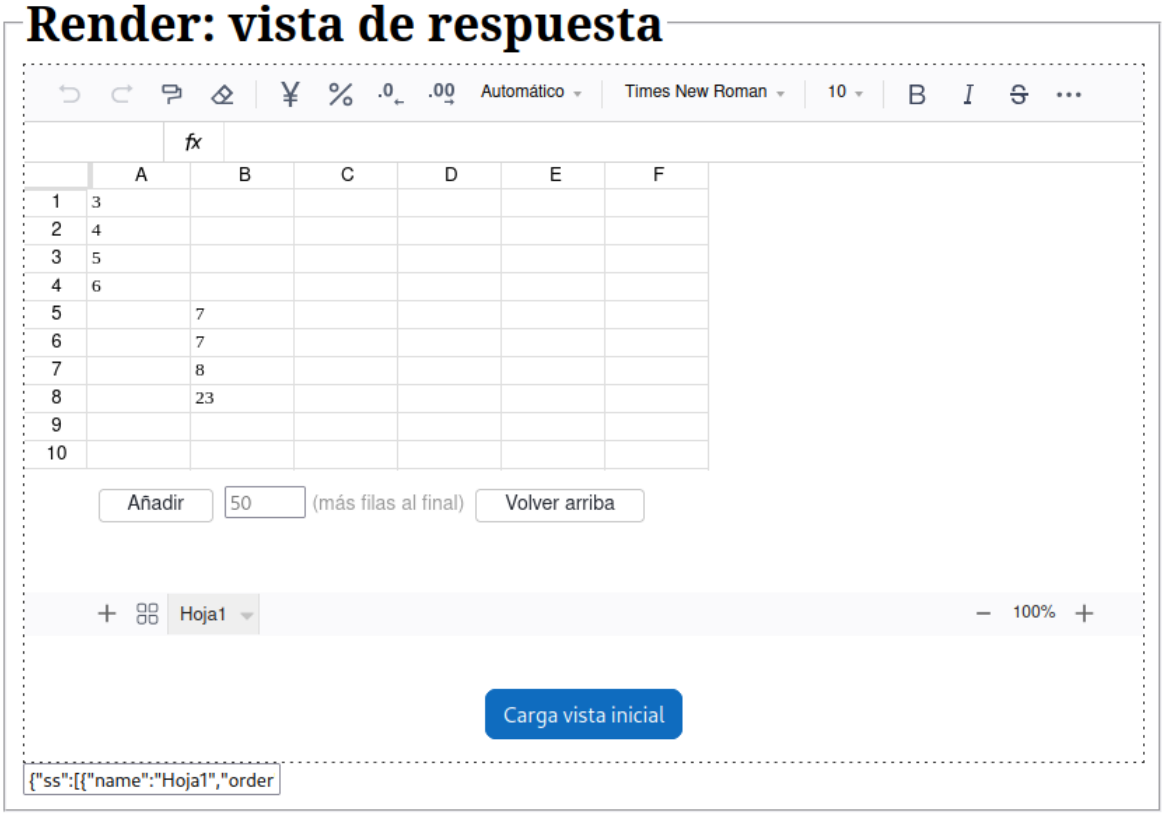


https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 75 - 78

3. Vista de respuesta (*renderer*): presentada en la Figura 7, es la interfaz que ve el estudiante al responder el examen, la cual carga automáticamente el estado inicial configurado.

