Prototipo de tutor virtual para entonos inmersivos 3d. Caso de estudio: Carrera de Software de la ESPE en Secondlife

Monga Aguirre, Juan Sebastian

Departamento de Ciencias de la Computación

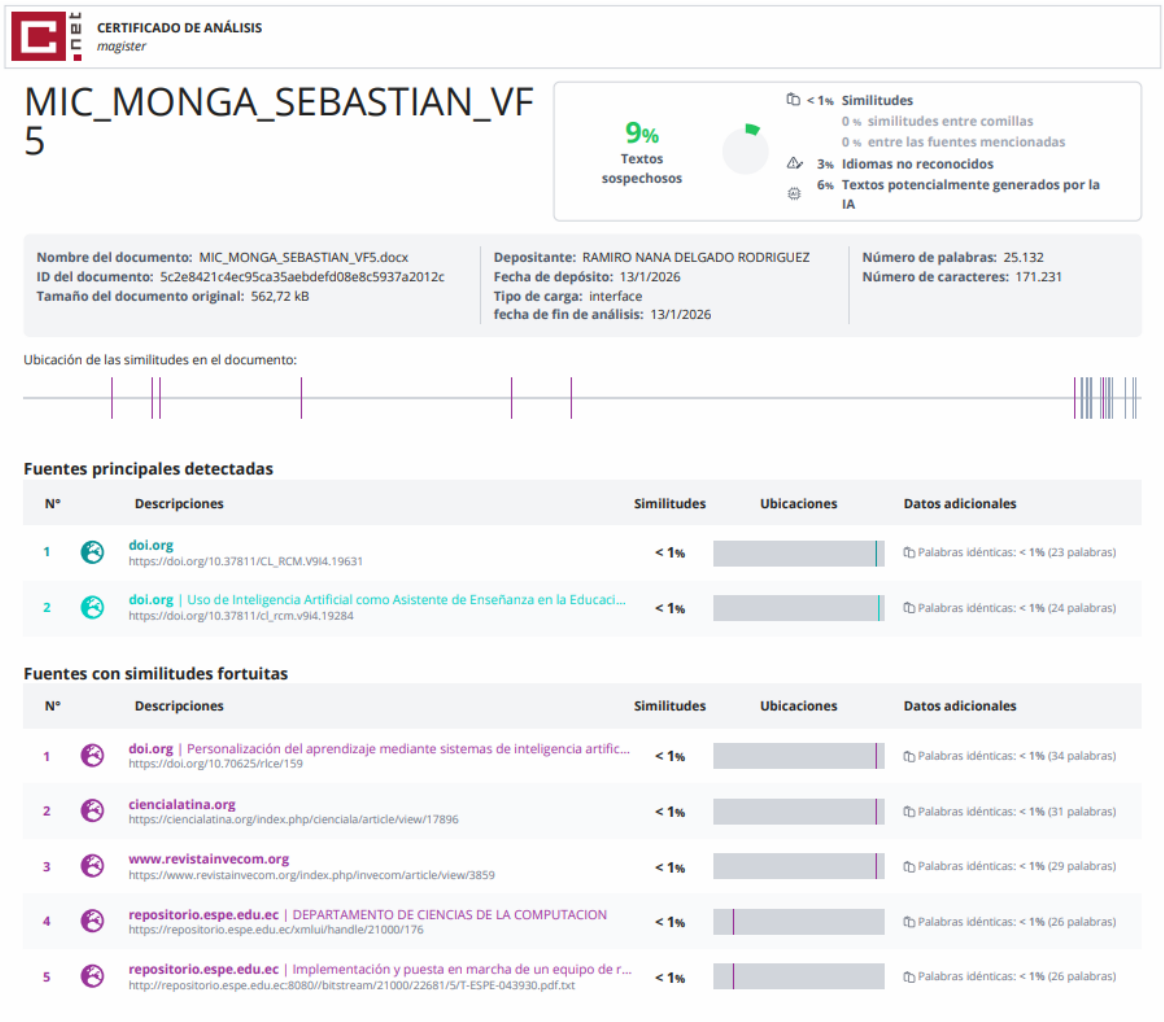
Carrera de Software

Trabajo de integración curricular, previo a la obtención del título de Ingeniero de Software

Ing. Delgado Rodríguez, Ramiro Nanac, PhD

5 de marzo de 2026

Análisis de Similitud de Contenidos



CERTIFICADO DE ANÁLISIS
magister

MIC_MONGA_SEBASTIAN_VF5

9% Textos sospechosos

- < 1% Similitudes
  - 0 % similitudes entre comillas
  - 0 % entre las fuentes mencionadas
- 3% Idiomas no reconocidos
- 6% Textos potencialmente generados por la IA

Nombre del documento: MIC_MONGA_SEBASTIAN_VF5.docx
ID del documento: 5c2e8421c4ec95ca35aebdefd08e8c5937a2012c
Tamaño del documento original: 562,72 kB

Depositante: RAMIRO NANA DELGADO RODRIGUEZ
Fecha de depósito: 13/1/2026
Tipo de carga: interface
fecha de fin de análisis: 13/1/2026

Número de palabras: 25.132
Número de caracteres: 171.231

Ubicación de las similitudes en el documento:

Fuentes principales detectadas

N°	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	doi.org https://doi.org/10.37811/CL_RCM.V9I4.19631	< 1%		Palabras idénticas: < 1% (23 palabras)
2	doi.org | Uso de Inteligencia Artificial como Asistente de Enseñanza en la Educaci... https://doi.org/10.37811/cl_rcm.v9i4.19284	< 1%		Palabras idénticas: < 1% (24 palabras)

Fuentes con similitudes fortuitas

N°	Descripciones	Similitudes	Ubicaciones	Datos adicionales
1	doi.org | Personalización del aprendizaje mediante sistemas de inteligencia artific... https://doi.org/10.70625/rlce/159	< 1%		Palabras idénticas: < 1% (34 palabras)
2	ciencialatina.org https://ciencialatina.org/index.php/cienciala/article/view/17896	< 1%		Palabras idénticas: < 1% (31 palabras)
3	www.revistainvecom.org https://www.revistainvecom.org/index.php/invecom/article/view/3859	< 1%		Palabras idénticas: < 1% (29 palabras)
4	repositorio.espe.edu.ec | DEPARTAMENTO DE CIENCIAS DE LA COMPUTACION https://repositorio.espe.edu.ec/xmlui/handle/21000/176	< 1%		Palabras idénticas: < 1% (26 palabras)
5	repositorio.espe.edu.ec | Implementación y puesta en marcha de un equipo de r... http://repositorio.espe.edu.ec:8080//bitstream/21000/22681/5/T-ESPE-043930.pdf.txt	< 1%		Palabras idénticas: < 1% (26 palabras)

Firmado electrónicamente por:
RAMIRO NANAC DELGADO RODRIGUEZ
Validar únicamente con FirmaEC

Ing. Delgado Rodríguez, Ramiro Nanac, PhD

Director

Departamento de Ciencias de la Computación

Carrera de Software

Certificación

Certifico que el trabajo de integración curricular: "**Prototipo de tutor virtual para entonos inmersivos 3d. Caso de estudio: Carrera de Software de la ESPE en Secondlife**" fue realizado por el señor **Monga Aguirre, Juan Sebastian,** el mismo que cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, además fue revisado y analizada en su totalidad por la herramienta de prevención y/o verificación de similitud de contenidos; razón por la cual me permito acreditar y autorizar para que se lo sustente públicamente.

Sangolquí,5 de marzo de 2026

Firmado electrónicamente por:
RAMIRO NANAC DELGADO RODRIGUEZ
Validar únicamente con FirmaEC

Ing. Delgado Rodríguez, Ramiro Nanac, PhD

Director

C.C.: 1707019178

Departamento de Ciencias de la Computación

Carrera de Software

Responsabilidad de Autoría

Yo, **Monga Aguirre, Juan Sebastian**, con cédula de ciudadanía n°1722194147, declaro que el contenido, ideas y criterios del trabajo de integración curricular**: Prototipo de tutor virtual para entonos inmersivos 3d. Caso de estudio: Carrera de Software de la ESPE en Secondlife** es de mi/nuestra autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí,5 de marzo de 2026

Monga Aguirre, Juan Sebastian

C.C.: 1722194147

Departamento de Ciencias de la Computación

Carrera de Software

Autorización de Publicación

Yo **Monga Aguirre, Juan Sebastian**, con cédula de ciudadanía n°1722194147, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de integración curricular: **Título: Prototipo de tutor virtual para entonos inmersivos 3d. Caso de estudio: Carrera de Software de la ESPE en Secondlife** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí,5 de marzo de 2026

Juan Sebastian Monga Aguirre
Time Stamping Security Data

Monga Aguirre, Juan Sebastian

C.C.: 1722194147

Dedicatoria

Dedico el presente trabajo a mi madre Martha quien me impulso a seguir adelante pese a los momentos difíciles. Su amor y apoyo incondicional fueron mi motor durante mi formación académica.

A mi padre Fernando quien me apoyo económicamente y me inspiro y me sirvió de ejemplo para poder terminar mi formación académica. A Anahí quien me apoyo desde antes de comenzar la carrera y me ha ido apoyando hasta poder alcanzar mi meta, Finalmente dedico a Coco, Luna, Emilio, Ignacia que tambien me han servido de inspiración para superar cualquier obstáculo a lo largo de mi experiencia universitaria. Todos y cada uno de los mencionados en esta dedicatoria representan una parte importante en mi vida, porque creyeron en mí y me motivaron a alcanzar mis metas académicas. Este trabajo va dedicado a todos ustedes con mucho cariño porque este logro es tambien de ustedes.

Agradecimiento

Agradezco en primer lugar a Dios el poder haber culminado esta etapa académica, por brindarme fortaleza para seguir adelante. Agradezco tambien a mi madre Martha quien me apoyo a lo largo de este trayecto, siempre estuvo ahí siempre creyó en mí, a mi padre Fernando que fue la persona que me inspiro y por la que me incline por el mundo de la tecnología, tambien agradezco a Anahí por aconsejarme y porque siempre estuvo animándome cuando no me salían bien las cosas.

De manera muy especial agradezco a mi tutor ING Ramiro Delgado, por su orientación y acompañamiento académico durante todo este proceso, le agradezco infinitamente por las herramientas facilitadas para poder realizar el presente trabajo.

Agradezco a todos y cada uno de los ingenieros con los que pude coincidir durante mi estancia en la universidad, gracias por compartir sus conocimientos que contribuyeron de gran forma a mi formación profesional.

Finalmente agradezco a mi mejor amigo en la Universidad, Kevin Guzmán que me apoyaba y ayudaba en clases.

Gracias a cada uno de ustedes por formar parte de esta linda etapa en la Universidad.

Índice de contenidos

Índice de tablas

Índice de figuras

Resumen

El trabajo se centró en el diseño y desarrollo de un prototipo de tutor virtual para entornos tridimensionales inmersivos, destinado a mejorar la experiencia educativa y el apoyo académico en la educación superior. tomando como caso de estudio la carrera del Software del Departamento de Ciencias Informáticas de la Universidad de las Fuerzas Armadas – ESPE. El problema que se vio es la falta de sistemas completos de soporte en espacios 3D, donde los estudiantes están tratando de navegar por áreas complejas sin una guía académica o informativa adecuada para ayudarles a comprender el contexto institucional y el proceso de aprendizaje. El objetivo principal era crear un prototipo de tutor virtual utilizando una configuración de microservicio que combina dos enfoques de mentoría: un tutor para el aprendizaje guiado basado en clases y otro que brinda información sobre el departamento, se desarrolló bajo un enfoque de ingeniería de software con carácter aplicado, utilizando un proceso iterativo e incremental basado en Sprints, que permitió el análisis de los requisitos según la norma IEEE 830, diseño arquitectónico distribuido, desarrollo de componentes y validación funcional del sistema. El prototipo se implementó en la plataforma Second Life, el cual para la comunicación con los servicios backend, integra un componente puente, además de un servicio encargado de la orquestación y microservicios especializados para cada tipo de tutoría, apoyado por persistencia relacional y recuperación semántica.

Palabras clave: tutor virtual, entornos inmersivos 3D, Second Life, microservicios, educación superior

Abstract

The work focused on the design and development of a virtual tutor prototype for immersive three-dimensional environments, aimed at enhancing the educational experience and academic support in higher education, using the Software career of the Computer Science Department at the University of the Armed Forces – ESPE as a case study. The problem identified is the lack of comprehensive support systems in 3D spaces, where students are trying to navigate complex areas without adequate academic or informational guidance to help them understand the institutional context and learning process. The main objective was to create a virtual tutor prototype using a microservice configuration that combines two mentoring approaches: one tutor for class-based guided learning and another providing information about the department. It was developed under a software engineering approach with an applied character, using an iterative and incremental process based on Sprints, which allowed for the analysis of requirements according to the IEEE 830 standard, distributed architectural design, component development, and functional validation of the system. The prototype was implemented on the Second Life platform, which, for communication with backend services, integrates a bridge component, as well as a service responsible for orchestration and specialized microservices for each type of tutoring, supported by relational persistence and semantic retrieval.

Keywords: virtual tutor, immersive 3D environments, Second Life, microservices, higher education

Capítulo I Introducción

El cambio digital en la enseñanza universitaria ha impulsado la introducción gradual de ambientes virtuales avanzados como apoyo a las formas de enseñar y aprender. En este marco, la educación a distancia ha pasado de usar sistemas viejos para manejar materiales a crear espacios interactivos y envolventes que buscan no solo imitar, sino también expandir y enriquecer lo que se vive en persona. Los espacios virtuales tridimensionales son un claro ejemplo de esta evolución, ya que posibilitan simular aulas, permitir el trato directo y que el alumno participe activamente en mundos virtuales dinámicos y muy visuales.

La incorporación de estas tecnologías envolventes en las universidades responde tanto a tendencias tecnológicas mundiales como a la necesidad de reforzar métodos enfocados en que el alumno aprenda por sí mismo, activamente y con sentido. En estos sitios, el estudiante deja de solo recibir datos y se vuelve alguien que explora, participa y crea saber mediante la vivencia. No obstante, lo complicado de los espacios en tres dimensiones requiere añadir ayudas para guiar a quien usa el sistema, hacer más fácil entender el lugar virtual y ayudar a lograr las metas educativas y de la institución.

Ante esto, los tutores virtuales inteligentes surgen como una opción para dar soporte constante dentro de estos mundos inmersivos. Estos asistentes digitales pueden ayudar al usuario al instante, contestar dudas, mostrarle caminos y adaptar la información según las distintas necesidades. Su valor aumenta en lugares virtuales complejos, donde no tener una guía clara puede dañar el aprendizaje, causar confusión y limitar el uso provechoso de los recursos disponibles.

En el ambiente universitario, la labor de un tutor virtual va más allá del apoyo puramente académico. En el ámbito de la institución, es igual de importante dar ayuda informativa para que el usuario entienda el ambiente universitario, sepa quién manda, conozca la organización, pueda hacer gestiones comunes y halle los servicios útiles. En los mundos virtuales 3D, esta

parte de dar datos es clave, pues el moverse e investigar en el espacio virtual necesita referencias claras para ayudar a navegar y entender el contexto de la institución.

Desde este punto de vista, este Trabajo de Integración Curricular se enfoca en crear un modelo inicial de tutor virtual para los espacios 3D en la plataforma Second Life, usando como ejemplo la carrera de Software del área de Ciencias de la Computación de la Universidad de las Fuerzas Armadas – ESPE. El modelo incluye de manera explícita un sistema de guía doble: un guía de materias, enfocado en ayudar con lo académico durante el proceso de enseñanza y aprendizaje, y un guía de información, centrado en dar ayuda institucional y de contexto sobre el ambiente de estudios del Departamento de Ciencias de la Computación. Ambos tutores trabajan juntos en el mismo espacio inmersivo con papeles distintos pero que se complementan, ayudando unidos a mejorar lo que vive el usuario y a reforzar la enseñanza mutua.

Antecedentes

En los últimos años, los entornos 3D inmersivos han captado bastante atención en el ámbito educativo porque pueden simular escenarios complicados y estimular el aprendizaje basado en la experiencia. Diversas investigaciones mostraron que estos entornos podrían motivar más, incentivar la participación activa y mejorar la comprensión del tema, particularmente en áreas como ingeniería, informática y ciencias aplicadas. Sin embargo, su éxito depende, en gran medida, de cómo se integran pedagógica y tecnológicamente, también de la existencia de mecanismos de soporte que guíen al usuario durante la interacción.

De forma parecida, el desarrollo de tutores virtuales y agentes inteligentes ha tenido avances importantes, permitiendo su aplicación en plataformas educativas, con el propósito de ofrecer retroalimentación, dirigir los procesos de aprendizaje, y promover la autonomía del estudiante. Aunque estas soluciones se han investigado mucho en entornos virtuales convencionales, su integración en espacios 3D inmersivos aún representa un desafío, sobre

todo en cuanto a la conexión entre la interacción tridimensional y los servicios que gestionan la lógica del tutor.

La arquitectura de microservicios verdaderamente ha emergido cual una estrategia fantástica para crear sistemas complicados, ya que impulsa la modularidad, aparte de la separación de deberes, y la compatibilidad entre piezas. En el contexto de los tutores virtuales, este acercamiento deja organizar separadamente funciones, tales como tutoría escolar y proporcionar información de la institución, entonces facilitando su integración dentro de un ámbito inmersivo.

Second Life, se estableció como una de las plataformas más utilizadas para experimentar en la educación en entornos 3D inmersivos, debido a su flexibilidad, aptitud de personalizar, y su adopción previa en ámbitos escolares. Su utilización hace posible reinventar lugares universitarios, lanzar escenarios interactivos y evaluar la interacción entre individuos y agentes virtuales en un ambiente controlado, por eso, es una plataforma perfecta para desarrollar esta tarea.

Planteamiento del problema

A pesar de las oportunidades educativas que ofrecen los entornos 3D inmersivos, una de sus principales desventajas es la falta de sistemas de apoyo integral que ayuden al usuario durante su interacción con el ambiente. En numerosas ocasiones, los estudiantes tienen que navegar por complejos espacios virtuales sin la ayuda necesaria para entender los objetivos de aprendizaje, moverse por el entorno o utilizar correctamente los recursos disponibles.

Este reto no se limita solamente a la asistencia educativa. En escenarios inmersivos de nivel universitario, también se muestra la carencia de información institucional clara y accesible que ayude al usuario a entender el contexto académico, identificar la estructura organizativa, conocer los trámites comunes o aclarar dudas generales sobre el funcionamiento del departamento o la carrera. La ausencia de una información puede causar confusión, falta de interés y zona incompleta en un mundo virtual.

Dentro del marco de la Carrera de Software del Departamento de Ciencias de la Computación de la ESPE, la usencia de tecnologías como los entornos inmersivos y la no existencia de un sistema de tutoría que brinde guía académica e información institucional da como resultado que el proceso de enseñanza se limite únicamente a tecnologías tradicionales. La experiencia de los estudiantes y la progresiva integración de estos entornos en la enseñanza se ven afectadas. Lo que se propone en este sentido es el desarrollo de un prototipo de tutor virtual que integre un tutor académico y un tutor informativo que orienten, acompañen, contextualicen y apoyen al usuario en espacios 3D inmersivos para mejorar la experiencia de aprendizaje, así como promover una educación autónoma, guiada y contextualizada.

Justificación

Desde la perspectiva académica, este estudio ayuda a examinar y aplicar sistemas de tutoría en línea dentro de ambientes educativos inmersivos, explorando un área de investigación que gana importancia y aún presenta escasa implementación práctica. La propuesta brinda la oportunidad de investigar un modelo de tutoría dual que combina apoyo académico y asesoría institucional, ofreciendo una comprensión más completa del papel de los tutores en línea en las universidades.

En el campo tecnológico, el proyecto es relevante al implementar conceptos de arquitectura de microservicios para crear un sistema modular que permita la convivencia de diferentes tipos de tutoría en un mismo espacio inmersivo. Este método favorece la escalabilidad, la interoperabilidad y el desarrollo futuro del prototipo, alineándose con las mejores prácticas en ingeniería de software.

Desde un enfoque práctico, el tutor de información agrega un valor considerable al optimizar la navegación, asesorar y mejorar la comprensión del entorno académico dentro del espacio inmersivo, mientras que el tutor de clases refuerza el apoyo pedagógico. La combinación de ambos tutores enriquece la experiencia del estudiante, haciéndola más

coherente y autónoma, fomentando así el uso educativo de los entornos 3D en la Carrera de Software de la ESPE.

Objetivo General

Desarrollar un prototipo de tutor virtual para entornos inmersivos 3D, basado en una arquitectura de microservicios, que integre un esquema de tutoría dual conformado por un tutor de clases orientado al acompañamiento académico guiado y un tutor de información institucional enfocado en el contexto del Departamento de Ciencias de la Computación de la Carrera de Software de la Universidad de las Fuerzas Armadas – ESPE**.**

Objetivos Específicos

Realizar una revisión sistemática de la literatura científica para la identificación de criterio, modelos y herramientas que se han utilizado para el desarrollo de tutores virtuales en entornos 3D inmersivos.

Diseñar la arquitectura del prototipo bajo el enfoque de microservicios, definiendo su estructura lógica y técnica.

Desarrolla el tutor de clase orientado al apoyo de orientación académica dentro del ambiente inmersivo 3D.

Desarrollar el tutor de información institucional que proporciona orientación contextual sobre la estructura, servicios y operaciones del departamento de ciencias de la computación.

Integrar ambos tutores dentro del entorno inmersivo 3D, evaluando su funcionamiento y aporte a la experiencia educativa.

Alcance

El trabajo actual se centra en la creación, elaboración y puesta en marcha de un prototipo operativo de un tutor virtual en un entorno tridimensional inmersivo, tomando como referencia la plataforma Second Life para la carrera de Software del Departamento de Ciencias de la Computación de la Universidad de las Fuerzas Armadas – ESPE. Este prototipo incluye

un modelo de tutoría que combina apoyo académico y orientación institucional, con el propósito de enriquecer la experiencia del usuario en el espacio virtual.

No se tiene en cuenta la creación de un producto comercial ni la inclusión de características avanzadas de adaptación inteligente. La evaluación del sistema se llevará a cabo a través de pruebas de funcionalidad y usabilidad en un entorno controlado, con el fin de analizar su viabilidad técnica y educativa, así como establecer una base para futuros desarrollos e investigaciones.

Capítulo II Marco teórico

Antecedentes internacionales

Gao et al. (2024) investigaron cómo los entornos inmersivos en 3D pueden tener avatares asistidos por educación donde los estudiantes pueden interactuar con un entorno histórico reconstruido digitalmente. Este estudio mostró cómo la interacción de avatares de IA usando lenguaje natural y su aprendizaje inmersivo, demostrando que los entornos 3D con Agentes Inteligentes pueden recrear experiencias educativas que de otro modo serían muy costosas en términos de viajes. Este documento presenta los avances técnicos del motor gráfico con los límites actuales y para mejorar el realismo y las interacciones en estos mundos virtuales

La evidencia más reciente también muestra que los entornos inmersivos han tenido impactos positivos en el aprendizaje en niveles más bajos. Así lo resume la revisión de un conjunto de artículos realizado por González et al. (2024), donde concluyen que el uso de simulaciones prácticas y el uso de 3D en los entornos de aprendizaje mejoran la educación y, sin duda, el uso de herramientas de aprendizaje del metaverso mejora la educación en la educación superior también. Sin embargo, aunque la evidencia apoya el uso de entornos 3D inmersivos, el uso de un diseño pedagógico apropiado es necesario, así como la disminución de la brecha de acceso y el uso de estas herramientas en un marco de políticas pedagógicas

de la institución. Se habla de que la evidencia de la educación superior es escasa en relación con el uso de estas herramientas.

Antecedentes nacionales

A nivel nacional, una revisión sistemática sobre entornos virtuales en la Educación Superior Ecuatoriana de Campoverde y Campoverde (2025) revela que estas plataformas digitales se están comenzando a analizar como instrumentos pedagógicos y que la especialización docente, la accesibilidad y las habilidades tecnológicas son elementos claves para su utilización. Hallazgos señalan que hay interés en incorporar tecnología inmersiva y simulaciones digitales en la enseñanza; sin embargo, se han identificado déficits en infraestructura y formación para su generalización en la nación

Recientemente, a nivel de posgrado, en Ecuador, Naranjo et al. (2025) realizaron un estudio sobre el uso de tecnologías, metodologías y prácticas de implementación de chatbots y sistemas inteligentes en la educación superior. Este estudio menciona que los chatbots de IA permiten una mayor personalización y eficiencia en el aprendizaje, aunque su efectividad depende de la infraestructura y formación docente. Los docentes tienen una postura positiva sobre el uso de estos sistemas y los autores mencionan su capacidad para la automatización de tutorías o el acompañamiento académico.

La investigación sobre IA aplicada a entornos virtuales de aprendizaje en educación superior en Ecuador de Toapanta et al. (2022), determinó que, para construir soluciones inmersivas y efectivas basadas en IA, resulta fundamental especificar las potencialidades de los entornos, las comunidades virtuales y la infraestructura TI. El estudio determinó que los prototipos con IA y simulaciones virtuales requieren de la integración de la IA con la tecnología pedagógica y un acompañamiento tecnológico que la sustente, para poder atender las demandas educativas del contexto ecuatoriano.

En conjunto, los estudios internacionales muestran que los agentes inteligentes en entornos virtuales tridimensionales mejoran la interacción, inmersión y aprendizaje en contextos

complejos, como la ingeniería y el desarrollo de software. Por lo tanto, los aportes mencionados constituyen una justificación para la elaboración de un tutor virtual inmersivo para la realidad educativa de la ESPE, Carrera de Software.

Concepto y evolución de los sistemas inteligentes educativos

Los sistemas inteligentes aplicados a la educación son herramientas informáticas que combinan la inteligencia artificial y enfoques pedagógicos con el fin de asistir, optimizar y personalizar los procesos educativos. Estos sistemas tienen la capacidad de simular el comportamiento del estudiante, procesar su rendimiento y ajustar de manera dinámica el contenido a sus diferentes niveles de complejidad cognitiva. Su propósito es generar mejores experiencias educativas que sean más interactivas y centradas en el estudiante. La evolución de los sistemas inteligentes en la educación comienza con el uso de tutoriales asistidos por computadores con un tipo de interactividad y retroalimentación que era lineal y bastante limitada (Arias et al., 2025).

Posteriormente, los avances en inteligencia artificial les permitieron a los computadores el uso de modelos cognitivos, de bases de conocimiento y de mecanismos de inferencia, lo que daba pie a los sistemas conocidos como tutores inteligentes. En años más recientes, la inteligencia de estos sistemas se vio potenciada por los avances en el aprendizaje automático y en el procesamiento del lenguaje natural. Hoy en día, estos sistemas son utilizados y se integran con tecnologías emergentes como la de los ambientes virtuales 3D y la de las arquitecturas distribuidas, que junto a las plataformas web han aumentado su potencial funcional y su alcance. Esta combinación de tecnologías ha permitido un mejor desarrollo de experiencias educativas, en especial en el nivel superior, las cuales se caracterizan por ser inmersivas y personalizadas (García et al., 2025).

Tipos de sistemas inteligentes en educación

Los sistemas educativos inteligentes tienen diferentes niveles de autonomía, interacción y adaptación al alumno, de modo que se distinguen los tutores inteligentes, los agentes

pedagógicos y los sistemas de aprendizaje adaptativos. Estos tipos abordan diferentes problemas educativos y tienen diferentes características de diseño y funcionalidad. Su implementación busca mejorar el proceso educativo ofreciendo un acompañamiento continuo y personalizado al alumno (Acevedo et al.,2025). A continuación, en la tabla 1 se presentan a detalle estos tipos:

Tabla 1

Tipos de sistemas inteligentes aplicados a la educación

Tipo de sistema inteligente	Definición	Características principales	Aplicación en educación superior
Tutores inteligentes	Sistemas que emplean inteligencia artificial para emular la función de un tutor humano para ayudar en el aprendizaje del alumno	• Consejo personal. • Modelamiento del Estudiante • Valuación de las prestaciones continuas • Modificación del contenido.	Asistencia en áreas como ingeniería y software que abarcan contenidos complejos, práctica guiada y refuerzo académico.
Agentes pedagógicos	Agentes virtuales que federalizan en el aula, que interactúan en un entorno digital con los alumnos, guían, motivan y ayudan en el proceso educativo.	• Interacción entre seres humanos y agentes. • Representación de avatares • Comunicarse utilizando palabras y lenguaje corporal. • Motivational approach	Ayudar en la implementación de entornos virtuales en 3D, simulaciones, aprendizaje inmersivo, etc.

Sistemas adaptativos de aprendizaje	El ritmo, actividades y contenidos de aprendizaje adaptados a partir del perfil del estudiante. Plataformas inteligentes.	• Ajuste automático. • Implementación de análisis de aprendizaje. • Se ajustará el nivel de dificultad • Monitoreo personal.	Optimización de entornos virtuales y plataformas educativas para el aprendizaje autodirigido y personalizado.

Nota. Matriz de los tipos de sistemas inteligentes que se aplican a la educación. Obtenido de (Acevedo et al., 2025)

Beneficios y limitaciones en educación superior

Sistemas inteligentes han traído cambios importantes en la enseñanza y el aprendizaje en la educación terciaria, sobre todo en entornos digitales y virtuales. Estas tecnologías mejoran la personalización del aprendizaje, la interacción entre estudiantes y docentes y el aprendizaje basado en la toma de decisiones. Sin embargo, la integración de estos sistemas presenta desafíos que deben ser considerados en la educación terciaria. De este modo, se hace necesario el análisis comparativo de las ventajas y de las desventajas que conllevan el uso de sistemas inteligentes (Chajuan et al., 2025). Contemplando el análisis de orden académico, se presenta la siguiente tabla.

Tabla 2

Beneficios y limitaciones de los sistemas inteligentes en educación superior

Dimensión	Beneficios	Limitaciones
Aprendizaje	• Personalización del proceso educativo • Retroalimentación inmediata y continua	• Dependencia tecnológica del estudiante • Posible sobreautomatización del aprendizaje

	• Mejora del aprendizaje autónomo	
Interacción educativa	• Mayor interacción estudiante-sistema • Acompañamiento permanente mediante tutores virtuales	• Interacción limitada en ausencia de diseños pedagógicos adecuados
Evaluación académica	• Evaluación continua basada en datos • Identificación temprana de dificultades de aprendizaje	• Riesgo de sesgos algorítmicos en la evaluación
Gestión institucional	• Optimización de recursos académicos • Escalabilidad de los servicios educativos	• Altos costos iniciales de implementación
Aspectos tecnológicos	• Integración con entornos virtuales y plataformas educativas • Uso de arquitecturas modernas y sistemas distribuidos	• Requerimientos de infraestructura y mantenimiento especializado
Formación docente	• Apoyo al rol del docente mediante analítica educativa	• Necesidad de capacitación continua del personal académico

Nota. Matriz de beneficios y limitaciones de los sistemas inteligentes en el ámbito de la educación superior. Obtenido de (Acevedo et al., 2025)

Tutores virtuales en entornos inmersivos 3D

Los tutores virtuales en entornos inmersivos 3D son la resultante de la integración de las tutorías en sistemas inteligentes y el desarrollo de tecnologías de realidad y mundos virtuales tridimensionales en las que los aprendizajes son facilitados en espacios interactivos que

simulan escenarios reales (Guamán et al., 2025). Estas plataformas brindan la posibilidad de vivir experiencias educativas que, sumadas a la presencia, interactividad y adaptabilidad a los entornos 3D enriquecidos con inteligencia artificial, resultan en escenarios educativos de alta versatilidad. Investigaciones recientes en educación superior han documentado el impacto positivo que el uso de estas plataformas en los estudiantes, a partir de la personalización y el control en el acceso a los contenidos, así como de la mejora sustancial en la motivación y el compromiso con el aprendizaje. Con lo cual, los tutores virtuales inmersivos resultan ser trascendentales a la hora de la innovación pedagógica en entornos educativos complejos (Filippone et al., 2025).

Definición de tutor virtual

Un agente virtual se comprende como un sistema automatizado capaz de brindar tutoría, asistencia y retroalimentación educativa a un estudiante, utilizando la inteligencia artificial para emular conductas y estilos de enseñanza de un docente. En ambientes tridimensionales, estos tutores se materializan como avatares o agentes que asisten a los alumnos en mundos virtuales tridimensionales, facilitando y personalizando itinerarios de aprendizaje. Su propósito es atenuar la fricción en los procesos de aprendizaje, estructurar itinerarios de entrenamiento, y reaccionar a las conductas de los alumnos en tiempo real. Estos tutores virtuales combinan técnicas de enseñanza y métodos automatizados de abstracción para brindar retroalimentación en tiempo real en ambientes educativos que son inmersivos, interrelacionando a sus usuarios de manera continua (El Hajji et al., 2025).

Los tutores virtuales tridimensionales integran técnicas pedagógicas y recientes desarrollos tecnológicos en interacción 3D y sistemas de IA para lograr una adecuada optimización de los procesos de aprendizaje. En ambientes virtuales tridimensionales los tutores no solo responden de manera textual o verbal a los usuarios, sino que también los guían en actividades exploratorias y colaborativas en el metaverso educativo. Esta modalidad de tutores eclipsa los sistemas de aprendizaje en línea de hace algunos años al proporcionar

experiencias de tipo real y contextualizadas. También la comunidad académica contemporánea valora la personalización de los contenidos según los logros de cada estudiante, una tendencia de alta importancia en la actualidad (Lampropoulos et al., 2025).

Los Tutoriales de Presencia Virtual en 3D aumentan la interactividad y la inmersión de los estudiantes en contextos de aprendizaje activo en los que la superación de retos puede contribuir a la mejora en la retención del conocimiento. En consecuencia, la tutoría ofrece y en visión tutoría aporta a la educación terciaria la inmersión, personalización y adaptabilidad que ofrece la tecnología (Acevedo et al., 2025).

Características de los tutores virtuales 3D

Una de las principales ventajas de los tutores virtuales en 3D es que cuentan con la posibilidad de personalizar la instrucción. Suelen funcionar basado en algoritmos que ajustan las trayectorias de aprendizaje de acuerdo con las acciones, logros y equivocaciones de cada aprendiz en el ambiente inmersivo. Estos sistemas logran que la comunicación entre el aprendiz y el tutor trascienda el texto, utilizando estímulos contextuales que son producidos en tiempo real en el entorno tridimensional. Esta personalización hace que los tutores virtuales 3D sean efectivos no solo para la supervisión del desempeño académico, sino que también permiten que el diseño de las actividades sea adaptativo a las necesidades cognitivas de cada aprendiz (Wei et al., 2025).

Además, los tutores 3D en entornos virtuales pueden ofrecer su profesionalidad en formativos complejos de manera inmediata, como, por ejemplo, en la enseñanza de aspectos en los que el error debe laminarse de forma rápida para que no se fijen contenidos erróneos. Esto se logra mediante la visualización de la manera en que el alumno interactúa con los objetos virtuales. Esto, a su vez, permite que el tutor realice las moderaciones que se consideren pertinentes en el momento que se produzca la interacción. Estas soluciones pueden ser en texto, voz, o en gestos que, acompañados por el tutor virtual, den a la solución un carácter más integral (Haetami y Khan, 2024).

La otra característica que los distingue es la posibilidad de integrar la colaboración en tiempo real que permite la interacción de varios alumnos y varios tutores virtuales, al mismo tiempo, en un mismo espacio tridimensional. Esto contribuye a darle una dimensión social al aprendizaje, ya que permite la cooperación, el intercambio de saberes y el trabajo conjunto en la resolución de problemas complejos. Esto es especialmente útil en carreras como la ingeniería y la programación, donde la colaboración es un pilar fundamental de la actividad académica (Toapanta et al., 2022).

El sistema tutor virtual reactivo tiene la característica de que puede generar respuestas inmediatas de acuerdo con el comportamiento de los alumnos en el sistema a través de divertidos automatismos. Este sistema puede ser de alta utilidad en entornos de aprendizaje con actividades estructuradas donde hay ejercicios que permiten una práctica de forma continua e instantánea. No cabe duda de que estos sistemas pueden no ser campeones de la complejidad, pero dependen de la interacción realizada (Naranjo et al., 2025).

El sistema tutor inteligente es capaz de modificar no solo las respuestas a las preguntas de un alumno, sino que también tiene la capacidad de modificar la forma de presentación de los contenidos, los grados de dificultad, las secuencias de las tareas que ofrece en función de los patrones de un alumno en particular, de su estilo de aprendizaje y de sus necesidades. El diseño de estos sistemas, en tal caso, requiere de herramientas de data mining y de modelación predictiva que identifiquen cuellos de botella y que ajusten la oferta del sistema a los requerimientos del alumno. Los tutores adaptativos utilizan IA y modelación de entornos tridimensionales (Wei et al., 2025).

Los sistemas de tutoría conversacional apoyados por Procesamiento del Lenguaje Natural (PLN) son capaces de establecer diálogos continuos con un estudiante en un entorno 3D y, por lo tanto, hacen que la interacción sea más humana. Estos modelos son representaciones de un avance importante, en el que los estudiantes pueden expresar una inquietud, obtener una aclaración más detallada y de una manera más compleja, mantener una

conversación pedagógica con el sistema. La tutoría conversacional en entornos inmersivos facilita la enseñanza activa, promueve la comprensión más contextualizada del conocimiento y la autorregulación del estudiante en su aprendizaje (Chajuan et al., 2025).

Modelos de tutores virtuales (reactivos, adaptativos, conversacionales)

La creación de estos avatares virtuales requiere un motor de graficación tridimensional. Esta graficación permite construir, de forma tridimensional, mundos educativos interactivos con un nivel de detalle visual, movilidad, y flexibilidad de modificaciones a un mundo virtual. Estas soluciones permiten unir AI, físicas y realismos visuales y, sobre todo, interactividad en tiempo real. Estas son, racionalmente, las bases gráficas de Entornos Inmersivos (Samala y Dilli, 2025).

Por otra parte, los docentes virtuales acceden a módulos de IA y aprendizaje automático para entender el comportamiento de los estudiantes y así modificar itinerarios, retroalimentación de forma automática y prever problemas a nivel educativo. La tutoría se tiene que basar en el contexto, y para ello hay que utilizar las IA y los sistemas de feedback en tiempo real para mejorar (Filippone et al., 2025).

La inclusión de sistemas de interacción natural como el procesamiento de lengua natural, el reconocimiento de la voz y el de gestos, permite a los estudiantes interactuar con los tutores de una forma más intuitiva. Estas herramientas de tutores virtuales permiten a los estudiantes interactuar a través de voz o texto y dan la percepción de una mayor presencia a los estudiantes en el entorno 3D. Esto motiva a los estudiantes a interactuar más con el software y hace que la tecnología sea más accesible para usuarios con menor habilidad en el uso de sistemas computacionales. Estas tecnologías han mejorado el aspecto pedagógico de los tutores virtuales y han aumentado la accesibilidad a los mismos para usuarios de educación superior (Guamán et al., 2025).

Herramientas utilizadas en su desarrollo

Los motores 3D son una de las nuevas tecnologías que permiten el desarrollo de tutores virtuales en mundos de aprendizaje inmersivos. Esto es debido a que crean escenarios tridimensionales que son interactivos, realistas, y muy dinámicos. Unity y Unreal Engine son ejemplos de plataformas que permiten la integración de objetos virtuales, y también de físicas, animaciones, y de eventos que son interactivos y que responden a las acciones del usuario. Estos motores también permiten el uso de componentes de inteligencia artificial y sistemas de feedback en tiempo real, lo que favorece el desarrollo de entornos inmersivos para el aprendizaje. Asimismo, su uso en dispositivos múltiples permite el acceso a mundos virtuales en la educación superior (Rasim R. , Rosmansyah, Langi, & Munir, 2025).

Entornos como Second Life permiten la creación de mundos virtuales accesibles a través de la web, donde los usuarios pueden interactuar entre sí a través de la creación de avatares. Esta característica ha permitido la inserción de tutores virtuales en escenarios educativos. Second Life proporciona la creación de mundos virtuales de forma sencilla, lo que ha permitido su uso en diversas instancias de la educación superior, donde se crean escenarios educativos, aulas y laboratorios virtuales. En la plataforma, los estudiantes pueden participar de forma sincrónica o de forma diferida junto con los agentes virtuales, lo que les permite participar en actividades de aprendizaje colaborativo y activo. Adicionalmente, la plataforma incluye diversas aplicaciones a través de scripts y servidores (Filippone et al., 2025). Los lenguajes de programación y frameworks utilizados en la construcción de tutores virtuales 3D incluyen lenguajes de programación propios del mundo virtual y también de backend.

Definición de entornos inmersivos

Los espacios tridimensionales inmersivos permiten a los usuarios interactuar en tiempo real con representaciones virtuales, lo que les da una sensación de estar dentro de un ambiente. Utilizan elementos visuales, sonoros y de interacción para simular ambientes reales o ficticios con fines pedagógicos. Estar inmersos en un ambiente virtual les permite a los

estudiantes explorar y manipular objetos, y tomar decisiones dentro de ese entorno. Facilitan a los estudiantes experiencias de aprendizaje en el que pueden tener un rol más activo. Han comenzado a utilizarse en un número creciente de instituciones de educación superior (Pimbo et al., 2024).

Características educativas de los entornos 3D

Entre las dos principales características educativas de 3D, se destaca la interactividad, donde el estudiante desempeña un papel activo en la construcción de su propio conocimiento. Este tipo de entorno facilita la representación visual de conceptos abstractos, la simulación de procesos complejos, y la posibilidad de realizar experimentos dentro de un escenario seguro y controlado (Filippone et al., 2025). Adicionalmente, el uso de avatares virtuales promueve el aprendizaje colaborativo, y la interacción entre pares y docentes. La comprensión y la retención del conocimiento se ven incrementadas, gracias a la posibilidad de recibir retroalimentación, y repetir situaciones. Estas características convierten a los entornos 3D en herramientas pedagógicas innovadoras (Pimbo et al., 2024).

Tabla 3

Características educativas de los entornos inmersivos 3D

Característica	Descripción	Aporte al aprendizaje activo
Interactividad	Permite al estudiante interactuar con objetos, escenarios y otros usuarios dentro del entorno virtual.	Favorece la participación activa y la toma de decisiones durante el aprendizaje.
Inmersión	Genera sensación de presencia mediante entornos tridimensionales visuales y auditivos.	Incrementa la motivación y el compromiso del estudiante con la actividad educativa.

Visualización de conceptos	Representa procesos abstractos o complejos de forma gráfica y manipulable.	Facilita la comprensión y retención del conocimiento.
Aprendizaje colaborativo	Posibilita la interacción entre múltiples estudiantes mediante avatares.	Impulsa el trabajo en equipo y la construcción social del conocimiento.
Simulación de escenarios de la vida real.	Reproduce contextos laborales o experimentales con seguridad.	Facilita el aprendizaje práctico sin riesgos reales.
Feedback	Da respuestas y resultados en realidad a las acciones del estudiante.	Potencia el aprendizaje independiente y la autovaloración.
Reexperimentación y exploración.	Permite que las actividades se repitan y que se puedan explorar diferentes soluciones.	El aprendizaje se consolida si se aplica.

Nota. Matriz de Características educativas de los entornos inmersivos 3D. Obtenido de (Pimbo et al., 2024).

Aprendizaje activo y constructivismo en mundos virtuales

Los mundos virtuales son un ejemplo de aprendizaje activo que se basa en el constructivismo, que considera que el aprendizaje se genera a través de la interacción del estudiante con el entorno. En el aprendizaje 3D, el estudiante deja de ser solo un observador y pasa a ser un participante activo que explora, experimenta, y toma decisiones. Este proceso de participación y toma de decisiones, en 3D, potencia la construcción del conocimiento. Adicionalmente, la interacción con los escenarios y objetos virtuales fortalece la comprensión de la construcción de conocimientos y conceptos, lo cual confirma que los mundos virtuales son un ejemplo de aprendizaje activo (González et al., 2024).

Del mismo modo, los entornos virtuales tridimensionales pueden favorecer el aprendizaje basado en la resolución de problemas y la colaboración, que son dos de los pilares del constructivismo social. Los participantes pueden comunicarse y trabajar en la elaboración

de soluciones, a través de la construcción de avatares dentro del ambiente virtual. Esta actividad los ayuda a desarrollar habilidades como el pensamiento crítico y la reflexión. Colaborar en un contexto simulado les permite combinar e integrar los saberes teóricos con los saberes prácticos. De esta manera, el aprendizaje se hace más significativo y contextual (Rasim et al., 2025).