Figura 7

Vista de respuesta del estudiante durante la evaluación

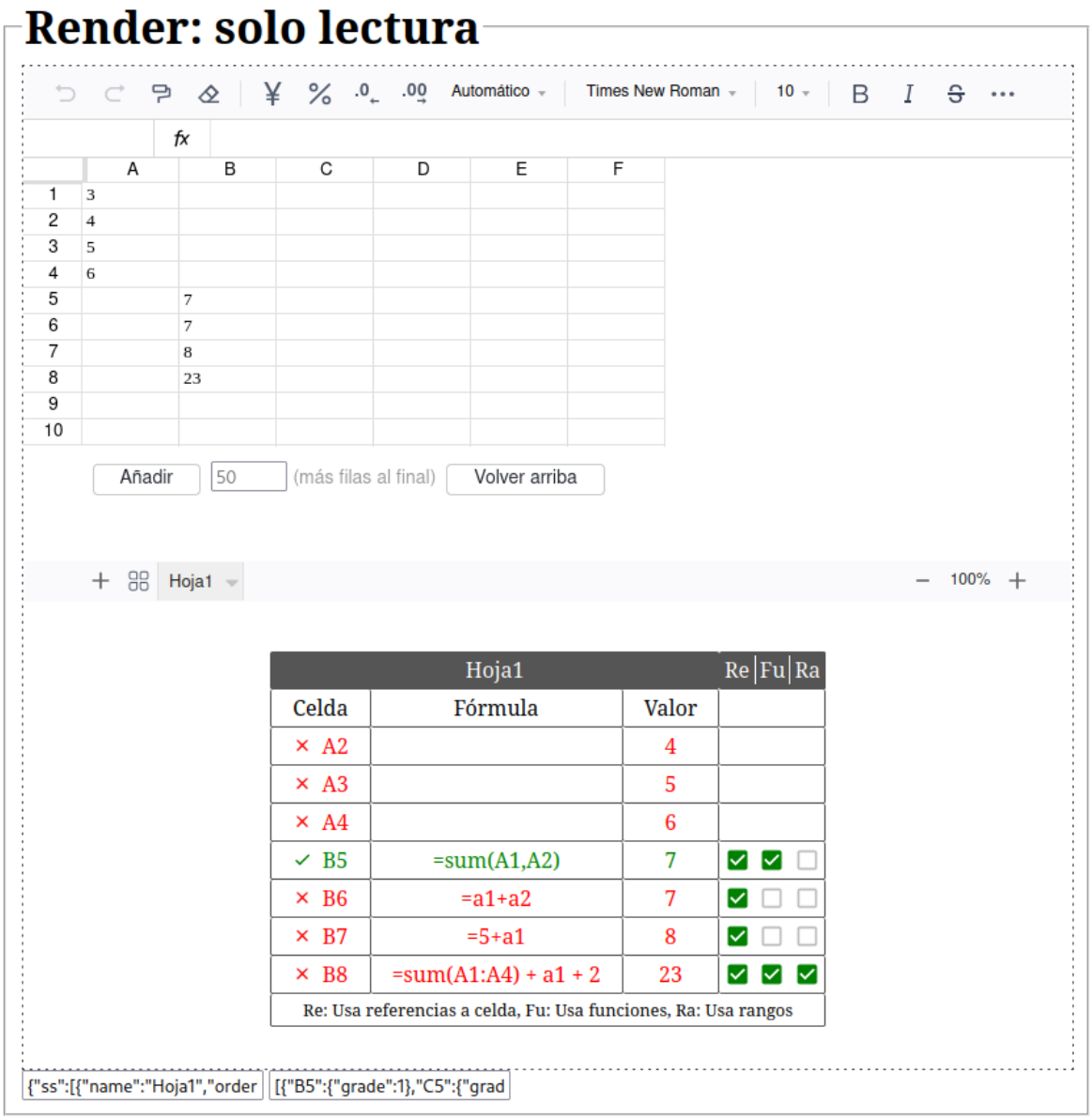


https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 76 - 78

4.Vista de sólo lectura (*renderer*): mostrada en la Figura 8.

Figura 8

Vista de sólo lectura con retroalimentación de la evaluación



Esta vista se presenta al estudiante a modo de retroalimentación, indicando cuáles celdas fueron correctas y cuáles incorrectas.

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 77 - 78

3.3.1 MECANISMO DE EVALUACIÓN

Una parte clave fue la implementación de un mecanismo de evaluación granular (Pordanjani & Salehi, 2025). La respuesta dada (RD) se califica de 0 a 1, comparándola con la respuesta esperada (RE). Para cada celda modificada, la calificación es correcta (1) sólo si cumple todas las condiciones preconfiguradas, es decir:

- El valor de la celda en RD es igual al valor en RE.
- Si en RE se usa fórmula, en RD también.
- Si en RE se usan referencias a celdas, en RD también.
- Si en RE se usan funciones, en RD también.
- Si en RE se usan rangos, en RD también.

Cada celda puede obtener un máximo de 5/5; si el valor es distinto, la calificación es 0.

4. CONCLUSIONES

El proyecto surgió de una doble necesidad crítica: por un lado, la obsolescencia técnica de los simuladores del TICómetro, que impedía su funcionamiento en Moodle 4.3 y la correcta evaluación de habilidades TIC; y por otro, la falta de un proceso de desarrollo estandarizado en el área responsable del servicio, donde la diversidad de entornos dificultaba la colaboración y el mantenimiento.