Por otro lado, los mundos virtuales ayudan a los estudiantes con la contextualización del aprendizaje, ya que los ayudan a recrear entornos laborales y situaciones del mundo real en un entorno controlado y seguro. Esta situación ayuda a los estudiantes a tener una motivación intrínseca al comprender la verdadera utilidad de los contenidos que están aprendiendo. Ser capaz de experimentar sin tener que afrontar riesgos reales, ayuda a la autonomía y la confianza en el proceso educativo. También, la posibilidad de volver a realizar una actividad o de ser capaz de explorar libremente, ayuda a la consolidación del aprendizaje. En la educación superior, estos entornos ayudan a la obtención de aprendizajes más significativos y con un mayor nivel de vínculo a la práctica profesional (Tene et al., 2024).

Second Life como entorno educativo

La plataforma Second Life ha conseguido ser utilizada como herramienta educativa por la creación de entornos virtuales inmersivos. A través de esta plataforma, educadores y educandos pueden interactuar por medio de avatares, lo que les ayuda a conseguir experiencias más interactivas y a enriquecer el proceso de enseñanza-aprendizaje. Gracias a su estructura abierta, esta plataforma ayuda a personalizar y a construir aulas virtuales, laboratorios y espacios colaborativos a distintas áreas del conocimiento. En el ámbito de la educación superior, la plataforma Second Life ha sido utilizada para la simulación de escenarios laborales y para el reforzamiento de la educación experiencial. Por todas estas características, esta plataforma se ha convertido en un espacio para la innovación de la práctica pedagógica (Setiawan et al., 2025). Desde un punto de vista pedagógico, Second Life

promueve el aprendizaje activo, ya que los estudiantes pueden explorar, experimentar y trabajar colaborativamente en un mismo espacio virtual (Pimbo et al., 2024).

Las plataformas en línea más avanzadas y completas desde el punto de vista tecnológico pueden proporcionar beneficios no solo para el diseño de entornos de aprendizaje, sino también para el desarrollo de tutores virtuales y sistemas educativos inteligentes. En el caso de Second Life, es posible programar eventos y comportamientos en sus entornos virtuales utilizando el Linden Scripting Language y también es posible la integración de servicios a través de APIs. Esta flexibilidad permite la inclusión de tutores virtuales inteligentes que pueden comunicarse y responder a los estudiantes en tiempo real. También, su compatibilidad y soporte para arquitecturas distribuidas permiten una fácil escalabilidad de los sistemas. Por lo tanto, Second Life es una de las mejores plataformas para el desarrollo y la experimentación de prototipos educativos de alta complejidad en la educación superior (Acevedo et al., 2025).

Modelos de interacción humano–agente en entornos virtuales

Los modelos de interacción humano-agente en espacios virtuales estudian la forma en la que los usuarios se comunican y colaboran con los agentes inteligentes en mundos digitales inmersivos. Estos modelos combinan la interacción humano-computador, la inteligencia artificial y el experience design (diseño de experiencias) para lograr una interacción más natural y efectiva. En entornos 3D, los agentes suelen representarse con avatares que perciben y responden a las acciones del usuario. El nivel de interacción que se puede lograr con un agente influye en el nivel de usabilidad (facilidad de uso del sistema), la aceptación tecnológica y el aprendizaje, de ahí la importancia de estos modelos en el diseño de entornos virtuales tutores (García et al., 2010).

El diseño educativo implica que los sistemas orientan, retroalimentan y apoyan al sujeto, de acuerdo con sus características. Los agentes que son inteligentes son un tipo de agentes que puede adaptar su comportamiento en base a, por lo menos, tres variables: el contexto, el

perfil del estudiante y las actividades que se van realizando en el espacio virtual. Este tipo de interacción que enriquece la experiencia formativa promueve también el aprendizaje autodirigido. Por otra parte, la interacción constante favorece el compromiso del estudiante en el espacio educativo. Por lo tanto, estos modelos aportan a procesos de enseñanza más efectivos y del usuario (Castro, 2025).

Interacción humano-computador (HCI) en entornos virtuales

El diseño y la evaluación de interfaces en las que los usuarios pueden interactuar eficiente y eficazmente con los sistemas virtuales tridimensionales son el foco de atención de la interacción humano-computador en entornos virtuales. En la HCI en estos entornos, la HCI va más allá de las interfaces, agregando componentes espaciales, de movimiento y de contexto. La filosofía de la HCI en estos entornos se basa en la intención de que las interacciones sean intuitivas, fáciles de comprender y sobre todas las cosas, que se alineen con las capacidades cognitivas de los usuarios. Un diseño de HCI que se ejecuta con éxito resulta en el aumento de la usabilidad, y la disminución de la carga cognitiva que el contexto educativo se involucra (Gordon, 2023).

La HCI utiliza avatares, menús en 3D y objetos interactivos en 3D que reaccionan a las acciones de los usuarios. Este tipo de interfaz permite que la HCI sea más intuitiva, lo que ayuda a los usuarios a explorar y manipular el entorno. También, la ayuda auditiva y la ayuda de la interfaz que ofrece retroalimentación en tiempo real a los usuarios mejora su entendimiento de las acciones que están realizando. Un diseño pensado en los usuarios mejora la eficiencia y la satisfacción de los usuarios en la interfaz. De esta forma la HCI ayuda a tener mejores experiencias educativas (Li, 2024).

El uso de HCI en entornos virtuales educativos facilita la interacción de los estudiantes con los sistemas inteligentes. La evaluación constante de la usabilidad y de la experiencia del usuario es fundamental para optimizar el diseño de estas plataformas. Además, la inclusión de criterios de accesibilidad estandarizados asegura que se integre la diversidad en los perfiles de

los usuarios. Una HCI optimizada, en el contexto de la educación superior, promueve el aprendizaje activo y la autodirección, lo que indica que la HCI debe ser uno de los pilares en la construcción de entornos virtuales educativos (Alcívar et al., 2023).

Interacción humano–agente inteligente

La interacción humano-agente inteligente se define como el proceso en el que los usuarios se comunican y colaboran con sistemas con automatismos cognitivos. En el e-learning esos agentes son un soporte del aprendizaje que entendería ciertas acciones del usuario y produce una respuesta adecuada al contexto. En este sentido, incluir inteligencia artificial en los agentes que van más allá de un análisis del comportamiento, permite y ayuda a ajustar la pedagogía de su intervención. Esta interacción posibilita la posibilidad de experiencias más personalizadas y continuas. Aumenta el acompañamiento educativo en el entorno virtual de los estudiantes (Ying et al., 2023).

La relación entre humanos e inteligencia artificial da la oportunidad de ofrecer explicaciones y retroalimentaciones en tiempo real, simulando un tutor. Pueden ser programados para identificar errores y dar estrategias para orientar a un estudiante en la resolución de tareas difíciles. Este tipo de interacción, en gran medida, apoya la autonomía del estudiante, pues brinda asistencia sin la interrupción del proceso que está teniendo. También, apoya en la autorregulación y aprendizaje autodirigido, de aquí que la interacción y el uso de estos sistemas sea de gran importancia en el ámbito de la educación superior (Muirhead et al., 2021).

La interacción humano–agente inteligente mejora gracias a la inmersión 3D que proporciona un acompañamiento visual de los agentes. Las representaciones en forma de avatares de los agentes favorecen la percepción de presencia y facilitan interacciones más fluidas y comprensibles. Por lo demás, los agentes pueden responder con base en eventos del entorno virtual. La fusión de ambas tecnologías (IA y entornos inmersivos) potencia la experiencia del usuario y el aprendizaje significativo y activo (Trunfio y Rossi, 2022).

Modalidades de interacción

Las maneras de relacionarse en los espacios virtuales educativos entre los diferentes actores influyen en los modos de interacción. La interacción en este entorno tridimensional puede realizarse principalmente a través de texto, voz y gestos o eventos. La utilización de distintas modalidades de interacción permite a los usuarios disfrutar de una comunicación más compleja, rica y flexible. Las multiliteracidades son cruciales para crear ambientes más complejos e inmersivos (Chajuan et al., 2025).

La interacción por texto es una de las más utilizadas, ya que permite al estudiante hacer preguntas, recibir instrucciones y proporcionarle retroalimentación en un orden y forma que le resulta muy útil en situaciones donde se exige ser muy preciso en los conceptos y donde todo debe quedar registrado. La interacción por voz, en comparación, dota de mayor naturalidad a la conversación usuario-agente y permite una comunicación más fluida. Ambas modalidades contribuyen a que la experiencia de aprendizaje sea más rica (Gordon, 2023).

La comunicación a través de acciones, gestos y eventos implica el uso de movimientos, selecciones y manipulación de objetos en un entorno virtual. Fortalece la sensación de inmersión y la presencia en 3D. Los eventos y acciones pueden ser interpretados por un agente inteligente, que le da la oportunidad de responder de forma más contextualizada. Mejora el estudio a través de experiencias vivas. En conjunto, la comunicación a través de acciones, eventos y gestos mejora el potencial académico de un AVA (idóneo de un AVA) (Filippone et al., 2025).

Experiencia del usuario (UX) en entornos inmersivos

La experiencia de los usuarios en sistemas 3D y de los entornos inmersivos. Aunque la usabilidad y la accesibilidad son las primeras responsables de la satisfacción del usuario. En el caso de la educación, la UX y la navegación intuitiva cobran importancia, junto a la reducción de carga cognitiva del estudiante. Una experiencia de usuario inmersiva permite que el estudiante se concentre en el aprendizaje y no tenga problemas con la tecnología. La

concordancia entre lo visual y lo funcional contribuye a que el usuario comprenda su entorno, lo que aumenta el compromiso y la permanencia del estudiante en la actividad (Naranjo et al., 2025).

Una experiencia de usuario en 3D inmersivas también está conectada a la sensación de presencia y realismo del espectador La interacción, la retroalimentación en tiempo real y la personalización de los entornos, son aspectos que inciden en el aprendizaje de los usuarios. Una experiencia positiva estimula la motivación, el juego y la participación. También permite que el usuario que adapte el entorno a sus necesidades y capacidades. En este sentido, se ven beneficiados los usuarios y se fortalecen los procesos en la educación superior (Filippone et al., 2025).

Concepto y principios de la arquitectura de microservicios

La arquitectura de microservicios descompone una aplicación en múltiples componentes independientes que se pueden desplegar de forma independiente. Cada microservicio tiene a su cargo una sola tarea y se comunica con otros microservicios mediante APIs RESTful y otras interfaces definidas (Arenale y Saldaña, 2024)**.**

Todo servicio debe poder ser diseñado, mejorado y gestionado de forma aislada. Sin que esto impacte al resto del sistema. Igualmente, el uso de diferentes recursos tecnológicos, atendiendo a las características de cada componente, es un aspecto favorable. El término se aplica sobre todo a los sistemas complejos y distribuidos educativos. Así, se puede confirmar que la arquitectura de microservicios es ideal para la fabricación de soluciones educativas innovadoras (Acevedo et al,, 2025).

Una de las principales diferencias entre microservicios y arquitecturas monolíticas – su escalabilidad. Un sistema monolítico si se quiere escalar, se tiene que escalar toda la app. En el escenario de microservicios, sólo requieren escalar los componentes que sean necesarios, esto genera un mejor uso de la capacidad disponible y mejora el rendimiento del sistema. Por

lo que la demanda de usuarios se presenta como variable en este tipo de entornos educativos (Naranjo et al., 2025). A través del uso de margen de aplicación se presentan beneficios como:

El sistema también tiene mejoras a nivel de mantenibilidad y resiliencia. En una arquitectura monolítica, si hay un fallo en la aplicación es posible que toda la misma se caiga. En los microservicios esto no ocurre, dado que cada componente es independiente y el fallo no replica en los otros. Aumenta de disponibilidad y fiabilidad el sistema educativo. Del mismo modo, los equipos de desarrollo pueden trabajar paralelamente en distintos servicios. Por lo tanto, la arquitectura de microservicios facilita la innovación, la evolución continua y la sostenibilidad de las plataformas (Muirhead et al., 2021).

Ventajas frente a arquitecturas monolíticas

Los beneficios de la Arquitectura de Microservicios, en comparación con la arquitectura monolítica tradicional, se centran en la flexibilidad y escalabilidad. En un sistema monolítico, cualquier tipo de modificación o extensión que se quiera llevar a cabo implica la modificación y redeploy de toda la aplicación, lo que implica un mayor nivel de riesgo en que se puedan presentar fallos. En el caso de los microservicios, se facilita el escalamiento de los componentes que lo necesitan, lo que es clave para entornos con un alto nivel de variabilidad de la población usuaria. Asimismo, esta característica ayuda a optimizar el uso de recursos. Por esto, la arquitectura de microservicios es más eficiente que la arquitectura monolítica para el caso de las plataformas educativas (García et al., 2025).

Los microservicios permiten actualizaciones más rápidas a nivel de sistema, pero se realizan de forma independiente y no hacen que el sistema en su conjunto se detenga. Esto, del mismo modo, previene paradas en el sistema, por lo que optimiza el servicio educativo. A su vez, cada microservicio se puede programar con tecnología que se considere más conveniente. Los sistemas monolíticos suelen estar construidos sobre la misma tecnología disponible por lo que el potencial se limita al mismo. Se puede señalar que los microservicios son más efectivos en el largo plazo (Rasim et al., 2025).

Patrones de diseño relevantes

Los patrones de diseño juegan un papel importante en la implementación de una arquitectura de microservicios. El patrón API Gateway actúa como un único punto de entrada para los clientes y gestiona las peticiones de los clientes y las envía a los servicios. Este patrón ayuda a que haya menos complejidad en la comunicación y más seguridad. Lo que ayuda a la centralización de la autenticación y de control de acceso. En el ámbito educativo, favorece la comunicación en plataformas virtuales y backend services. Esto optimiza la experiencia del usuario y la eficiencia del sistema (Arenale y Saldaña, 2024).

Service Discovery es la otra solución dios que permite a los microservicios resolverse de forma dinámica en la arquitectura. Este patrón es uno de los más importantes a la hora de diseñar distribuidos porque los servicios pueden reubicarse o cambiar su estado. Asimismo, el término comunicación síncrona y asíncrona expresa el modo particular por el que se relacionan los servicios. El uso de la comunicación síncrona asegura la obtención de rápidas respuestas. Mientras que, en la comunicación asíncrona, en donde se prefiere el uso de protocolos de mensajería, se mejora la tolerancia a fallos. En el caso de los sistemas inteligentes de educación, el uso combinado de estos mecanismos garantiza que el consumo de recursos sea óptimamente eficiente, aprovechando los recursos del sistema de tal forma que logra un elevado nivel de disponibilidad y escalabilidad.

Aplicación de microservicios en sistemas educativos inteligentes

La integración de microservicios en sistemas inteligentes de enseñanza permite modular y desacoplar la integración de variados servicios. Pueden funcionar como servicios independientes en un entorno de trabajo cada componente como la analítica de aprendizaje, evaluación, gestión de usuarios y tutores virtuales. Este aspecto no solo facilita la personalización del aprendizaje en diferentes contextos educativos, sino que también garantiza que la mejora o actualización de un componente no interfiera con el funcionamiento de los

demás. En 3D inmersivos, esta arquitectura permite múltiples interacciones en tiempo real. Mejora el aprendizaje de las personas (Castro, 2025).

Las nuevas tecnologías como la inteligencia artificial y el procesamiento de lenguaje natural se pueden integrar más fácilmente gracias a los microservicios. Los sistemas educativos inteligentes pueden consumir servicios especializados para el análisis de la conducta o la retroalimentación adaptativa. También mediante esta arquitectura se logra mejorar la disponibilidad y resiliencia. En la educación superior, es vital para las plataformas con alta demanda tecnológica. Por ende, los microservicios son la base perfecta para desarrollar innovadores y escalables sistemas educativos (Chajuan et al., 2025).

Tecnologías empleadas en el prototipo del tutor virtual

La construcción de un tutor virtual en entornos inmersivos 3D es un sistema que integra múltiples tecnologías que sean funcionales escalables y adaptables. La interacción entre el usuario y el sistema, la interacción entre componentes y la comunicación entre sistemas deben estar soportadas por estas tecnologías. En el ámbito educativo, una correcta selección impactará de manera positiva en el resultado del aprendizaje. Se permite integrar sectores virtuales con servicios de inteligencia ajenos. El uso combinado de lenguajes, APIs y almacenamientos distribuidos refuerza el sistema arquitectura del sistema para dar una solución tecnológica flexible (Arias et al., 2025)**.**

Por otro lado, la tecnología a usar debe cumplir o responder a los requerimientos (funcionales y no funcionales) del sistema. Elementos como el rendimiento, la seguridad, la disponibilidad y la usabilidad son esenciales. En ambientes inmersivos como Second Life, la posibilidad de incorporar tecnología hace crecer las capacidades nativas. Esto hace posible usar inteligencia artificial y servicios personalizados. El tutor virtual puede generar situaciones educativas más dinámicas de este modo. Por lo tanto, la tecnología se convierte en un facilitador de aprendizaje significativo (Gao et al., 2024).

Linden Scripting Language (LSL) en Second Life

El Lenguaje de Scripting de Linden (LSL) es un lenguaje de programación utilizado en Second Life para dar instrucciones a objetos y avatares. Es capaz de gestionar la interacción, las acciones de los clientes y sus respuestas automáticas. Se pueden crear tutores virtuales simples y objetos interactivos en educación mediante LSL. Asimismo, brinda a los usuarios la opción de automatizar procesos como la entrega de información y la evaluación formativa. Su diseño basado en sucesos es adecuado para entornos inmersivos. Por tanto, se presenta como una herramienta clave del prototipo (El Hajji et al., 2025).

Por su parte, LSL permite el contacto con servicios externos mediante peticiones HTTP, lo cual enriquece sus funciones. Es de especial importancia para la fusión de Second Life con sistemas inteligentes. Por medio de esta vía, el tutor virtual puede enlazarse a bases de datos y a motores de procesamiento. No obstante, LSL no es apto para sistemas computacionales de gran complejidad. Su incorporación a servicios de backend aun así compensa ello. De este modo LSL se presenta como un puente entre el mundo virtual y los sistemas inteligentes (Acevedo et al., 2025).

APIs RESTful para integración de servicios

Las API RESTful son clave para la integración de servicios en arquitecturas de sistemas distribuidos. Facilitan la interacción entre el medio inmersivo y los elementos de backend del sistema educativo. Por medio de estas interfaces, el tutor virtual puede enviar y recibir información. Asimismo, fomentan la interoperabilidad entre distintas tecnologías y plataformas. En sistemas que utilizan inteligencia educativa, el acceso a servicios de análisis, personalización y evaluación se realiza a través de alguna API RESTful. Esto garantiza una arquitectura flexible y escalable (García et al., 2010).

A través del uso de los APIs RESTful es posible dividir las responsabilidades que existen en un sistema. Cada servicio puede ser desarrollado y mantenido de manera autónoma. Esto produce un efecto positivo sobre el sistema, ya que el diseño es más fácil de

mantener y la complejidad total del sistema es menor. Cuando se trata del prototipo de tutor virtual, las APIs permitirán añadir ciertas funcionalidades sin incrementar la complejidad del entorno inmersivo (Setiawan et al., 2025). También se podría mencionar que las API's utilizan estándares aceptados en gran parte de la industria lo que también genera regresos en este sentido en la robustez y sostenibilidad del sistema educativo (Li, 2024).

Procesamiento básico de lenguaje natural (NLP)

El NLP permite que el tutor virtual analice e interactúe con la entrada de texto del usuario. Esta tecnología facilita el contacto entre el alumno y el sistema, haciéndolo más fluido y cercano. En el sector educativo, el NLP básico se utiliza para la identificación de preguntas frecuentes y la formación para la generación de respuestas. Además, contribuye a facilitar el acceso al sistema. Su grado de complejidad es apropiado para prototipos educativos, por lo que constituye un recurso válido para las fases más tempranas del desarrollo (Gordon, 2023).

El análisis de los lenguajes naturales podría ser un servicio externo a través de ciertas APIs. Esto no sólo mantiene el rendimiento en el entorno 3D, sino que también acelera los efectos del tutor virtual. Similarmente, facilita el análisis de los patrones de interacción del usuario (estudiante). Esta información puede servir para optimizar la actividad de feedback educativo, aunque necesita un diseño que evite ambigüedades. En consecuencia, el estudio de la comunicación de los lenguajes naturales básicos refuerza la de los sistemas y la comunicación humano-agente (Arenale y Saldaña, 2024).

Almacenamiento distribuido

La tecnología de almacenamiento distribuido permite el manejo eficiente y seguro de grandes volúmenes de datos. En el caso de los sistemas educativos inteligentes, esta tecnología permite almacenar datos de los usuarios, sus interacciones y logros de aprendizaje. El almacenamiento distribuido ofrece gran disponibilidad y tolerancia a fallas, así como acceso simultáneo a los datos desde diferentes servicios. En el prototipo del tutor virtual, se utiliza

almacenamiento distribuido el cual admite mejoramiento de fiabilidad del sistema por soportar persistencia de dato en tiempo real (Guamán et al., 2025).

El almacenamiento distribuido ayuda con la escalabilidad del sistema educativo. Cuando hay más usuarios, el sistema se puede escalar sin degradar el rendimiento. En el contexto de la universidad es relevante (Samala y Dilli, 2025).

Integración tecnológica del sistema

La tecnología integral del sistema tiene que ver con la unificación de todas las tecnologías que componen el prototipo. Esto asegura que el backend, la inteligencia de las herramientas y los servicios inmersivos están funcionando de forma sincronizada. En el contexto educativo, una correcta integración puede optimizar la experiencia de los usuarios. Asimismo, proporciona la certidumbre de que se están cumpliendo tanto los requerimientos funcionales como no funcionales. Para que el sistema pueda cumplir su función, los distintos componentes del sistema deben comunicarse eficientemente. La integración es por ello un elemento básico del diseño (Haetami y Khan, 2024).

En otras palabras, la inversión en la mejora y mantenimiento evolutivo de un sistema, gracias a la incorporación de tecnologías en sus procesos, permite en un sistema la incorporación de funcionalidades que no alteran su estructura. Además, mejora la compatibilidad en sistemas ajenos y en futuras expandibles. Esta integración permite poder ofrecer un sistema educativo más fluido y personalizado en el prototipo del tutor virtual. También, refuerza la línea arquitectónica microservicios. Por eso, la lógica tecnológica hace posible en el sistema educativo inmersivo la sostenibilidad en el tiempo y la adaptación a nuevas necesidades (Guamán et al., 2025).

Requerimientos funcionales

Los requisitos del prototipo del tutor virtual especifican el funcionamiento del sistema en la experiencia inmersiva 3D. Estos requisitos se constituyen como la base objetiva para establecer las potencialidades, limitaciones y expectativas del sistema educativo. La

identificación objetiva de requisitos en el contexto educativo contribuye a la concreción de los objetivos de orden pedagógico (Filippone et al., 2025). Los requisitos, de igual forma, guían el diseño, desarrollo y evaluación del prototipo. Rivkin y Azevedo (2022) ofrecen un par de definiciones básicas de requisitos y pueden ser funcionales o no funcionales. En tal sentido, el ordenamiento de los requisitos contribuirá a una mejor estructuración del sistema. Por lo tanto, se incrementa la efectividad y la fiabilidad del tutor virtual (Gordon, 2023).

La definición de los requerimientos es esencial para conocer la interoperabilidad de otras tecnologías y plataformas educativas posibles. Esto elimina restricciones técnicas y establece criterios de validación. La flexibilidad y personalización del aprendizaje es un requerimiento de un sistema educativo inteligente, y al mismo tiempo, simplifica la evolución y mantenimiento del sistema. En situaciones inmersivas como Second Life, se nos presentan con mayor relevancia su aproximación a la realidad y a la computadora. Los requerimientos son, sin duda, uno de los aspectos más importantes que se darán dentro del diseño del prototipo (Naranjo et al., 2025).

Los tutores virtuales han de ofrecer determinados servicios y funciones a los usuarios. Esto ende a importantes posibilidades de comunicación y socialización con los estudiantes, responder preguntas y proporcionar feedback. El sistema tiene que registrar la evolución de las interacciones y del aprendizaje Cuando el estudiante ingresa a un entorno de aprendizaje virtual, el tutor virtual debe ser capaz de fundirse con el avatar del estudiante. Igualmente, deben ser capaces de comunicarse con otros sistemas mediante interfase de servicio Se refiere a que toda actividad educativa tiene como finalidad la creación del saber, por lo que todo acto es intencional. Por lo tanto, le dan funcionalidad al emergente (Castro, 2025)**.**

Los usuarios deben gestionarse y el contenido debe personalizarse, tal cual marca el otro requisito funcional que hay en la lista. El sistema deberá poder ajustar las respuestas del tutor en función del contexto de aprendizaje. Del mismo modo, será necesario optimizar el acceso y la navegación virtual. Una forma de actuar sabia, educada, culta y avanzada. La

correcta realización del prototipo contribuye a potenciar el aprendizaje. Esto permite la valoración del desempeño también del tutor virtual. Con la ayuda de los requisitos funcionales, los sistemas serán capaces de cumplir sus metas (Gao et al., 2024).

Requerimientos no funcionales

Construir sistemas abarca algo más que la definición de funcionalidades. La escalabilidad, la usabilidad, la seguridad y el rendimiento son requisitos igual de importantes. La seguridad se encarga de que la información del alumno esté protegida y de que el acceso al sistema sea controlado y monitorizado. De acuerdo con lo anterior, los sistemas deben ser utilizables, lo que quiere decir que los estudiantes podrán tener un acceso sencillo y sin complicaciones al entorno inmersivo. La aceptación de los usuarios hacia el sistema construido se ve afectada por todos estos factores.

Asimismo, mediante la escalabilidad el sistema modifica su comportamiento al presentarse variaciones en el número de usuarios. Es relevante en el ámbito educativo universitario. Los requerimientos no funcionales también incluyen disponibilidad y confiabilidad del sistema. Una correcta definición da como resultado una experiencia de usuario sólida y satisfactoria. Además, evitan que el mantenimiento y versión del prototipo se vuelva complicado. En esencia, los requerimientos no funcionales garantizan la calidad del tutor virtual (Gordon, 2023).

Norma IEEE 830: especificación de requisitos de software

La norma IEEE 830 define los requisitos para la especificación de requisitos de software. Ensures the clarity, consistency and completeness of the documentation of the system. Se detalla la estructura y el contenido de un documento de requisitos. Mejora la comunicación entre los desarrolladores, los usuarios y otras partes interesadas. En el entorno educativo, su uso busca ajustar software y objetivos pedagógicos en cuanto a funcionalidad. Igualmente, minimiza la propagación de la ambigüedad en el proceso de desarrollo (Rojas D. , 2016). La norma señala la comunicación de los requerimientos funcionales y no funcionales

que tienen un impacto en la calidad del producto. IEEE 830 indica que los requisitos de software son una descripción de un sistema de software (IEEE Std. 830, 2008).

La norma IEEE 830 también promueve el uso de un lenguaje adecuado para todas las partes interesadas en el proceso. Los proyectos académicos y los prototipos tecnológicos requieren un conocimiento importante del contexto. Una especificación estructurada facilita la validación y verificación del sistema y del producto. Ayuda a detectar precozmente inconsistencias, requisitos deficientes y fases del desarrollo. Según la norma, para el tutor virtual están documentadas las necesidades educativas y las necesidades técnicas. El uso de ese sistema fortalece la planificación y ejecución del proyecto en particular. Por lo tanto, IEEE 830 permitirá realizar el diseño del sistema de forma más rigurosa (Arenale y Saldaña, 2024).

Documentación de requisitos del sistema

El desarrollo de software educativo debe comenzar por la elaboración de un requerimiento de sistema, el cual es uno de los elementos centrales del mismo. El presente documento recoge ordenadamente todas aquellas necesidades de un sistema, tanto funcionales como no funcionales. Por medio del presente documento se logra una mejor comprensión de lo que se puede y lo que no se puede lograr con el prototipo. En el contexto educativo, los objetivos académicos y las soluciones tecnológicas se conseguirían ajustar mejor entre sí. Estos documentos sirven de base para el diseño, la implementación y la verificación del sistema. Una buena documentación permite que los errores y trabajos a retroceder se reduzcan mejor, en un proyecto de carácter tecnológicos educativos esto se vuelve esencial (Trunfio y Rossi, 2022).

Por su parte, la trazabilidad de los requisitos durante todo el ciclo de vida del software se facilita utilizando documentación. Permite asegurar que cada requerimiento se ha implementado y validado correctamente. Igualmente facilita actualizaciones o mejoras del sistema futuras. En lo que respecta al tutor virtual la documentación contribuye a lograr coherencia entre los elementos del entorno inmersivo y los servicios y funciones inteligentes de

backend. Mejora la transparencia y reproducibilidad del proyecto científico-académico. Por tanto, una adecuada documentación de requisitos potencia la calidad y sostenibilidad del sistema (García et al., 2025).

Servicios del backend del tutor virtual

Los servicios de backend del tutor virtual son la parte central que lleva la lógica de sistema educativo inteligente. Estos servicios permiten gestionar la información, controlar las interacciones y comunicar el espacio inmersivo con los componentes inteligentes. En el ámbito de la educación, el backend asegura el funcionamiento y fiabilidad. Asimismo, utiliza otras tecnologías como procesamiento de lenguaje natural y análisis de aprendizaje. Gracias a su arquitectura orientada a microservicios, resulta más modular, escalable y optimiza mejor el uso de los recursos del sistema. De esta forma, el backend se presenta como un recurso esencial para el funcionamiento del tutor virtual (Li, 2024).

Permite que la entidad sea virtual y albergue otras entidades. Promueve la comunicación y el feedback con tantos usuarios como quieras. El backend tiene un procesamiento en otras entidades. La separación ayuda a que el backend funcione mejor y se le puede añadir nuevas funcionalidades sin que se entere el usuario. Los sistemas educativos inteligentes deben prestar especial atención a esto. De esta forma, el backend cobra especial importancia en el prototipo. Esto permite que la experiencia de aprendizaje se ajuste y tenga en cuenta al usuario (El Hajji et al., 2025).

Servicios de autenticación y gestión de usuarios

Los servicios de autenticación y gestión de usuarios permiten asegurar la información y controlar el acceso al sistema. Estos servicios evitan el uso de un tutor virtual para los estudiantes y los administradores cuya identidad no esté validada. Se hace indispensable la protección de los datos personales y académicos en el sistema educativo. Estos servicios también permiten asignar permisos y funciones según el perfil del usuario. Una correcta gestión

de usuarios, además, mejora el seguimiento del avance académico y, por lo mismo, la confianza en el sistema (Acevedo et al., 2025).

Los servicios mencionados permiten así mismo almacenar datos sobre el comportamiento y las interacciones de los usuarios que pueden utilizarse para personalizar la experiencia educativa para cada usuario y para el ámbito de construcción de reportes y estadísticas de uso. La autenticación logra que los mensajes sean efectivamente registrados y también garantiza el cumplimiento de normas de seguridad y privacidad en el caso del tutor virtual. Por consiguiente, estos servicios son elementos esenciales dentro de la educación inteligente (Haetami y Khan, 2024).

Servicios de interacción y respuestas del tutor

Los servicios de respuestas y los de interacciones del tutor sirven para responder preguntas de los alumnos. Se encargan de procesar las consultas y dar una respuesta adecuada. En ambientes educativos, ofrecen feedback en tiempo real y específico, y además integran opciones básicas de comprensión textual y generación de lenguaje natural. Este aspecto resulta muy importante porque optimiza la interacción entre el aprendiz y el tutor virtual. De este modo, hace que el aprendizaje sea más efectivo (Toapanta et al., 2022).

Además, la información del usuario requiere que estas respuestas se ajusten a su historial por parte del tutor. Esto potencia una interacción más personalizada y significativa. Por otro lado, permiten el uso de diversos tipos de tutor virtual. Estos servicios en el prototipo se comunican con Second Life a través de APIs. Esto garantiza que la interacción se desarrolle de manera fluida en el entorno inmersivo. Por ende, los servicios de interacción integran el sistema educativo inteligente para que tenga éxito (Trunfio y Rossi, 2022).

Servicios de almacenamiento y análisis de datos

Los servicios que almacenan y analizan datos pueden registrar y gestionar la información sobre las interacciones, tales como los comentarios del tutor, las acciones del alumno o los logros alcanzados. En el área de los sistemas educativos, esta información se

puede procesar para evaluar el rendimiento y optimizar el proceso educativo. El sistema de almacenamiento distribuido permite alta disponibilidad y protege la integridad de los datos. En el prototipo, estos prestatarios soportan la persistencia de la información, lo que, en primer lugar, alienta el uso de datos para decisión (Muirhead et al., 2021).

En adición, el análisis de datos puede entender de manera particular, el comportamiento y la demanda de cada uno de los alumnos. Esta información puede guiar modificaciones en contenido y en estrategias de enseñanza. De igual manera, simplifica la producción de reportes académicos. En los sistemas de realidad virtual, el análisis de datos contribuye a mejorar la experiencia del usuario. Por eso, los servicios mejoran la inteligencia del tutor virtual y, de esta manera, el sistema educativo (Rasim et al., 2025).

Comunicación con Second Life

La comunicación en Second Life se realiza a través de servicios de integración entre el Immersive Environment y el backend. Esto se traduce en compartir información en tiempo real entre los diferentes actores. En el prototipo del tutor virtual se realizan peticiones HTTP para el envío de datos. Así es como las acciones del avatar se encolan en el backend. Esto permite al profesor virtual responder de manera dinámica. De este modo, resulta fácil interactuar de forma continua en el 3D (Wei et al., 2025)**.**

La posibilidad de integración con otras plataformas también permite el uso de otras funcionalidades nativas de Second Life. La comunicación de Second Life con otras herramientas permite el uso de entornos inmersivos combinado con el tratamiento y almacenamiento de datos y la mejora de la inteligencia y de otros servicios. Asimismo, la flexibilidad y la escalabilidad se fortalecen, que son de importancia particular en los entornos educativos. Por lo tanto, la comunicación dentro de Second Life es esencial en el prototipo (Setiawan et al., 2025).

Capítulo III Análisis y Diseño del Sistema

El presente capítulo expone el análisis y diseño del Prototipo de Tutor Virtual para Entornos Inmersivos 3D, aplicado como caso de estudio a la Carrera de Software de la Universidad de las Fuerzas Armadas – ESPE, implementado sobre la plataforma Second Life. En esta sección se determina los requerimientos del sistema los requisitos siguiendo la norma IEEE 830, además se describe la arquitectura propuesta, la cual es definida como una solución de microservicios, que está orientada a garantizar la facilidad de mantenimiento, la modularidad y escalabilidad. Tambien se incluye el diseño de los componentes, seguido de la lógica de interacción y el modelado de datos que se encarga de la persistencia de sesiones e interacciones. El diseño se centra en la necesidad de asistencia inteligente dentro de entornos inmersivos, integrando un sistema de bi-tutoría: un tutor especializado para apoyo académico y un tutor de información contextual. Ambos se integran a través de un servicio orquestador y un componente puente que permite la interoperabilidad entre Second Life y los servicios de backend a través de interfaces RESTful. Reflejando el alcance del proyecto, este capítulo establece límites para el prototipo como un artefacto funcional para la validación de requisitos y el comportamiento del sistema, dejando la construcción y los detalles de implementación para el capítulo sobre desarrollo.

Especificación de Requisitos

La especificación de requisitos se elaboró siguiendo la norma IEEE 830, con el propósito de definir de manera clara, verificable y trazable las capacidades del prototipo. Para ello, se identificaron los procesos fundamentales de interacción que deben ser soportados en el entorno inmersivo, considerando las características del canal de comunicación (chat local e IM), la necesidad de continuidad conversacional y la integración con servicios backend.

A partir del análisis del contexto de uso, se determinaron cuatro requerimientos funcionales de alto nivel que estructuran el comportamiento esperado del sistema: (i) tutoría académica guiada, (ii) tutoría informativa contextual, (iii) orquestación y enrutamiento de

solicitudes, y (iv) persistencia de sesiones e interacciones. Estos requerimientos se descomponen en requisitos funcionales y no funcionales detallados, de acuerdo con los criterios institucionales y técnicos del prototipo.

Requisitos Funcionales

Tabla 4

Requerimiento funcional RF 1.0 – Interacción del tutor con el usuario en Second Life

Campo	**Especificación**
Código	RF 1.0
Nombre	Interacción tutor–usuario en entorno inmersivo 3D
Descripción	El sistema debe permitir que el usuario interactúe con el tutor virtual mediante mensajes textuales en Second Life (chat local e IM), enviando consultas y recibiendo respuestas asociadas a la misma sesión de interacción.
Entradas	Mensaje del usuario; identificador de sesión asociado al avatar.
Salidas	Respuesta del tutor segmentada según restricciones del canal; confirmaciones de orientación o continuidad
Precondiciones	El avatar del tutor se encuentra activo en la región; el sistema backend está disponible.
Postcondiciones	El mensaje queda asociado a una sesión y se registra como interacción del usuario.
Prioridad	Alta

Tabla 5

Requerimiento funcional RF 2.0 – Respuestas automatizadas para tutoría académica (tutor de clases)

Campo	**Especificación**

Código	RF 2.0
Nombre	Tutor de clases con guía pedagógica
Descripción	El sistema debe proporcionar un modo de tutoría académica que estructure la asistencia mediante una secuencia lógica de sesión (levantamiento de información, planificación breve, tutoría y verificación), adaptando la interacción al nivel y preferencias del usuario.
Entradas	Mensaje del usuario; tema, objetivo y estilo.
Salidas	Respuestas con explicación del tutor clase y verificación mediante pregunta.
Precondiciones	El usuario selecciona el modo de clases
Postcondiciones	La sesión actualiza su estado y queda disponible para la siguiente interacción.
Prioridad	Alta

Tabla 6

Requerimiento funcional RF 3.0 – Respuestas informativas sobre el departamento

Campo	**Especificación**
Código	RF 3.0
Nombre	Tutor de información contextual
Descripción	El sistema debe proporcionar un modo de tutoría informativa orientado a responder consultas rápidas sobre información institucional, contexto del entorno, preguntas frecuentes y contenidos referenciales, priorizando precisión y brevedad según el canal.

Entradas	Pregunta del usuario; identificador de sesión.
Salidas	Respuesta informativa contextual; sugerencia de continuidad o cambio de modo cuando aplique.
Precondiciones	El usuario seleccionó o activó el modo “información”.
Postcondiciones	La interacción queda registrada para trazabilidad y análisis posterior.
Prioridad	Alta

Tabla 7

Requerimiento funcional RF 4.0 – Orquestación y enrutamiento de solicitudes

Campo	**Especificación**
Código	RF 4.0
Nombre	Orquestador de solicitudes y respuestas
Descripción	El sistema debe disponer de un componente central que reciba solicitudes, determine el modo de tutoría (clases o información), enrute la petición al servicio correspondiente y consolide la respuesta para su entrega al entorno inmersivo.
Entradas	Solicitud con modo, mensaje, session_id, canal y user_name.
Salidas	Respuesta consolidada; metadatos de enrutamiento; lista de fragmentos (chunks) para Second Life.
Precondiciones	Disponibilidad de al menos un servicio de tutoría.
Postcondiciones	La solicitud se enruta correctamente y se devuelve respuesta controlada al cliente puente.
Prioridad	Alta

Tabla 8

Requerimiento funcional RF 5.0 – Persistencia de sesiones e interacciones

Campo	Especificación
Código	RF 5.0
Nombre	Persistencia de sesiones, usuarios e interacciones
Descripción	El sistema debe almacenar información mínima de usuarios, sesiones y mensajes para garantizar continuidad, trazabilidad y soporte a validación del prototipo.
Entradas	Identificador de usuario/sesión; mensajes; eventos de sesión.
Salidas	Registros persistentes para consulta y análisis.
Precondiciones	Base de datos disponible; esquema inicial creado.
Postcondiciones	Sesión e interacciones disponibles para auditoría y evaluación.
Prioridad	Media–Alta

Tabla 9

Requerimiento Funcional RF 6.0 – Segmentación y entrega adaptada al canal

Campo	Especificación
Código	RF 6.0
Nombre	Segmentación y entrega por fragmentos
Descripción	El sistema debe segmentar respuestas que excedan el límite del canal, produciendo fragmentos ordenados con coherencia semántica y entrega secuencial.
Entradas	Respuesta consolidada; restricción de longitud del canal.
Salidas	Lista ordenada de fragmentos listos para entrega.
Precondiciones	La respuesta excede la longitud máxima permitida por el canal.

Postcondiciones	El usuario recibe el contenido completo sin truncamiento.
Prioridad	Alta

Tabla 10

Requerimiento Funcional RF 7.0 – Cambio de modo de tutoría (clases ↔ información)

Campo	**Especificación**
Código	RF 7.0
Nombre	Cambio de modo de tutoría
Descripción	El sistema debe permitir cambiar el modo de tutoría entre “clases” e “información” durante una sesión activa, manteniendo consistencia del contexto y registrando el cambio para trazabilidad.
Entradas	Solicitud de cambio de modo; identificador de sesión; modo destino.
Salidas	Confirmación del modo activo; respuesta conforme al modo seleccionado.
Precondiciones	Existe una sesión activa asociada al avatar.
Postcondiciones	El modo de tutoría queda actualizado y persistido en el contexto de sesión.
Prioridad	Alta

Tabla 11

Requerimiento Funcional RF 8.0 – Continuidad de sesión y recuperación de contexto

Campo	**Especificación**
Código	RF 8.0
Nombre	Continuidad conversacional por sesión

Descripción	El sistema debe recuperar el contexto de la sesión a partir del identificador de avatar y sesión, de manera que cada mensaje se procese considerando el historial relevante y el estado vigente del modo de tutoría.
Entradas	Mensaje del usuario; identificador de avatar; identificador de sesión.
Salidas	Respuesta coherente con el historial; actualización del contexto.
Precondiciones	Existe historial previo o estado inicial definido para la sesión.
Postcondiciones	Se conserva coherencia conversacional entre interacciones.
Prioridad	Alta

Requisitos no Funcionales

Los requisitos no funcionales se definen como atributos de calidad que condicionan el desempeño del prototipo, especialmente por operar en un entorno inmersivo con restricciones propias.

Tabla 12

Requerimiento no funcional RNF 1.0 – Escalabilidad

Campo	**Especificación**
Código	RNF 1.0
Nombre	Escalabilidad por microservicios
Descripción	La arquitectura debe permitir agregar nuevos servicios o ampliar capacidades sin alterar el núcleo del sistema, mediante un esquema distribuido basado en microservicios y APIs REST.
Prioridad	Alta

Tabla 13

Requerimiento no funcional RNF 2.0 – Modularidad y mantenibilidad

Campo	**Especificación**
Código	RNF 2.0
Nombre	Modularidad por separación de responsabilidades
Descripción	Los componentes deben mantener responsabilidades desacopladas (cliente puente, orquestación, tutoría académica, tutoría informativa, persistencia) para facilitar mantenimiento, pruebas y evolución.
Prioridad	Alta

Tabla 14

Requerimiento no funcional RNF 3.0 – Resiliencia y tolerancia a fallos

Campo	**Especificación**
Código	RNF 3.0
Nombre	Resiliencia con degradación controlada
Descripción	El sistema debe tolerar indisponibilidad parcial de componentes, aplicando degradación controlada y preservando la continuidad de interacción, evitando fallos abruptos en la experiencia del usuario.
Prioridad	Alta

Tabla 15

Requerimiento no funcional RNF 4.0 – Compatibilidad con Second Life

Campo	**Especificación**
Código	RNF 4.0
Nombre	Adaptación al canal Second Life

Descripción	El sistema debe adaptar la entrega de respuestas a restricciones del entorno (tamaño de mensajes, latencias, canal chat/IM), garantizando que la información sea transmitida de forma íntegra y comprensible.
Prioridad	Alta

Tabla 16

Requerimiento no funcional RNF 5.0 – Seguridad y protección de datos

Campo	**Especificación**
Código	RNF 5.0
Nombre	Seguridad de datos de sesión e interacción
Descripción	El prototipo debe proteger datos de sesiones e interacciones mediante controles de validación, manejo de credenciales fuera del código fuente y separación lógica de servicios, reduciendo exposición de información sensible.
Prioridad	Alta

Diseño del Sistema

El diseño del prototipo se estructura como una solución distribuida que desacopla la interacción inmersiva (Second Life) de la lógica de tutoría inteligente. Este desacoplamiento responde a dos condiciones del problema: en primer lugar, las plataformas inmersivas presentan restricciones propias en el manejo de mensajes, tiempos de respuesta y formatos de comunicación; en segundo lugar, la tutoría inteligente requiere integrar capacidades externas, tales como procesamiento de lenguaje natural, consulta a bases de conocimiento y persistencia del estado conversacional. En consecuencia, se define una arquitectura basada en

microservicios, donde cada servicio concentra una responsabilidad clara y se comunica mediante interfaces RESTful.