Los objetivos se lograron mediante el trabajo colaborativo de un grupo interdisciplinario de personas académicas. La implementación de un entorno estándar de desarrollo ágil y fácilmente replicable, así como el diseño de una plantilla de complemento reutilizable, resolvieron la problemática. Sobre esta base, se actualizó tecnológicamente el simulador de hoja de cálculo, dotándolo de un sistema de evaluación granular. Esto valida la nueva arquitectura como una solución robusta y mantenible, disponible para futuros desarrollos.

Actualmente, se ha adoptado esta plantilla y el entorno *moodle-docker* (Moodle HQ, s. f.) como el estándar para nuevos simuladores (edición de video y de audio). Asimismo, esta infraestructura se utiliza para concluir la actualización del simulador de búsqueda en Internet y se continúa con el trabajo para la publicación de estas herramientas en el repositorio oficial de la comunidad (Moodle, s.f.).

Aunque la adopción de la arquitectura basada en contenedores ha demostrado una mejora cualitativa significativa en la agilidad del trabajo diario, una futura línea de análisis podría definir y cuantificar estas métricas de productividad. Por otro lado, la nueva versión del simulador de hoja de cálculo fue aplicada a alumnos de nuevo ingreso a finales de 2025. En 2026 el equipo pedagógico se encuentra analizando los datos recopilados. Los resultados cuantitativos de esta aplicación y su comparativa con versiones anteriores del simulador serán objeto de un trabajo futuro para validar estadísticamente las mejoras en la evaluación. Este proyecto va más allá de sólo actualizar una herramienta de evaluación. A nivel institucional, sienta las bases para crear de forma ágil y colaborativa futuros recursos educativos dentro de la Universidad. Además, a un nivel más global, la meta de publicar estos complementos en la comunidad de Moodle

https://doi.org/10.22201/dgtic.30618096e.2026.4.2.169 Vol. 4, Núm. 2. abril-junio 2026, págs. 78 - 78

abre la puerta para que otras instituciones superen las limitaciones de la plataforma y puedan evaluar habilidades digitales con mayor precisión.

AGRADECIMIENTOS

Agradezco a todo el equipo de trabajo de la Dirección de Innovación en Tecnologías para la Educación de la DGTIC-UNAM, cuyas discusiones dieron pie a una solución robusta para los retos que presentó este proyecto: María Ramírez Bedolla, Mónica Avila Quintana, Adriana Areli Bravo Lozano y María Elizabeth Martínez Sánchez, Miguel Zuñiga González, Francisco Isaac Moguel Pedraza y Javier Rodrigo Díaz Espinosa.

REFERENCIAS

Juca-Maldonado, A. X., García-Vera, J. S., Juca Maldonado, F., & Carrión González, J. T. (2025). Competencias tecnológicas de la generación Z: Una evaluación cuantitativa de las habilidades frente a la percepción de los nativos digitales. *Revista Transdisciplinaria de Estudios Sociales y Tecnológicos (RTEST), 5*(2), 16–24. https://doi.org/10.58594/rtest.v5i2.160

López-Mora, D., Villamar Coloma, M., Bravo-Pino, Á., & Lozano-Rodríguez, E. (2019). El uso de las metodologías ágiles y su importancia para el desarrollo de software. *Killkana Técnica, 3*(1), 23–28. https://doi.org/10.26871/killkana_tecnica.v3i1.473

Lucas, W., Nunes, R., Bonifácio, R., Carvalho, F., Lima, R., Silva, M., Torres, A., Accioly, P., Monteiro, E., & Saraiva, J. (2025). Understanding the adoption of modern JavaScript features: An empirical study on open-source systems. *Empirical Software Engineering*, *30*, Article 107. https://doi.org/10.1007/s10664-025-10663-9

Moodle. (s.f.). *Moodle*. https://moodle.org/

Moodle HQ. (s. f.). *moodle docker.* GitHub. https://github.com/moodlehq/moodle-docker

Pordanjani, Z., & Salehi, K. (2025). Limitations of electronic assessment: A systematic review. *Qualitative Research*, *3*(1), 111–130. https://doi.org/10.15157/qr.2025.3.1.111-130

Ruilisi. (s. f.). *fortune-sheet.* GitHub. https://github.com/ruilisi/fortune-sheet

Universidad Nacional Autónoma de México. (s. f.). *TICómetro*. Educatic. https://ticometro.educatic.unam.mx

Wang, W. (2022). Research on using Docker container technology to realize rapid deployment environment on virtual machine. En *2022 8th Annual International Conference on Network and Information Systems for Computers (ICNISC)*, (pp. 541–544). IEEE. https://doi.org/10.1109/ICNISC57059.2022.00112

Wikipedia contributors. (2025). *Asynchronous module definition*. Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/wiki/Asynchronous_module_definition