Desde una perspectiva de ingeniería de software, el enfoque de microservicios permite que el sistema sea modular, escalable y mantenible. Cada componente puede evolucionar de forma independiente sin comprometer la integridad global del prototipo, lo cual es especialmente relevante en un contexto de tecnologías emergentes donde la sustitución de modelos, el incremento de fuentes de información o el cambio de mecanismos de persistencia son escenarios plausibles. Esta arquitectura también favorece la validación, debido a que las capacidades del sistema se evalúan por servicio (tutor de clases, tutor de información, orquestación e integración con Second Life), facilitando pruebas funcionales y de usabilidad.

Arquitectura Distribuida Basada en Microservicios

La arquitectura general contempla cinco bloques principales: (i) el entorno inmersivo Second Life como capa de interacción con el usuario; (ii) un componente puente desarrollado en C# que actúa como intermediario entre Second Life y el backend; (iii) un servicio orquestador como punto central de enrutamiento y normalización; (iv) el microservicio de tutor de clases para acompañamiento académico; y (v) el microservicio de tutor de información, orientado a respuestas contextuales apoyadas en una base de conocimiento vectorizada. Adicionalmente, se incorpora una base de datos MySQL destinada a la persistencia del tutor de clases.

Esta separación es coherente con el requisito de compatibilidad con Second Life: la plataforma inmersiva no opera como un entorno de ejecución de backend ni está diseñada para integrar directamente procesos avanzados de análisis y persistencia. Por ello, el sistema traslada la lógica de tutoría a servicios externos y mantiene en Second Life únicamente el canal de interacción, reduciendo complejidad dentro del entorno inmersivo y concentrando el control del comportamiento del tutor en componentes controlables y verificables.

Componentes del Sistema

El cliente puente constituye el mecanismo de interoperabilidad entre Second Life y el backend. Su función, a nivel de diseño, es transformar eventos conversacionales del entorno inmersivo en solicitudes HTTP hacia el orquestador, y, a su vez, entregar al entorno inmersivo las respuestas generadas por el backend. Este componente se justifica porque Second Life no expone nativamente un entorno de microservicios; requiere, por el contrario, un actor externo que mantenga la conexión gestione sesiones y garantice que la comunicación con servicios REST sea consistente.

En el diseño se considera que este puente debe manejar dos condiciones del canal: el soporte de chat local e IM, y la necesidad de segmentación de mensajes cuando el contenido excede límites operativos. En consecuencia, el puente se entiende como una capa de adaptación que preserva la integridad del mensaje, evita saturación del canal y reduce pérdidas de información. Adicionalmente, opera como un punto de captura de identificadores de usuario (por ejemplo, el UUID del avatar) para mantener trazabilidad de sesión a nivel del backend.

El orquestador se diseña como un servicio central encargado de administrar el flujo entre el cliente puente y los servicios de tutoría. Su responsabilidad se centra en tres aspectos: (i) recibir solicitudes de interacción con metadatos mínimos (modo, mensaje, sesión, canal); (ii) decidir el destino de la solicitud (tutor de clases o tutor de información) conforme al modo seleccionado o inferido; y (iii) normalizar la salida para devolver una respuesta consistente al cliente puente, preferentemente en forma de fragmentos compatibles con el canal.

Este componente es esencial para mantener un único punto de entrada al backend. A nivel de diseño, ello permite simplificar la comunicación desde el entorno inmersivo, evitando que el cliente puente deba conocer la lógica interna de cada microservicio o sus variaciones. Además, el orquestador contribuye a la resiliencia del sistema, ya que permite manejar

escenarios de indisponibilidad parcial de un servicio y retornar respuestas controladas sin comprometer la operación de los demás componentes.

El microservicio Tutor de Clases es un servicio para brindar tutoría académica. Su necesidad más significativa en cuanto al diseño es la continuidad de la conversación, puesto que el acompañamiento pedagógico tiene fases, preferencias del usuario y estado de sesión que debe ser mantenido en el tiempo. Por este motivo se necesita almacenar los datos en una base de datos, donde se registrarán las entidades con relación al usuario, la sesión, los mensajes, los subtemas y el progreso.

Esta persistencia permite sostener un flujo coherente: el sistema puede recuperar el estado previo de la tutoría (por ejemplo, fase de la sesión, nivel y objetivo definido) y dar continuidad sin reiniciar el proceso. Asimismo, la base de datos habilita trazabilidad y evidencia para validación del prototipo, al registrar las interacciones, los eventos de sesión y los puntos de memoria asociados a la tutoría.

Desde el punto de vista de análisis, esta decisión responde a un requerimiento no funcional clave: integridad y trazabilidad de información, aplicable especialmente en soluciones educativas donde la evaluación del comportamiento del sistema requiere evidencia estructurada.

El microservicio de tutor de información se encarga de responder preguntas de carácter informativo y/o contextuales acerca del departamento, mediante la metodología de recuperación aumentada por generación (RAG). En contraste con el tutor de clases, el microservicio de tutoría de información no requiere de una compleja persistencia conversacional ni el manejo de estados pedagógicos. La meta de estos sistemas de chat es responder preguntas de manera operacional, a partir de un conocimiento acotado y predefinido.

La ausencia de base de datos relacional reduce complejidad operativa y costo, manteniendo el servicio enfocado en recuperación semántica y respuesta contextual. Además, desde una perspectiva de diseño, esta separación evita mezclar requisitos pedagógicos (tutor

de clases) con requisitos de consulta documental (tutor de información), sosteniendo una modularidad clara.

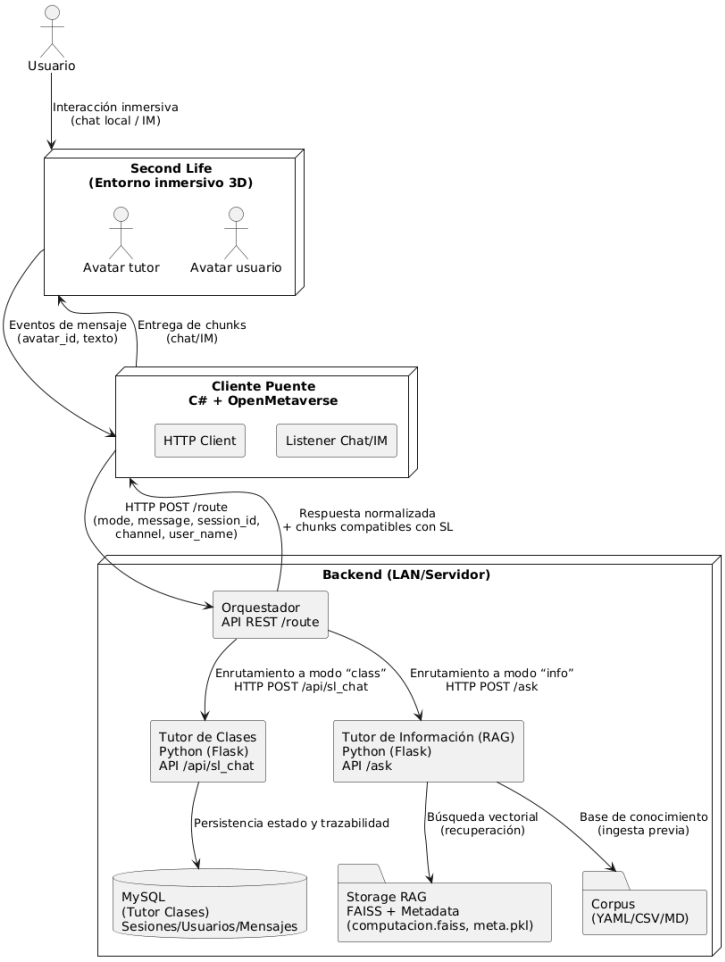
Diagramas Requeridos

En esta sección se presentan los diagramas de diseño que describen la arquitectura y el comportamiento del sistema. Para efectos del documento, se estructuran de acuerdo con cuatro vistas: arquitectura general, componentes, flujo de interacción y modelo de datos.

Diagrama de Arquitectura General del Sistema

Figura 1

Arquitectura General del Prototipo Tutor Virtual (Second Life + Microservicios)



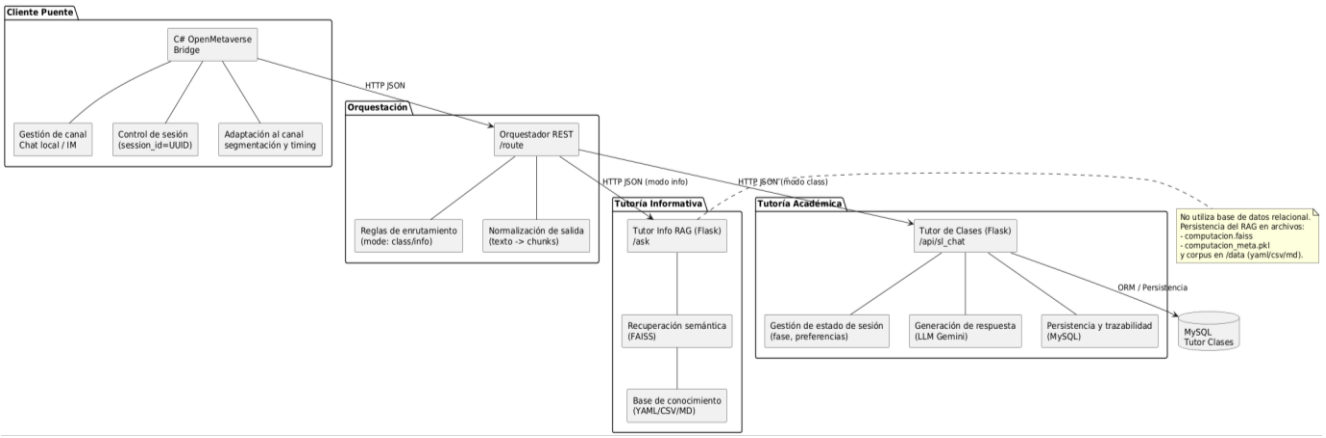
El diagrama de arquitectura general muestra una vista macro del prototipo, evidenciando cómo se integran el entorno inmersivo, el componente puente y los microservicios

del backend. El contexto que se puede inferir del diagrama es que Second Life no es más que una capa de interacción. En este sentido, se pueden observar un par de vías. Por un lado, está la académica o modo clase que va hacia el tutor de clase con persistencia en una base de datos. Y por el otro, está la información o modo información que va hacia el tutor RAG con indexación por vectores. La centralidad que tendrá el orquestador como punto de enrutamiento, permite que el cliente puente hable a un solo endpoint, minimizando el acoplamiento y favoreciendo la consistencia.

Diagrama de Componentes

Figura 2

Diagrama de Componentes del Backend (Microservicios y Responsabilidades)



El diagrama de componentes descompone el sistema en módulos funcionales y muestra las responsabilidades asignadas a cada uno. El propósito de este diagrama es evidenciar que el prototipo no es un sistema monolítico, sino una solución compuesta por servicios especializados.

En este diagrama se observa que el cliente puente concentra la comunicación con Second Life; el orquestador administra el enrutamiento y la normalización de respuestas; el tutor de clases gestiona una lógica de tutoría con estado; y el tutor de información resuelve consultas apoyándose en recuperación semántica. El diagrama incluye explícitamente la

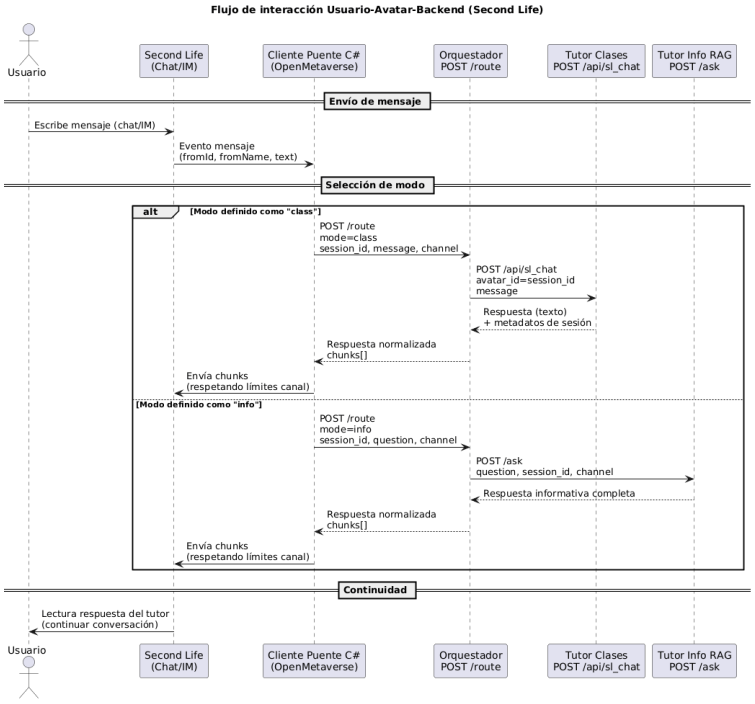
condición de que el tutor info no usa base de datos relacional, dado que su persistencia se realiza mediante archivos del índice FAISS y metadatos asociados.

Este diagrama aporta claridad para justificar requisitos no funcionales como modularidad, mantenibilidad y escalabilidad, mostrando cómo nuevos componentes pueden integrarse sin alterar el núcleo del sistema.

Diagrama de Flujo de Interacción

Figura 3

Diagrama Flujo de interacción Usuario-Avatar-Backend



El diagrama de flujo de interacción describe el proceso extremo a extremo desde que el usuario emite un mensaje en Second Life hasta que recibe la respuesta del tutor. Esta representación se plantea como un diagrama de secuencia, debido a que el objetivo es mostrar el intercambio de mensajes entre actores y servicios.

En este flujo se incluyen los puntos de decisión asociados al modo de operación: si el usuario se encuentra en modo clases, la solicitud se dirige al tutor académico; si se encuentra en modo información, se dirige al tutor RAG. Asimismo, se evidencia el papel del orquestador

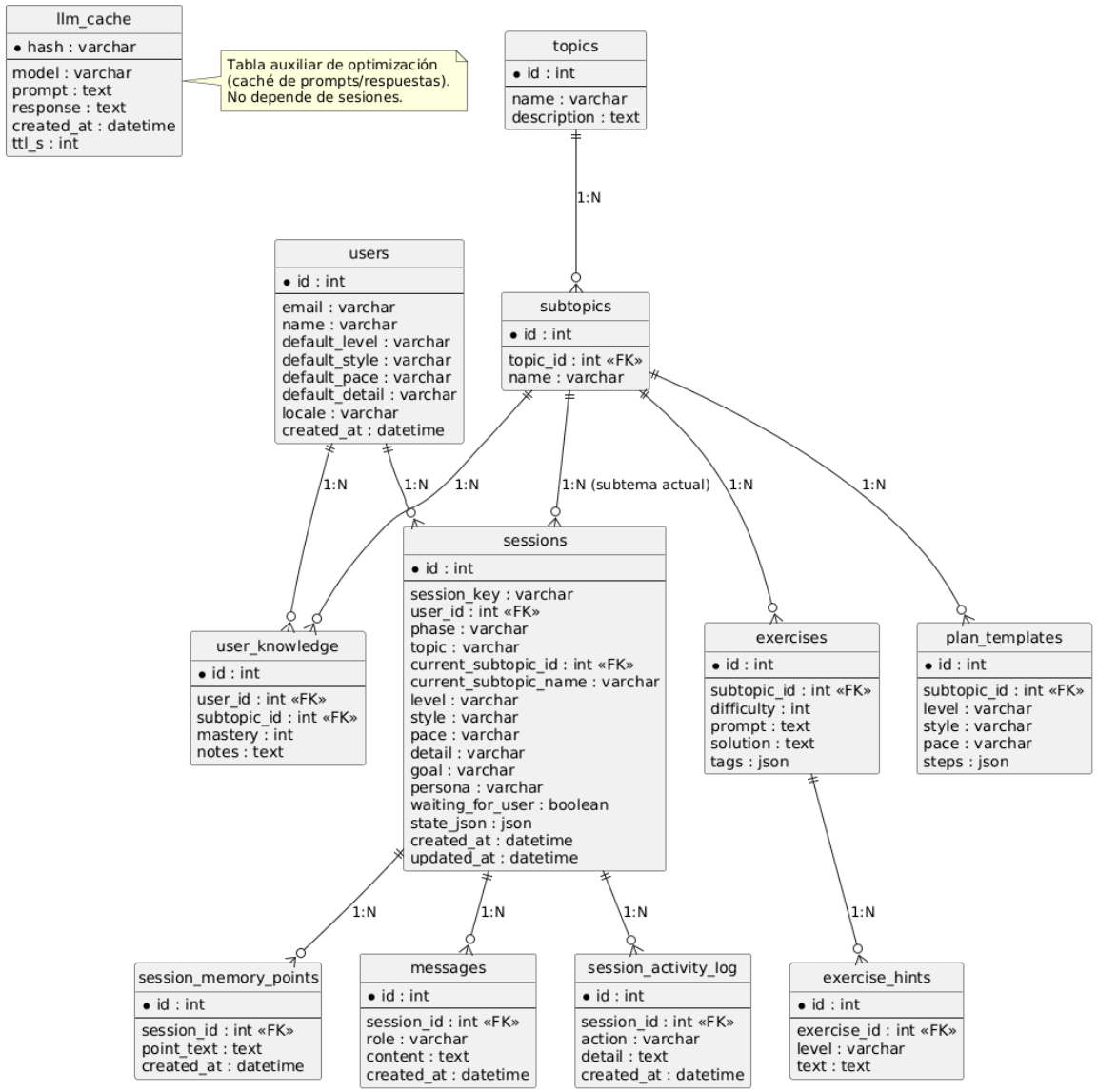
como intermediario de decisión y preparación de la respuesta, y el rol del puente C# como encargado de entregar la respuesta en el canal inmersivo.

Este diagrama es especialmente útil para la etapa de validación posterior, ya que permite definir casos de prueba funcionales basados en el recorrido lógico de una interacción típica.

Modelo Entidad-Relación

Figura 4

Modelo entidad-relación



El modelo entidad–relación representa la estructura de datos asociada al tutor de clases. Su finalidad es demostrar que el prototipo mantiene trazabilidad y continuidad mediante persistencia estructurada, con integridad referencial entre usuarios, sesiones y mensajes. El diseño incluye entidades vinculadas al dominio pedagógico del sistema, tales como temas y subtemas, así como recursos de apoyo como plantillas de plan, ejercicios y pistas.

El modelo también contempla entidades de auditoría y soporte, tales como registros de actividad y puntos de memoria asociados a la sesión, los cuales permiten analizar el comportamiento del tutor y sustentar evidencias para pruebas. Adicionalmente, se incluye una entidad de caché que, desde la perspectiva de diseño, funciona como mecanismo de optimización y no como parte del núcleo funcional.

Es importante destacar que este ERD aplica únicamente al tutor de clases, dado que el tutor de información se soporta en almacenamiento vectorial (FAISS) y archivos de conocimiento, por lo cual no requiere un modelo relacional equivalente.

Casos de Uso del Prototipo

Los casos de uso se derivan de los requisitos funcionales del prototipo y describen, desde la perspectiva del usuario y del entorno inmersivo, las interacciones necesarias para operar el sistema. El diseño contempla dos modalidades de asistencia: tutoría académica (modo clases) y tutoría informativa (modo información), ambas mediadas por un cliente puente en C# y coordinadas mediante un orquestador que enruta solicitudes hacia el servicio correspondiente. La persistencia relacional se aplica únicamente al tutor de clases, debido a su necesidad de continuidad pedagógica y trazabilidad. Con base en este alcance, se establecen los siguientes casos de uso principales.

- CU-01 Iniciar interacción con el tutor y establecer canal de comunicación.
- CU-02 Seleccionar o cambiar modo de asistencia (clases / información).
- CU-03 Recibir tutoría académica guiada (sesión de clases).
- CU-04 Realizar consulta informativa contextual (RAG).

- CU-05 Registrar y persistir sesión e interacciones (tutor de clases).
- CU-06 Consultar progreso académico del usuario.
- CU-07 Reiniciar sesión de tutoría académica.
- CU-08 Verificar disponibilidad operativa de servicios (salud del sistema).

Tabla 17

CU-01 Iniciar interacción con el tutor y establecer canal de comunicación

Campo	**Especificación**
Objetivo	Permitir que el usuario inicie una conversación con el avatar tutor dentro de Second Life y que el sistema establezca el canal de atención (chat local o IM) con una sesión identificable.
Actores	Actor principal: Usuario (avatar). Actores secundarios: Avatar tutor, Cliente puente (C#), Orquestador.
Precondiciones.	El avatar tutor se encuentra conectado a Second Life mediante el cliente puente además el avatar está activo en la región. El acceso al orquestador es por medio de HTTP.
Postcondiciones.	El sistema asigna un identificador de sesión (session_id) el cual está asociado al avatar usuario y la interacción queda habilitada.
Flujo principal	El usuario por medio del chat local dentro de Second Life envía un mensaje al avatar tutor. El mensaje del usuario enviado desde Second Life es recibido por el cliente puente. El cliente puente asocia un session_id o identificador del usuario (UUID) y posterior se registra el canal de origen.

	El sistema retorna un mensaje inicial al usuario, en el cual se indica que ambos tutores están disponibles y las opciones de uso ya sea clases o información del departamento.
Flujos alternos y excepciones	A1. Si el usuario no envía contenido en el mensaje o el mensaje es inválido, el sistema no procesa la solicitud y no genera respuesta. A2. Si el orquestador no está disponible, el cliente puente informa una condición de indisponibilidad temporal al usuario.

Tabla 18

CU-02 Cambiar en el modo de asistencia (clases o información)

Campo	**Especificación**
Objetivo	Permitir al usuario definir el tipo de asistencia que requiere: modo clases o modo información (información sobre el departamento), y además que el usuario cambiar entre modos durante la interacción.
Actores	Actor principal: Usuario. Actores secundarios: Tutores, Cliente puente, Componente Orquestador.
Precondiciones.	Existe una sesión asociada a un usuario y que además este activa. El tutor debe estar disponible.
Postcondiciones.	El modo queda definido para la sesión del usuario y se enruta la siguiente interacción al servicio correspondiente.
Flujo principal	El usuario solicita en el chat local el modo “clases” o “información. El cliente puente se encarga de interpretar la intención del usuario y escoge el modo correspondiente.

	El cliente puente posteriormente confirma el modo seleccionado por el usuario y solicita la siguiente entrada según corresponda (tema que necesita ayuda o pregunta referente al departamento). Las interacciones posteriores se envían al orquestador con el modo ya definido.
Flujos alternos y excepciones	A1. Si el usuario en el mensaje dentro del chat del entorno no define modo, el tutor responde con opciones de escoger entre “clases” o “información”. A2. Si el usuario decide cambiar de modo en mitad de una conversación, el sistema se encarga de confirmar el cambio además de adaptar el contexto de la siguiente interacción.

Tabla 19

CU-03 Recibir tutoría académica guiada (sesión de clases)

Campo	**Especificación**
Objetivo	Conducir una sesión de tutoría académica estructurada dentro de Second Life, desde la recolección inicial de parámetros hasta la generación de orientación pedagógica, manteniendo continuidad por sesión.
Actores	Actor principal: Usuario. Actores secundarios: Avatar tutor, Cliente puente, Orquestador, Tutor de clases, Base de datos (tutor de clases).
Precondiciones.	Existe conectividad tanto con el microservicio de tutor clase como con el orquestador y El usuario ha escogido el modo clases.

Postcondiciones.	La sesión queda registrada y actualizada en la base de datos, el cual incluye historial minino necesario para continuidad y el estado pedagógico.
Flujo principal	El usuario inicia una solicitud de tutoría académica indicando un tema al tutor clases. El cliente puente se encarga de enviar la solicitud al orquestador con parámetros: modo “class” y un identificador de sesión session_id. La solicitud al tutor clases es enrutada por el orquestador. El tutor de clases solicita al usuario información inicial para la sesión (tema, estilo, objetivo). El tutor presenta un plan de estudio para validación del usuario una vez completada la parametrización. El usuario acepta o modifica el plan, y el tutor inicia la guía académica con explicación corta, ejemplos y finaliza con una pregunta de verificación. El sistema registra en base de datos los mensajes de la sesión y el estado conversacional asociado. El usuario continúa con nuevas preguntas o responde a la verificación, manteniendo continuidad de la sesión
Flujos alternos y excepciones	A1. Si el usuario entrega información incompleta en el levantamiento inicial, el tutor solicita nuevamente el dato faltante. A2. Si la sesión se interrumpe (usuario deja de escribir), el estado persiste y puede retomarse cuando el usuario envía un nuevo mensaje con el mismo session_id.

	A3. Si el servicio tutor de clases no responde, el orquestador retorna un mensaje de degradación controlada y sugiere reintento.

Tabla 20

CU-04 Realizar consulta informativa contextual (RAG)

Campo	**Especificación**
Objetivo	Responder a las consultas acerca del departamento de Ciencias de la Computación de manera rápida, sin requerir persistencia relacional.
Actores	Actor principal: Usuario. Actores secundarios: Tutor Avatar, Cliente puente, Orquestador, Tutor de información (RAG), Almacenamiento vectorial FAISS (archivos).
Precondiciones.	El usuario ha escogido el modo información y el servicio RAG se encuentra disponible.
Postcondiciones.	El tutor entrega una respuesta sobre información del departamento al usuario. Esta operación no requiere de base de datos relacional ya que no necesita persistencia.
Flujo principal	El usuario realiza una pregunta sobre algún tema relacionado con el departamento de Ciencias de la Computación al tutor en Second Life. El cliente puente recibe dicha pregunta y posterior envía la solicitud al orquestador con el modo de información. El orquestador se encarga enrutar la consulta al tutor correspondiente.

	El tutor que brinda información es el encargado de realizar una recuperación semántica sobre el índice vectorial y el corpus disponible. El tutor info posterior retorna una respuesta completa al orquestador. El orquestador se encarga de adaptar la respuesta enviar por el tutor información al canal (segmentación si aplica) y la devuelve al cliente puente. El cliente puente entrega el contenido al usuario en Second Life.
Flujos alternos y excepciones	A1. Si la consulta no logra obtener información sobre la pregunta que realizo el usuario, el tutor responde indicando que no cuenta con información suficiente. A2. Si el índice no está disponible, el sistema retorna un mensaje de indisponibilidad del módulo informativo.

Tabla 21

CU-05 Registrar y persistir sesión e interacciones (tutor de clases)

Campo	**Especificación**
Objetivo	Mediante persistencia de mensajes, sesiones y estado pedagógico, asegurar la trazabilidad y continuidad del proceso de tutoría académica.
Actores	Actor principal: Tutor Clases. Actores secundarios: Base de datos, Orquestador, Cliente puente.
Precondiciones.	Existe una interacción en modo clases con session_id válido.
Postcondiciones.	Los datos de la sesión se almacenan en la base de datos y están disponibles para recuperación.

Flujo principal	El sistema identifica un id asociado al avatar dentro de SecondLife session_id. Si session_id no existe, el sistema se encarga de registrar una nueva sesión vinculada al usuario. Se procede a almacenar tanto los mensajes de usuario y tutor como una parte del historial conversacional. En la base de datos, se persiste el estado mínimo de tutoría para continuidad como es el caso de fase, preferencias, subtema y objetivo. La sesión queda disponible para consultas y validación del prototipo.
Flujos alternos y excepciones	A1. Si la base de datos no se encuentra disponible, el sistema opera en modo degradado y no garantiza continuidad.

Tabla 22

CU-06 Consultar progreso académico del usuario

Campo	**Especificación**
Objetivo	Permitir obtener una síntesis del progreso o estado de aprendizaje del usuario, a partir de registros almacenados por el tutor de clases.
Actores	Actor principal: Usuario. Actores secundarios: Tutor de clases, Base de datos, Cliente puente.
Precondiciones.	El usuario dispone de registros previos o una sesión asociada. La base de datos está disponible.
Postcondiciones.	El sistema entrega al usuario una respuesta con información de progreso o estado general, en formato adecuado al canal.

Flujo principal	El usuario solicita su progreso o estado de aprendizaje. El sistema consulta la información disponible asociada al usuario y sus subtemas. El tutor entrega un resumen informativo del estado y sugiere continuidad.
Flujos alternos y excepciones	A1. Si no existen registros previos, el tutor indica que no hay historial suficiente y propone iniciar una sesión.

Tabla 23

CU-07 Reiniciar sesión de tutoría académica

Campo	**Especificación**
Objetivo	Permitir que el usuario reinicie la sesión en modo clases cuando requiere comenzar una tutoría desde cero.
Actores	Actor principal: Usuario. Actores secundarios: Tutor de clases, Base de datos, Cliente puente, Orquestador.
Precondiciones.	Existe una sesión asociada al identificador de sesión.
Postcondiciones.	La sesión se reinicia para efectos del prototipo (se elimina o se marca como reiniciada, según la política definida) y el tutor queda listo para un nuevo levantamiento inicial.
Flujo principal	El usuario solicita reinicio de sesión. El sistema procesa la solicitud y ejecuta el reinicio lógico de la sesión. El tutor confirma el reinicio y solicita nuevamente los datos iniciales de tutoría.

Flujos alternos y excepciones	A1. Si no existe sesión previa, el tutor informa que iniciará una nueva sesión de todas formas.

Tabla 24

CU-08 Verificar disponibilidad operativa de servicios (salud del sistema)

Campo	**Especificación**
Objetivo	Disponer de un mecanismo para monitorear y verificar la disponibilidad de los servicios.
Actores	Actor principal: Sistema encargado de verificar la salud de los tutores. Actores secundarios: Tutor de clases, Tutor de información, Orquestador.
Precondiciones.	Acceso a endpoints de salud.
Postcondiciones.	Se obtiene confirmación del estado operativo(disponible).
Flujo principal	Se invoca el endpoint de salud en el tutor correspondiente. El servicio responde con su estado operativo. El resultado se utiliza como evidencia de disponibilidad en pruebas.
Flujos alternos y excepciones	A1. Si el endpoint no responde, se considera indisponibilidad del componente.

Figura 5

Diagrama general de casos de uso

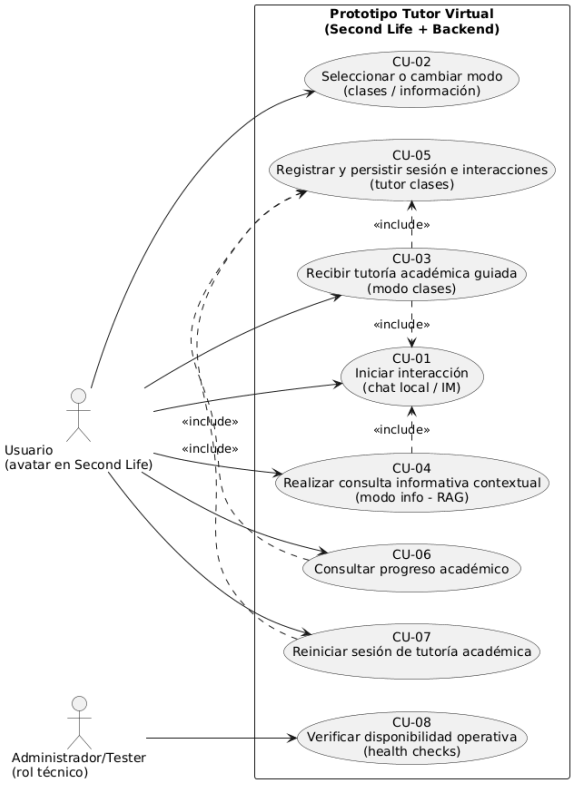


Figura 6

Diagrama de casos de uso – Modo Clases

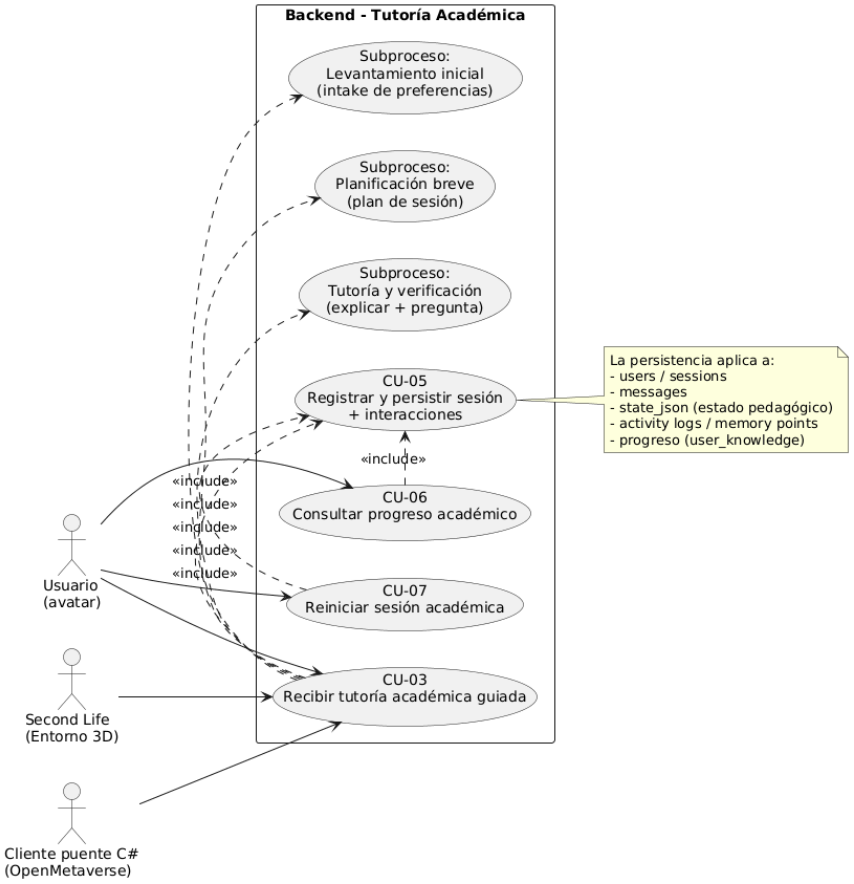
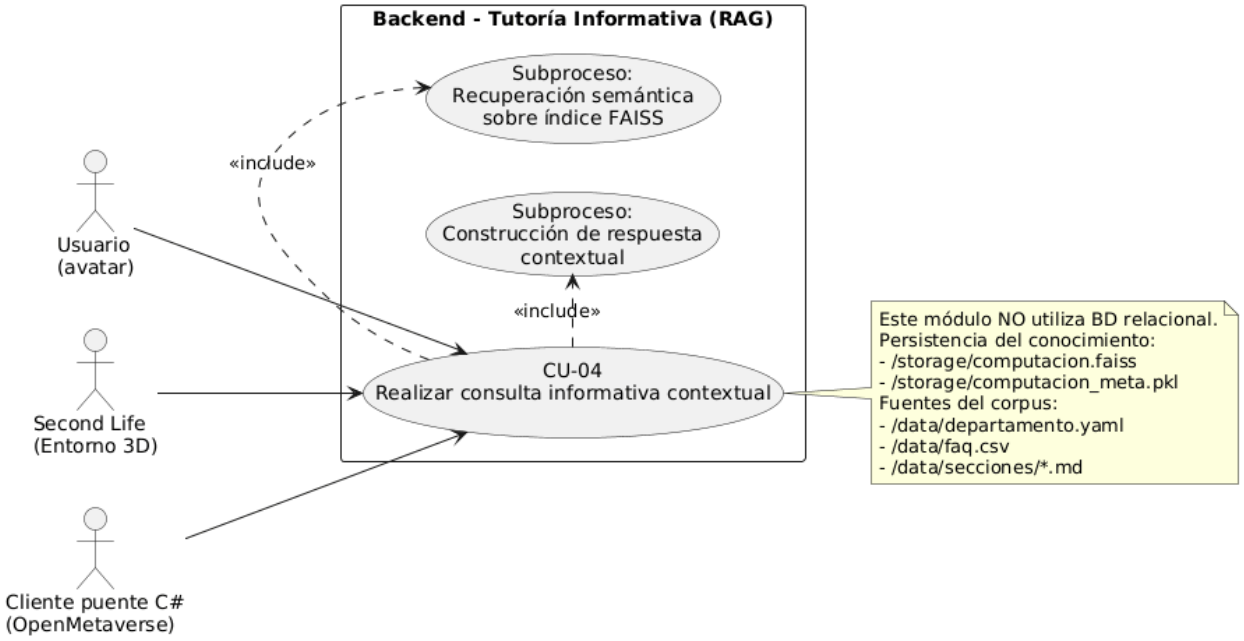


Figura 7

Diagrama de casos de uso – Modo Información



Capítulo IV Desarrollo del sistema

Enfoque de Desarrollo

El desarrollo del Prototipo de Tutor Virtual para Entornos Inmersivos tridimensionales se llevó a cabo mediante un enfoque iterativo e incremental, inspirado en principios de metodologías ágiles basadas en Sprints. Esta decisión metodológica respondió directamente a la naturaleza heterogénea y distribuida del sistema, el cual integra un entorno inmersivo en Second Life, un componente puente de comunicación, múltiples servicios backend especializados y mecanismos de tutoría inteligente basados en modelos de lenguaje.

A diferencia de sistemas monolíticos o de desarrollo lineal, el prototipo presenta una fuerte dependencia entre componentes con características y restricciones muy distintas. Por un lado, el entorno inmersivo impone limitaciones estrictas en cuanto a tamaño de mensajes, latencia aceptable y ritmo de interacción; por otro, los servicios backend requieren mecanismos de orquestación, persistencia, resiliencia y control de estado. En este contexto, un enfoque

iterativo permitió desacoplar progresivamente la complejidad, validando cada capa funcional antes de avanzar hacia niveles superiores de integración.

El desarrollo se organizó en iteraciones sucesivas con objetivos funcionales claramente delimitados. En las primeras etapas se priorizó la conectividad básica entre el entorno inmersivo y los servicios externos, estableciendo un flujo mínimo viable de interacción usuario–avatar–backend. Este paso inicial fue clave para validar tempranamente la viabilidad técnica de la comunicación con Second Life y para identificar restricciones reales del canal que no pueden ser plenamente anticipadas desde el diseño teórico.

A partir de esta base, las iteraciones posteriores incorporaron de manera gradual componentes de mayor complejidad lógica. En particular, se introdujo primero el tutor de información, cuyo comportamiento es esencialmente reactivo y sin dependencia fuerte de estado conversacional. Esta elección permitió validar la integración con modelos de lenguaje y flujos de respuesta completos dentro del entorno inmersivo, sin añadir aún la complejidad de persistencia pedagógica o continuidad de sesión prolongada.

Una vez estabilizado este flujo informativo, el desarrollo avanzó hacia la implementación del tutor de clases, el cual exige un manejo explícito de estado, fases pedagógicas, persistencia relacional y control más fino de la conversación. La incorporación de este componente se realizó cuando ya existía una infraestructura de comunicación y orquestación suficientemente madura, lo que redujo el riesgo de introducir errores sistémicos y facilitó el aislamiento de problemas durante la integración.

El enfoque por sprints también permitió introducir progresivamente requisitos no funcionales críticos, tales como resiliencia frente a fallos parciales, control de frecuencia de solicitudes, adaptación del contenido al canal inmersivo y continuidad de sesión. En lugar de abordar estos aspectos de forma abstracta o anticipada, se integraron a medida que el sistema alcanzaba un nivel de complejidad que los hacía necesarios, asegurando que cada decisión técnica respondiera a necesidades reales observadas durante el desarrollo.

La duración total del proceso fue de aproximadamente dos meses y medio, distribuidos en iteraciones consecutivas que combinaron desarrollo, integración y validación funcional. Cada sprint produjo un incremento tangible del sistema, verificable en el entorno inmersivo, lo que facilitó la evaluación continua del prototipo y permitió realizar ajustes oportunos sin comprometer la coherencia global del diseño.

En conjunto, el enfoque iterativo e incremental no solo permitió gestionar la complejidad técnica del prototipo, sino que resultó fundamental para alinear el desarrollo con las particularidades de Second Life, garantizar la trazabilidad con el diseño del sistema y asegurar que las decisiones de implementación estuvieran sustentadas en validaciones progresivas y observables.

Planificación de Sprints

Los Sprints se han planificado en función de hitos integradores y funcionales, priorizando el desarrollo progresivo del sistema en función de las capacidades observables, antes que en función de tiempos. La idea de esta especificación se da, en gran medida, por la interactividad de componentes distribuidos y por las particularidades que este entorno inmersivo Second Life brinda para la toma de decisiones técnicas en relación con la latencia, el tamaño de los mensajes y la continuidad de la conversación.

Cada sprint se concibió como una unidad de avance funcional completa, orientada a consolidar un conjunto coherente de capacidades del prototipo. En lugar de desarrollar componentes de manera aislada, se priorizó que cada iteración produjera un flujo funcional verificable desde el entorno inmersivo hasta el backend, incluso si dicho flujo era inicialmente limitado en alcance. Este enfoque permitió validar tempranamente la viabilidad de las decisiones de diseño y reducir el riesgo de integraciones tardías complejas.

En una primera etapa, la planificación se centró en establecer las bases del sistema, tanto a nivel del entorno virtual como de la conectividad con servicios externos. Este sprint inicial tuvo un carácter fundacional, ya que permitió comprobar que el entorno inmersivo podía actuar

efectivamente como canal de interacción con servicios backend, sentando las bases para el desarrollo posterior de los tutores inteligentes.

Los sprints intermedios se orientaron al desarrollo progresivo de los componentes de tutoría y del servicio orquestador. En esta fase, la planificación consideró explícitamente la diferencia de complejidad entre el tutor de información y el tutor de clases. El primero fue incorporado antes debido a su naturaleza esencialmente reactiva y a la ausencia de dependencia fuerte de estado persistente, lo que facilitó la validación temprana de flujos de recuperación y generación de respuestas dentro del entorno inmersivo. El tutor de clases, por su parte, fue planificado en un sprint posterior, una vez que la infraestructura de comunicación y orquestación había alcanzado un nivel suficiente de estabilidad.

A medida que los componentes centrales fueron tomando forma, la planificación de los sprints incorporó objetivos asociados a la integración avanzada y al fortalecimiento de requisitos no funcionales. En particular, se incluyeron actividades orientadas a la orquestación de servicios, el control de frecuencia de solicitudes, el manejo de fallos parciales y la adaptación sistemática de las respuestas al canal Second Life. Estas tareas no se trataron como actividades accesorias, sino como parte integral del desarrollo funcional del sistema. El sprint final se planificó con un enfoque predominantemente integrador y de validación. En esta etapa se consolidaron todos los componentes desarrollados, se verificó el flujo completo de interacción usuario–avatar–backend en escenarios representativos y se realizaron ajustes orientados a la estabilidad del prototipo. La planificación de este sprint priorizó la coherencia global del sistema y su comportamiento consistente dentro del entorno inmersivo, más que la incorporación de nuevas funcionalidades.

Tabla 25

Resumen de sprints del desarrollo del prototipo

Sprint	Objetivo principal	Componentes involucrados	Resultado

Sprint 1	Base del entorno inmersivo y conectividad inicial	Second Life, componente puente	Interacción básica usuario–avatar y conexión externa operativa
Sprint 2	Tutor informativo y flujo simple de consultas	Tutor de información, orquestador, puente	Respuestas informativas funcionales en entorno inmersivo
Sprint 3	Tutor de clases con manejo de estado	Tutor de clases, persistencia, orquestador	Tutoría académica guiada con continuidad
Sprint 4	Orquestación avanzada e integración	Todos los servicios	Flujo completo usuario–backend estable
Sprint 5	Validación funcional y ajustes finales	Sistema completo	Prototipo integrado y funcional

El Sprint 1 permitió establecer un entorno inmersivo operativo y validar la comunicación básica con el backend, confirmando que los eventos generados por los usuarios podían ser capturados y procesados externamente. En el Sprint 2, se integró el tutor de información junto con un flujo inicial de orquestación, logrando respuestas informativas completas dentro de Second Life. La tercera etapa del proyecto se caracteriza por la incorporación de un tutor de clase, con manejo explícito de estado y de persistencia, que de alguna manera asegura continuidad a las clases. El objetivo del sprint 4 del proyecto se enfocó que el sistema actuara de forma consistente independientemente del modo de operación. Finalmente, en el Sprint 5 se llevó a cabo la validación de todas las funcionalidades del prototipo, y se definieron en conjunto su estabilidad y consistencia.

En conjunto, la planificación de los sprints permitió un desarrollo controlado y progresivo del sistema, asegurando que cada iteración aportara valor funcional tangible y facilitando la alineación constante entre diseño, implementación y comportamiento observado en el entorno inmersivo.

Desarrollo del Entorno Inmersivo en Second Life

El entorno inmersivo en Second Life constituye el punto de contacto principal entre el usuario y el sistema de tutoría virtual, por lo que su desarrollo fue concebido como un componente funcional del prototipo y no únicamente como un soporte visual.

Sobre esta base se construyó un salón de clases virtual, cuya disposición responde a criterios pedagógicos más que estéticos, privilegiando la claridad espacial, la orientación del usuario y la identificación inmediata del punto de interacción con el tutor.

El salón de clases fue diseñado para simular un espacio académico reconocible, con una organización que refuerza la percepción de formalidad y estructura propia de un entorno educativo. Este diseño cumple una función cognitiva relevante: reduce la carga de exploración del usuario, minimiza la desorientación característica de entornos virtuales abiertos y contribuye a que la atención se concentre en la interacción pedagógica. En este sentido, el entorno actúa como un facilitador del proceso de aprendizaje y no como una distracción.

En este espacio se incorporó permanentemente el avatar tutor, quien encarna y representa visualmente dicho sistema inteligente. La inmovilidad del avatar en la sala de clases hace que el usuario lo identifique inmediatamente como el agente responsable de la tutoría, por lo tanto, no se confunde al respecto con quién interactuar. Esta decisión de diseño ayuda a la continuidad de la experiencia y favorece la naturalidad de la interacción, dado que el avatar es un elemento estable del entorno y no un agente efímero.

El entorno inmersivo también fue concebido considerando las particularidades técnicas de Second Life como canal de interacción. En particular, se tuvo en cuenta que la comunicación principal con el sistema se realiza a través de chat local e instantáneo, mecanismos que condicionan tanto el ritmo de interacción como la forma en que se presentan las respuestas. Por ello, el diseño del entorno prioriza la proximidad espacial entre el usuario y el avatar tutor, facilitando el uso del chat local y reduciendo fricciones en la interacción.

Adicionalmente, el salón de clases funciona como un punto de inicio y retorno para las sesiones de tutoría. El usuario puede aproximarse al avatar para iniciar una interacción, retomar una sesión previa o cambiar el tipo de consulta, sin necesidad de navegar por múltiples espacios. Esta centralización espacial contribuye a la continuidad del uso del sistema y a la percepción de coherencia entre sesiones.

Desde una perspectiva de integración, el entorno inmersivo se desarrolló en estrecha relación con el componente puente, de modo que los eventos generados en el salón de clases como mensajes enviados por el usuario pudieran ser capturados y transmitidos de forma consistente al backend. La validación temprana de esta interacción permitió ajustar aspectos del entorno, como la ubicación del avatar y la modalidad de comunicación predominante, para asegurar un flujo estable de mensajes.

El desarrollo del entorno inmersivo en Second Life no se limitó a la construcción de un escenario visual, sino que se orientó a crear un espacio pedagógico funcional, alineado con los objetivos del prototipo y adaptado a las restricciones técnicas del canal. Este entorno actúa como el punto de anclaje de la experiencia de tutoría, integrando de manera coherente la dimensión espacial con la lógica conversacional y los servicios inteligentes del sistema.

Desarrollo del componente puente (Second Life-Backend)

El componente puente se desarrolló como un elemento fundamental para la interoperabilidad entre el entorno inmersivo de Second Life y los servicios backend del sistema de tutoría virtual. Su función principal es actuar como intermediario entre dos contextos tecnológicos con características muy diferentes: un mundo virtual orientado a interacción social en tiempo real y un conjunto de servicios distribuidos que operan mediante comunicación HTTP y procesamiento asincrónico.

Desde el punto de vista funcional, el puente es responsable de capturar los eventos de interacción generados por los usuarios dentro de Second Life, principalmente a través de chat local y mensajería instantánea, y transformarlos en solicitudes estructuradas que puedan ser

procesadas por el servicio orquestador. Esta transformación no se limita a un simple reenvío de mensajes, sino que implica la gestión explícita de sesión, control de concurrencia y adaptación al canal.

Uno de los primeros aspectos abordados durante su desarrollo fue la identificación consistente de las sesiones de usuario. Para ello, se estableció un esquema de identificación de sesión basado en el avatar del usuario, lo que permite asociar cada mensaje entrante con un session_id lógico en el backend. Esta decisión resulta crítica para sostener continuidad conversacional, especialmente en escenarios donde un mismo usuario mantiene interacciones prolongadas o alterna entre distintos modos de tutoría.

El componente puente también incorpora mecanismos de control para evitar comportamientos no deseados derivados de la naturaleza asincrónica del entorno inmersivo. En particular, se implementaron estrategias de deduplicación de mensajes, basadas en ventanas temporales y huellas de contenido, que permiten evitar el procesamiento repetido de eventos idénticos. Complementariamente, se aplicaron intervalos de enfriamiento por usuario, con el objetivo de regular la frecuencia de envío de solicitudes hacia el backend y prevenir sobrecargas involuntarias.

Un elemento central del desarrollo del puente fue el manejo de las respuestas generadas por los servicios de tutoría. Second Life impone límites estrictos tanto en el tamaño máximo de los mensajes como en la forma en que estos pueden ser transmitidos a los usuarios. Para cumplir con estas restricciones, el puente implementa un proceso de fragmentación segura del contenido, dividiendo las respuestas en segmentos compatibles con el canal y enviándolos de forma progresiva.

Este proceso de entrega considera aspectos técnicos como la codificación de caracteres y la longitud efectiva en bytes, garantizando que cada fragmento sea seguro para su transmisión. Además, se incorporó un retardo configurable entre el envío de fragmentos consecutivos, lo que contribuye a preservar la legibilidad de la respuesta y a evitar pérdidas de

mensajes en el entorno inmersivo. Desde la perspectiva del usuario, este mecanismo mantiene una experiencia conversacional continua, aun cuando las respuestas completas superan las limitaciones del canal.

El puente también gestiona información contextual asociada a cada usuario, como el modo de interacción preferido (tutoría académica o consulta informativa). Adicionalmente, se incorporó la detección de cambios de modo a partir del lenguaje natural utilizado por el usuario, permitiendo alternar dinámicamente entre tipos de tutoría sin requerir comandos explícitos ni interrumpir la conversación en curso.

La comunicación entre el componente puente y el backend se realiza mediante solicitudes HTTP hacia el servicio orquestador, incluyendo de forma explícita el canal de origen. Esta información permite que el backend adapte sus respuestas a las características del entorno inmersivo desde etapas tempranas del procesamiento.

El desarrollo del componente puente permitió abstraer la complejidad propia de Second Life y presentar al backend una interfaz de comunicación controlada y consistente. Al mismo tiempo, garantiza que las respuestas generadas por el sistema inteligente se integren de manera fluida en el entorno inmersivo, cumpliendo los requisitos no funcionales de compatibilidad con el canal, continuidad de sesión y calidad de la experiencia conversacional.

Desarrollo del servicio orquestador

El servicio orquestador se desarrolló como el núcleo de coordinación del backend, con la responsabilidad de centralizar la comunicación entre el componente puente y los servicios de tutoría, y de aplicar de manera uniforme políticas de control, resiliencia y adaptación al canal. Su incorporación permitió desacoplar la lógica del entorno inmersivo de la lógica interna de los tutores, favoreciendo una arquitectura modular y extensible.

Desde una perspectiva funcional, el orquestador expone un punto de entrada unificado que recibe las solicitudes provenientes del entorno inmersivo junto con información contextual mínima, como el identificador de sesión y el canal de origen. A partir de estos datos, el servicio

determina el modo de operación requerido tutor de información o tutor de clases y enruta la solicitud al microservicio correspondiente. Este mecanismo evita que el componente puente deba conocer detalles internos de los tutores y permite modificar o ampliar la lógica backend sin afectar la integración con Second Life.

Un aspecto para tomar en cuenta en el desarrollo del componente orquestador fue la integración explicita de requisitos no funcionales, tales como la estabilidad y confiabilidad del sistema. Por lo que se implementó un control de frecuencias de solicitudes asociados a cada sesión, limitando así el número de peticiones aceptadas dentro de una ventana temporal determinada. Dicho mecanismo implementado protege a los diferentes servicios backend de ráfagas de mensajes generadas por error, además de comportamientos repetitivos del cliente o alguna particularidad del canal inmersivo.

De la misma manera el componente orquestador integra un esquema denominado circuit breaker, el cual está diseñado para evitar la propagación de errores aislando los distintos fallos que puedan presentarse. Cuando un servicio excede un número determinado de fallos consecutivos, el componente orquestador se encarga de suspender temporalmente el enrutamiento hacia el componente que está fallando y además se encarga de responder al usuario mencionando que el servicio no se encuentra disponible. El servicio vuelve a ser evaluado de forma progresiva tras un periodo de enfriamiento, este enfoque aplicado resulta clave en un sistema donde se integran modelos de lenguaje, además de servicios externos que tienen respuestas variables.

Los mismos retrasos que se producen de forma consistente en el tiempo de respuesta de los tutores fueron controlados por políticas de tiempo de espera y reintento en las llamadas HTTP. Con estas políticas se busca que el usuario continúe teniendo interacción con la interfaz y que la experiencia conversacional siga intacta. A través de los health checks, el orquestador obtiene información continua del estado de los servicios para tomar decisiones de enrutamiento. El papel del orquestador es más que solo enrutar y asegurar resiliencia.

También debe adaptar el contenido al canal inmersivo. Las respuestas de los tutores pueden tener estructuras o extensiones incompatibles, pues son propias de entornos textual más ricos y el chat de Second Life no lo soporta. Debido a ello, el orquestador normaliza el contenido, eliminando marcas de formato que no soporta y adecuando el texto para un entorno con una interacción limitada de tipo conversacional.

El orquestador también se encarga de fragmentar las respuestas en paneles de tamaño adecuado que puede enviar a través del canal. Para que el usuario final tenga la información de forma ordenada se hace un proceso de ‘chunking’ antes de enviar la respuesta al componente puente. En caso de fallos parciales, el orquestador aplica estas técnicas de degradación controlada. En vez de presentar errores o interrumpir al usuario, el servicio devuelve resultados acordes a lo educativo, creando contexto, para que el usuario (al menos), sepa que hay ciertas funciones que no están disponibles en ese momento. Este comportamiento es consistente y le da al usuario final una experiencia no discontinua. Esto también da la sensación de que el sistema tiene una mayor resistencia.

El trabajo en el servicio orquestador permitió construir un punto de control para el sistema distribuido donde se unifican las capas de enrutamiento, resiliencia, adaptación al canal y manejo de errores. Esta seguridad fue clave para mantener la estabilidad del prototipo y para asegurar que la interacción en el entorno inmersivo no sólo fuera coherente y fluido, sino que también estuvo alineada con la pedagogía del sistema.

Desarrollo del tutor información

El tutor de información se desarrolló con el objetivo de proporcionar respuestas rápidas, precisas y coherentes a consultas de carácter institucional y contextual, actuando como complemento al tutor de clases. Su diseño responde a un modelo de interacción esencialmente reactivo, orientado a resolver dudas puntuales sin requerir continuidad pedagógica prolongada ni manejo explícito de estado conversacional.

La construcción de este componente ha sido realizada a través de un método de recuperación y generación de conocimiento (RAG por sus siglas en inglés). Este método combina la generación de un modelo de lenguaje y una base de conocimiento institucional preorganizada. Esta decisión técnica resulta especialmente adecuada en contextos académicos, donde es necesario garantizar que las respuestas se fundamenten en fuentes controladas y mantengan un tono institucional consistente.

El proceso de desarrollo inició con la preparación del corpus de conocimiento institucional. Este corpus se conformó a partir de información relevante estructurada en distintos formatos textuales, los cuales fueron tratados de manera homogénea para su posterior procesamiento. Una vez consolidado el corpus, se aplicó un proceso de segmentación (*chunking*) con solapamiento, con el objetivo de preservar continuidad semántica entre fragmentos y mejorar la calidad de la recuperación posterior. La definición del tamaño de los fragmentos y del solapamiento respondió a un equilibrio entre granularidad semántica y eficiencia en la búsqueda.

A través de un modelo de incrustación específicamente diseñado para este propósito, cada segmento se convirtió en un vector. Las representaciones vectoriales se almacenan en un índice de búsqueda semántica basado en FAISS, normalizadas y con una similitud de producto interno. Esta configuración permite búsquedas eficientes y precisas, incluso para colecciones relativamente grandes de fragmentos, y proporciona tiempos de respuesta adecuados para la interacción en tiempo real.

El tutor de información sigue implícitamente el flujo RAG durante la fase de consulta. Se lleva a cabo una transformación de la consulta de usuario en un vector y luego, a partir de un índice, se obtienen set de fragmentos relevantes (top-k). Estos fragmentos, junto con la plantilla, se utilizan como contexto dentro de un prompt. Esta plantilla le asigna al sistema un rol de asistente, incluye instrucciones sobre el estilo, claridad y concisión de la respuesta y, en caso de que el contexto lo justifique, limita la respuesta a lo que se incluye en el contexto.

A diferencia del tutor de clases, el tutor de información no utiliza una base de datos relacional para la persistencia de información. Esta decisión se fundamenta en la naturaleza del componente, que no requiere almacenar sesiones, historial conversacional extenso ni progreso del usuario. En su lugar, el componente opera sobre artefactos persistentes asociados al índice semántico, los cuales encapsulan el conocimiento institucional y permiten su reutilización eficiente sin introducir complejidad adicional.

Durante el desarrollo se prestó especial atención a la adecuación de las respuestas al canal inmersivo. Si bien el tutor de información genera respuestas completas en formato textual, el proceso de fragmentación y adaptación a las restricciones de Second Life se delega al servicio orquestador. Esta separación de responsabilidades permitió mantener el tutor enfocado exclusivamente en la calidad semántica de la respuesta, mientras que la compatibilidad con el canal se gestiona de manera centralizada.

En conjunto, el desarrollo del tutor de información permitió incorporar al sistema una capacidad de consulta institucional robusta, basada en recuperación semántica y generación controlada. Este componente aporta valor inmediato al usuario al resolver dudas puntuales de forma eficiente, al tiempo que mantiene una arquitectura simple y alineada con su propósito específico dentro del prototipo.

Desarrollo del tutor de clases

El tutor de clases constituye el componente de mayor complejidad del sistema, al estar orientado a ofrecer un acompañamiento pedagógico estructurado y sostenido a lo largo de una sesión educativa. Los tutores introducidos en la asistencia a los estudiantes cumplen un rol más reactivo que los tutores de clase. Los actuales tienen que gestionar la continuidad de la conversación, la evolución de las fases pedagógicas, la adecuada adaptación de las interacciones a los perfiles y a los avances de los estudiantes. Por consiguiente, en su desarrollo se prestó especial atención a la implementación explícita de un modelo de estados de la conversación y de la pedagogía. La lógica de tutor presenta un estado de clase que se

brinda constantemente y que captura, simultáneamente, datos contextualizados y decisiones pedagógicas.

Este estado incluye variables relacionadas con el tema de estudio, nivel del estudiante, estilo de explicación, ritmo, nivel de detalle, objetivo de la sesión y rol del tutor. Adicionalmente, incorpora información sobre la fase pedagógica en curso, si el sistema se encuentra a la espera de una respuesta del usuario y un conjunto de puntos de memoria que resumen aprendizajes relevantes detectados durante la interacción. La existencia de este estado permite que cada intervención del tutor sea coherente con el historial de la sesión y con los objetivos previamente definidos.

La tutoría presenta diferentes fases. La fase inicial denominada intake, incluye la recolección de información básica sobre las necesidades del usuario. Para tal efecto, se implementa un modelo con variables pedagógicas que tolera ciertas deficiencias y ambigüedades o bien entradas incompletas. Se usan en este modelo estrategias de fallback que, en la falta de ciertos parámetros, tratan de suponer o solicitar información, para que el diálogo no se detenga. Una vez finalizada esta primera fase y tras haber recogido los datos mínimos, el tutor entra en una breve fase de planificación, donde se organiza la sesión según el tema que se deba trabajar y el nivel que posea el alumno. Esta organización no se pone a disposición del usuario, pero sí sirve de guía para el tutor en la etapa de enseñanza activa. En dicha fase, el modelo de lenguaje genera explicaciones concisas, acompañadas de ejemplos o mini-ejercicios, y culmina con preguntas de verificación que permiten evaluar la comprensión del estudiante y decidir el siguiente paso de la interacción.

A través de la verificación se estudia la relación entre los objetivos y resultados. Se considera el grado de comprensión según las respuestas de los usuarios y se decide si se avanza, refuerza o redefine. Este ciclo contribuye a sostener una tutoría guiada, evitando interacciones puramente expositivas y promoviendo la participación activa del estudiante. Al crear persistencia en una base de datos relacional, se asegura la continuidad de los flujos

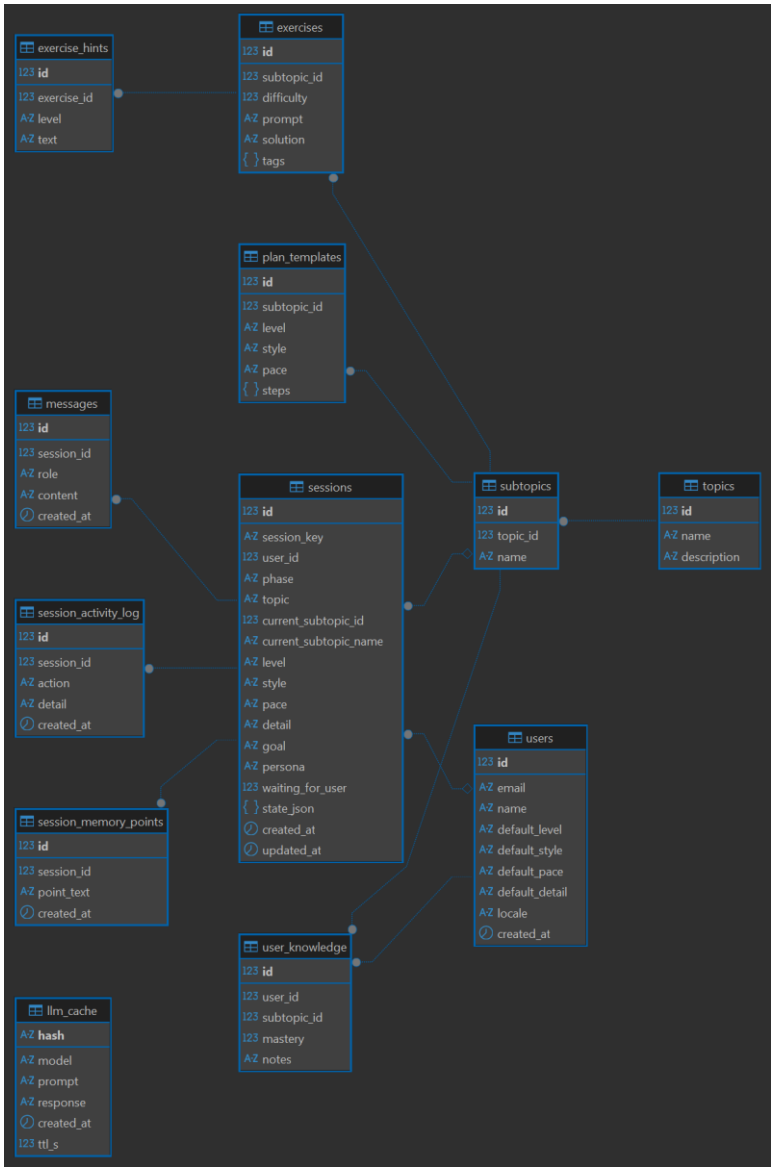
interaccionales en tutor clases. Las sesiones retoman las conversaciones que se ha tenido con usuarios en el pasado. Los mensajes capturan el intercambio conversacional y permiten analizar el histórico.

Los puntos de memoria capturan conceptos clave o dificultades recurrentes identificadas durante la tutoría. Las estructuras asociadas al conocimiento del usuario y su progreso permiten mantener una noción básica de dominio alcanzado.

Las plantillas de ensayos y las hojas de ejercicios son útiles para estructurar las sesiones de tutoría y elevar la experiencia educativa a través de la provisión de ejercicios pedagógicos y estímulos educativos. El modelo genera respuestas que el sistema graba y almacena en caché para poder acelerar las llamadas posteriores. Además, queda un registro de cada sesión que sirve de evidencia para un auditor que comprueba el comportamiento del sistema. Se desarrollaron otros mecanismos de seguimiento orientados a la interacción y a la calidad del sistema. Un mecanismo, que depuró con anterioridad al uso de modelo de lenguaje, las conversaciones de lenguaje irrelevante y ofensivo, mejoró el comportamiento del sistema en un sentido general. Además, se preservó la coherencia pedagógica mientras se implementaron estrategias de recorte de historial, centradas en restricciones operativas, para eliminar cambios irrelevantes en el historial de conversación.

Figura 8

Diagrama Base de Datos



El tutor de clases también considera las limitaciones del canal inmersivo, gestionando internamente la segmentación del contenido generado cuando es necesario, aunque la responsabilidad principal de adaptación al canal recae en el servicio orquestador. Esta doble capa de control contribuye a que las respuestas mantengan una longitud y estructura compatibles con la interacción en Second Life.

En conjunto, el desarrollo del tutor de clases permitió materializar un modelo de tutoría académica guiada que combina estado persistente, fases pedagógicas y generación controlada de lenguaje. Este componente aporta continuidad, coherencia y personalización a la experiencia educativa, y constituye el eje central del valor pedagógico del prototipo.

Tabla 26

Estructuras de persistencia utilizadas por el tutor de clases

Entidad	Propósito	Información almacenada	Uso en la tutoría
users	Identificación del estudiante	Datos básicos y preferencias	Personalización inicial
sessions	Continuidad de la tutoría	Estado, fase, objetivos	Retomar sesiones
messages	Historial conversacional	Interacciones usuario–tutor	Contexto pedagógico
session_memory_points	Hitos de aprendizaje	Conceptos clave detectados	Refuerzo pedagógico
topics / subtopics	Organización temática	Estructura curricular	Navegación didáctica
plan_templates	Planes reutilizables	Secuencia de pasos	Guía de la sesión
exercises / hints	Ejercicios y apoyo	Problemas y pistas	Aprendizaje activo
user_knowledge	Progreso del estudiante	Nivel de dominio	Adaptación del tutor
llm_cache	Respuestas generadas	Prompt y respuesta	Optimización de rendimiento
exercises / hints	Ejercicios y apoyo	Problemas y pistas	Aprendizaje activo

Integración Progresiva y Validación Funcional

La integración del sistema se realizó de manera progresiva, conforme se completaban los distintos componentes. Inicialmente, se validó la interacción entre el tutor de información, el orquestador y el componente puente, asegurando un flujo funcional simple desde el usuario hasta el backend.

Más adelante, se sumó el tutor de clases, controlando la gestión de sesiones, la persistencia de estado y la continuidad del diálogo. Mediante la realización de pruebas de flujo

completo sobre estas integraciones, se logró simular interacciones reales entre el usuario, el avatar tutor y los servicios backend. En los últimos sprints se realizaron validaciones orientadas a la gestión de errores y la estabilidad del sistema, a través de la evaluación del comportamiento frente a cortes de servicio, latencias y cambios de modo de tutoría. Estas pruebas y los últimos retoques aseguraron el funcionamiento del sistema prototipo en el entorno inmersivo.

Tabla 27

Endpoints implementados en el backend del sistema

Componente	**Enpoint**	**Método**	**Parámetros Principales**	**Descripción Funcional**
Tutor de información	/ask	POST	question, session_id, channel	Procesa consultas informativas mediante recuperación semántica y generación de respuesta
Tutor de información	/health	GET	–	Verifica disponibilidad del servicio
Tutor de clases	/api/sl_chat	POST	avatar_id, message, topic (opcional)	Gestiona sesiones de tutoría académica con manejo de estado
Tutor de clases	/api/health	GET	–	Verifica estado del tutor académico
Orquestador	/route	POST	mode, message, session_id, channel	Enruta solicitudes al tutor correspondiente y adapta la respuesta
Orquestador	/health	GET	–	Monitorea el estado de los servicios integrados

Capítulo V Pruebas

Objetivo de las pruebas

El presente capítulo tiene como objetivo describir de manera sistemática la estrategia de pruebas aplicada al Prototipo de Tutor Virtual para Entornos Inmersivos tridimensionales, con el propósito de verificar su correcto funcionamiento, la integración entre componentes distribuidos, la compatibilidad con el canal inmersivo de Second Life, la robustez frente a fallos parciales y la calidad del comportamiento conversacional generado por los tutores virtuales.

Las pruebas se orientaron a confirmar que el sistema desarrollado responde de forma coherente a los requisitos funcionales y no funcionales definidos previamente, garantizando continuidad pedagógica en el modo académico, respuestas informativas consistentes en el modo RAG y una experiencia de interacción estable dentro del entorno inmersivo.

Estrategia y alcance

La estrategia de pruebas adoptada se estructuró por niveles progresivos, iniciando con validaciones por componente y avanzando hacia pruebas de integración y flujo completo extremo a extremo. Esta estrategia favoreció la separación de ciertos comportamientos, la detección de anomalías tempranas y la comprobación continua de la coherencia del sistema en su conjunto. En el primer nivel de pruebas, se realizaron pruebas funcionales de cada uno de los componentes, donde se pusieron a prueba la lógica principal del puente, orquestador y tutores. En esta fase se implementaron pruebas de integración entre servicios para validar la correcta comunicación HTTP y el paso de contexto correcto.

Finalmente, se llevaron a cabo pruebas end-to-end que simularon interacciones reales usuario–avatar–backend dentro de Second Life. El alcance de las pruebas incluyó:

- Funcionamiento funcional de los modos clases e información.
- Integración del entorno inmersivo con los servicios backend.
- Compatibilidad con las restricciones del canal de Second Life.
- Resiliencia ante fallos parciales y manejo controlado de errores.

- Validación cualitativa del contenido generado por los tutores.

Quedaron fuera del alcance pruebas de carga masiva, estrés prolongado o métricas cuantitativas de rendimiento, debido a que el prototipo fue concebido como una solución funcional y validada cualitativamente, y no como un sistema en producción a gran escala.

Entorno y configuración de pruebas

Las pruebas se ejecutaron en un entorno controlado donde los distintos componentes del sistema se encontraban activos de forma simultánea. El entorno incluyó el mundo virtual de Second Life como capa de interacción, el componente puente encargado de capturar mensajes de chat local e IM, el servicio orquestador como punto central de enrutamiento y control, el tutor de información basado en recuperación semántica y el tutor de clases con persistencia relacional.

Gracias a que se usan servicios http para comunicarse, se pudieron validar flujos distribuidos, sin necesidad de exponer las rutas internas ni credenciales. El canal de interacción se probó en el chat local y en la mensajería instantánea, teniendo en cuenta las particularidades de cada uno. La base de datos relacional se utilizó exclusivamente para el tutor de clases, mientras que el tutor de información operó sobre artefactos persistentes de búsqueda semántica.

Diseño de casos de prueba

El diseño de los casos de prueba se basó en los escenarios de uso definidos en el análisis del sistema y en los flujos reales de interacción observados durante el desarrollo. Para evidenciar la cobertura de pruebas, se estructuraron matrices que relacionan componentes, tipos de prueba y resultados esperados.

Tabla 28

Matriz de pruebas por componente

Entidad	Propósito	Información almacenada	Uso en la tutoría

Entorno Second Life	Funcional	Verificar interacción mediante chat local e IM	Mensajes enviados por el usuario son recibidos por el puente
Componente puente	Funcional / Integración	Validar captura de mensajes, sesión por avatar y envío fragmentado	Respuestas entregadas completas y legibles
Orquestador	Integración / Resiliencia	Verificar enrutamiento, rate limiting y degradación controlada	Solicitudes dirigidas al tutor correcto o mensaje controlado
Tutor de información	Funcional	Validar recuperación semántica y respuesta informativa	Respuesta contextual coherente sin uso de BD relacional
Tutor de clases	Funcional	Verificar tutoría guiada con estado y persistencia	Continuidad de sesión y coherencia pedagógica
Base de datos	Integración	Confirmar persistencia de sesiones y mensajes	Registros creados y recuperables
Entorno Second Life	Funcional	Verificar interacción mediante chat local e IM	Mensajes enviados por el usuario son recibidos por el puente
Componente puente	Funcional / Integración	Validar captura de mensajes, sesión por avatar y envío fragmentado	Respuestas entregadas completas y legibles
Orquestador	Integración / Resiliencia	Verificar enrutamiento, rate limiting y degradación controlada	Solicitudes dirigidas al tutor correcto o mensaje controlado
Tutor de información	Funcional	Validar recuperación semántica y respuesta informativa	Respuesta contextual coherente sin uso de BD relacional
Tutor de clases	Funcional	Verificar tutoría guiada con estado y persistencia	Continuidad de sesión y coherencia pedagógica
Base de datos	Integración	Confirmar persistencia de sesiones y mensajes	Registros creados y recuperables

Tabla 29

Casos de prueba funcionales e integración (End-to-End)

ID	Escenario	Entrada	Flujo Esperado	Criterio de Aceptación	Resultado
CP-01	Cambio de modo info → class	Mensaje solicitando cambio de modo	Orquestador enruta al tutor académico	Confirmación de modo y nueva interacción	Éxito
CP-02	Mensaje largo	Consulta extensa del usuario	Respuesta fragmentada en varios mensajes	Mensaje completo sin pérdida de contenido	Éxito
CP-03	Mensaje repetido	Texto enviado dos veces consecutivas	Dedupe evita respuesta duplicada	No se generan respuestas redundantes	Éxito
CP-04	Límite de frecuencia	Envíos rápidos consecutivos	Rate limiting y reintento controlado	Respuesta diferida o mensaje de espera	Observación
CP-05	Servicio no disponible	Tutor info temporalmente inactivo	Degradación controlada del orquestador	Mensaje informativo al usuario	Éxito
CP-06	Persistencia de sesión	Retomar conversación académica	Recuperación del estado previo	Continuidad pedagógica	Éxito
CP-07	Consulta RAG sin contexto	Pregunta ambigua	Respuesta controlada y solicitud de aclaración	No se inventa información	Éxito

Capítulo VI Conclusiones, recomendaciones y trabajos futuros

Conclusiones

El desarrollo del prototipo permitió comprobar la viabilidad técnica de los entornos inmersivos tridimensionales, específicamente Second Life, como canal válido para la implementación de sistemas de tutoría virtual orientados al ámbito universitario. A través de la integración con servicios backend distribuidos, el entorno inmersivo dejó de ser un espacio meramente visual para convertirse en un medio funcional de interacción educativa.

La puesta en marcha de un sistema de tutoría dual, tutor de información y tutor de clase, demostró ser una forma efectiva de atender las necesidades diferenciadas en un único ámbito. La separación de funciones permite, por un lado, ofrecer un feedback pedagógico que está guiado académicamente y es continuo y por el otro, las correspondientes respuestas informativas contextualizadas de carácter institucional. Además, la separación permite no haber interferencias entre ambos tipos de tutorización.

La estructura de microservicios permite incorporación progresiva de distintos componentes como ambiente inmersivo, tutores inteligentes, persistencia de información etc. Esta forma de arquitectura permitió que los prototipos fueran modulares, de mantenimiento y extensibles, lo que estaba alineado con los requerimientos no funcionales expuestos durante el análisis y diseño.

El servicio orquestador se hizo parte crucial del sistema al permitir el enrutamiento de las solicitudes, aplicar políticas de resiliencia y modificar las respuestas según el canal inmersivo.

El componente puente resultó fundamental para superar las restricciones propias de Second Life, particularmente en lo relativo al tamaño de mensajes, ritmo de envío y gestión de sesiones. La adaptación de las respuestas mediante fragmentación controlada y la identificación consistente de usuarios permitió mantener una experiencia conversacional fluida y coherente dentro del entorno inmersivo.

El tutor clases mostró que la orientación didáctica puede ser considerada como una tutoría académica sostenida y ajustada en tridimensionales, que incorpora el manejo explícito del estado, las fases de enseñanza y la persistencia relacional. La disposición de la interacción en etapas sirvió para mantener la coherencia del programa e impulsar una participación más activa del alumno en la sesión.

El enfoque iterativo e incremental basado en sprints fue determinante para gestionar la complejidad del proyecto. Este enfoque permitió validar tempranamente la comunicación con

Second Life, introducir progresivamente componentes más complejos y realizar ajustes fundamentados en el comportamiento observado del sistema, garantizando coherencia entre el diseño propuesto y el prototipo desarrollado.

Recomendaciones

Para futuros desarrollos similares, se recomienda validar tempranamente la interacción real con el entorno inmersivo, priorizando la conectividad y las restricciones del canal antes de incorporar lógica avanzada de tutoría, con el fin de reducir riesgos técnicos en etapas posteriores.

Se debe mantener una separación de responsabilidades clara entre los componentes, específicamente entre la adaptación al canal inmersivo y la lógica de generación de respuestas, lo que facilita el mantenimiento y la evolución del sistema. Es importante que desde las etapas iniciales se implementen mecanismos de gestión de errores, así como de degradación controlada, para que las fallas parciales no devengan en interrupciones disruptivas de la experiencia del usuario en el entorno inmersivo.

Trabajos Futuros

Resulta pertinente explorar la incorporación de métricas de aprendizaje y seguimiento académico, con el fin de evaluar de forma más objetiva el impacto del tutor de clases en el proceso de enseñanza-aprendizaje.

A nivel técnico, se podría profundizar en estrategias de escalabilidad y rendimiento, evaluando el comportamiento del sistema ante un mayor número de usuarios concurrentes o sesiones simultáneas.

Desde una perspectiva pedagógica, se abre la posibilidad de desarrollar mecanismos más avanzados de personalización, que ajusten la tutoría académica no solo al nivel del estudiante, sino también a patrones de aprendizaje identificados a lo largo del tiempo.

Referencias

Acevedo, M., Cabezas, N., Serna, P., & Araujo, S. (2025). *Desafíos y oportunidades de la inteligencia artificial en la educación superior latinoamericana: una revisión sistemática de la literatura*. Obtenido de Revista InveCom, 6(1). 1-10: https://zenodo.org/records/15508755

Alcívar, K., Bastidas, D., Toctaguano, S., & Mora, A. (2023). *Interacción Humano-Computador en el Metaverso Educativo*. Obtenido de PENTACIENCIAS, 5(2), 94–104.: https://www.editorialalema.org/index.php/pentaciencias/article/view/487

Alvira, F. (2011). *La encuesta una perspectiva general metodológica.* Madrid: Centro de Investigaciones sociológicas.

Arenale, L., & Saldaña, J. (2024). *METODOLOGÍAS, ARQUITECTURAS, TÉCNICAS Y ELEMENTOS EMERGENTES UTILIZADAS EN EL DESARROLLO DE APLICACIONES MÓVILES EDUCATIVAS: UNA REVISIÓN DE LITERATURA*. Obtenido de https://repositorio.ciedupanama.org/bitstream/handle/123456789/925/3%20%281%29.pdf?sequence=1&isAllowed=y

Arias, J., Arias, D., Muñoz, E., Campos, J., Lastra, E., & Guzmán, F. (2025). *Personalización del aprendizaje mediante sistemas de inteligencia artificial adaptativa en entornos virtuales educativos. Revista Latinoamericana de Calidad Educativa, 2(2), 69–76.* Obtenido de https://doi.org/10.70625/rlce/159

Campoverde, E., & Campoverde, M. (2025). *Desafíos y Oportunidades de la Inteligencia Artificial en la Educación Superior Ecuatoriana. Ciencia Latina Revista Científica Multidisciplinar, 9(3), 2684–2704*. Obtenido de https://doi.org/10.37811/cl_rcm.v9i3.17896

Castro, J. (2025). *Manejo de los entornos virtuales y el proceso de enseñanza aprendizaje en institutos de educación superior tecnológica*. Obtenido de revista InveCom, 5(4). 1-9: https://zenodo.org/records/14787697

Chajuan, L., Shuqi, M., Weikang, Z., & Zhiyi, Z. (2025). *Investigación sobre el impacto del aula de realidad virtual inmersiva en la experiencia y concentración de los estudiantes*. Obtenido de Realidad Virtual 29 , 82: https://doi.org/10.1007/s10055-025-01153-w

El Hajji, M., Ait Baha, T., Berka, A., Ait Nacer, H., El Aouifi, H., & Es-Saady, Y. (2025). *Una arquitectura para la tutoría inteligente en realidad virtual: integración de LLM e interacción multimodal para el aprendizaje inmersivo*. Obtenido de https://doi.org/10.3390/info16070556

Fernández, C., Baptista, P., & Hernández, R. (2014). *Metodología de la Investigación.* México: Mc Graw Hill.

Filippone, A., Barbieri, U., Marsico, E., Bevilacqua, A., & Di Fuccio, R. (2025). *Can 3D virtual worlds be used as intelligent tutoring systems to innovate teaching and learning methods? Future challenges and possible scenarios for metaverse and artificial intelligence in education. Engineering Proceedings, 87, 110.* Obtenido de https://doi.org/10.3390/engproc2025087110

Gao, H., Lindquist, M., & Serrano Vergel, R. (2024). *AI-Driven Avatars in Immersive 3D Environments for Education Workflow and Case Study of the Temple of Demeter, Greece.* Obtenido de https://delatorre.ai/avatares-de-ia-en-educacion-una-revolucion-en-la-ensenanza-virtual-a-traves-de-entornos-inmersivos/?utm

García, M., Ramos, C., & González, V. (2010). *MODELOS DE INTERACCIÓN EN ENTORNOS VIRTUALES DE APRENDIZAJE*. Obtenido de https://www.researchgate.net/publication/45254932_MODELOS_DE_INTERACCION_EN_ENTORNOS_VIRTUALES_DE_APRENDIZAJE

García, V., Moreira, R., Ponce, R., & Loor, M. (2025). *Aprendizaje adaptativo a través de la Inteligencia Artificial en la Educación Superior. Revista Científica de Innovación Educativa y Sociedad Actual "ALCON", 5(4), 480–489*. Obtenido de https://doi.org/10.62305/alcon.v5i4.775

González Torres, V., Bracho, P., Lucero, E., Carrillo, M., & Santander, R. (2024). *Immersive learning in the metaverse: A review of evidence on pedagogical effectiveness and implementation gaps in Higher Education.* Obtenido de https://mr.ageditor.ar/index.php/mr/article/view/97?utm

Gordon, R. (2023). *Interacción humano-computador y sus aportes en el desarrollo de la informática aplicada a la educación*. Obtenido de https://www.researchgate.net/publication/373397076_Interaccion_humano-computador_y_sus_aportes_en_el_desarrollo_de_la_informatica_aplicada_a_la_educacion

Guamán, L., Chucho, J., Inga, W., & Chucho, J. (2025). *Implementación de Sistemas de Tutoría Inteligente Basados en IA para la Personalización del Aprendizaje en Matemáticas*. Obtenido de Ciencia Latina Revista Científica Multidisciplinar, 9(1), 752-766.: https://doi.org/10.37811/cl_rcm.v8i6.15792

Haetami, A., & Khan, O. (2024). *The impact of virtual reality on collaborative learning in higher education*. Obtenido de International Journal of Educational Narratives, 2(6), 535–545.: https://doi.org/10.70177/ijen.v2i6.1611

Heinemann, K. (2016). *Introducción a la metodología de la investigación empírica.* Berlín: Paidotribo.

Hernández Sampieri, R. (2015). *Metodología de la Investigación.* Buenos Aires: Mc Graw Hill.

Hernández, R., Fernández, C., & Baptista, P. (2014). *Metodología de la Investigación.* México: McGraw-Hill.

IEEE Std. 830. (2008). *Especificacı´on de Requisitos seg´un el est´andar de IEEE 830*. Obtenido de https://www.fdi.ucm.es/profesor/gmendez/docs/is0809/ieee830.pdf

Lampropoulos, G., Fernández, P., De Bosque, A., & Vergara, D. (2025). *Virtual reality in engineering education: A scoping review*. Obtenido de Education Sciences, 15(8), 1027.: https://doi.org/10.3390/educsci15081027

Li, T. (2024). *Interacción persona-computadora en entornos de realidad virtual con fines educativos y empresariales*. Obtenido de Gestión del Desarrollo de Sistemas Complejos: https://doi.org/10.32347/2412-9933.2024.57.112-117

Muirhead, K., Macaden, L., Smyth, K., Chandler, C., Clarke, C., Polson, R., & O'Malley, C. (2021). *Establecimiento de la eficacia de la educación tecnológica sobre la demencia para profesionales de la salud y la asistencia social: una revisión sistemática. Syst Rev. 2021 , 10 , 252.* Obtenido de https://pubmed.ncbi.nlm.nih.gov/34548101/

Naranjo Ríos, A., Jácome, A., & Quezada, D. (2025). *Uso de Chatbots en la enseñanza de Educación Superior ecuatoriana: una revisión sistemática de los modelos de estudios. ASCE MAGAZINE, 4(3), 2431–2453*. Obtenido de https://doi.org/10.70577/ASCE/2431.2453/2025

Oviedo, P., Goyes, A., Castiblanco, M., & Herrera, S. (2016). *Innovar la Enseñanza. Estrategias Derivadas de la Investigación.* Bogotá: Consejo Latinoamericano de Ciencias Sociales.

Pimbo, A., Godoy, W., Pinos, M., & Tibán, S. (2024). *Entornos virtuales tridimensionales (3D): alternativa pedagógica para la enseñanza y el aprendizaje de las derivadas*. Obtenido de EDUCARE ET COMUNICARE Revista de investigación de la Facultad de Humanidades 12(2):5-16: https://doi.org/10.35383/educare.v12i2.1140

Rasim, R., Rosmansyah, Y., Langi, A., R, Z., & Munir, M. (2025). Immersive *Intelligent Tutoring System for remedial learning using virtual learning environment. Indonesian Journal of Science and Technology*. Obtenido de https://ejournal.upi.edu/index.php/ijost/article/view/38954

Rasim, R., Rosmansyah, Y., Langi, R., & Munir, M. (2025). *Immersive Intelligent Tutoring System for remedial learning using virtual learning environment*. Obtenido de Indonesian Journal of Science and Technology: https://ejournal.upi.edu/index.php/ijost/article/view/38954

Rojas, D. (2016). *Publicación: Aplicación de la norma IEEE 830 para la Especificación de Requisitos de Software (ERS) del Sistema Integrado de Información de Prestaciones Sociales (SIIPS) para la Caja de Retiro de las Fuerzas Militares (CREMIL) - participación como ingenie*. Obtenido de http://repositorio.unillanos.edu.co/entities/publication/bd169460-4d50-41fa-bdba-aafd31987293

Samala, A., & Dilli, P. (2025). *Virtual reality in education: global trends, challenges, and impacts — game changer or passing trend?* Obtenido de Discover Education: https://doi.org/10.1007/s44217-025-00650-z

Setiawan, S., Nugroho, R., Anwar, M., & Asmara, D. (2025). *Evaluating the impact of VR, AR, and wearable devices on engineering students' learning performance. International Journal of Educational Technology.* Obtenido de https://journal.pandawan.id/itee/article/view/962

Tene, T., Tixi, J., Palacios, M., Mendoza, M., Vacacela, C., & Bellucci, S. (2024). *Integrating immersive technologies with STEM education: a systematic review. Frontiers in Education, 9*. Obtenido de https://doi.org/10.3389/feduc.2024.1410163

Toapanta, S., Gómez, E., Zambrano, O., & Ordóñez, E. (2022). *Analysis of Artificial Intelligence Applied in Virtual Learning Environments in Higher Education for Ecuador. IOS Press. páginas: 436-443.* Obtenido de https://repositorio.ister.edu.ec/jspui/handle/68000/234

Trunfio, M., & Rossi, S. (2022). *Avances en la investigación del metaverso: Líneas de investigación y agenda futura. Mundos virtuales 2022 , 1 , 103–129.* Obtenido de https://doi.org/10.3390/virtualworlds1020007

Wei, Z., Liao, J., Lee, L., Qu, H., & Xu, X. (2025). *Towards enhanced learning through presence: A systematic review of presence in virtual reality across tasks and disciplines*. Obtenido de arXiv.: https://arxiv.org/abs/2511.20919

Ying, C., Giap, W., & Sha, Y. (2023). *Diseño y evaluación de entornos de aprendizaje de realidad virtual inmersiva: una revisión sistemática de la literatura*. Obtenido de International Journal of Educational Technology.: https://doi.org/10.3390/su15031964

Apéndices