

**Probador Certificado de ISTQB®**

**Programa de Estudio**

**Nivel Especialista**

**Prueba con IA Generativa (IAGen)**

**Versión ES V01.01**

Traducción realizada por

**Spanish Software Testing Qualifications Board**

Traducción del Programa de Estudio

“ISTQB® Certified Tester - Specialist Level - Testing with Generative AI (CT - GenAI)  
- Version V1.0”



## Nota sobre derechos de propiedad intelectual

Nota sobre Derechos de Propiedad Intelectual (Copyright) © International Software Testing Qualifications Board (en adelante denominado ISTQB®).

ISTQB® es una marca registrada de International Software Testing Qualifications Board.

Derechos de Propiedad Intelectual (Copyright) © 2025, los autores Abbas Ahmad, Gualtiero Bazzana, Alessandro Collino, Olivier Denoo y Bruno Legeard.

Todos los derechos reservados. Por la presente, los autores transfieren los derechos de autor al ISTQB®. Los autores (como actuales titulares de los derechos de autor) y el ISTQB® (como futuro titular de los derechos de autor) han acordado las siguientes condiciones de uso:

- Se podrán copiar extractos de este documento, para uso no comercial, siempre que se cite la fuente. Cualquier Proveedor de Formación Acreditado puede utilizar este programa de estudio como fuente para un curso de formación si los autores y el ISTQB® son reconocidos como fuente y propietarios de los derechos de autor del programa de estudio y siempre que cualquier anuncio de dicho curso de formación pueda mencionar el programa de estudio sólo después de haber recibido la acreditación oficial de los materiales de formación por parte de un Comité Miembro reconocido por el ISTQB®.
- Cualquier persona o grupo de personas puede utilizar este programa de estudio como fuente para artículos y libros, si los autores y el ISTQB® son reconocidos como la fuente y los propietarios de los derechos de autor del programa de estudio.
- Cualquier otro uso de este programa de estudio está prohibido sin la aprobación previa por escrito del ISTQB®.
- Cualquier Comité Miembro reconocido por el ISTQB® puede traducir este programa de estudio siempre y cuando reproduzca el mencionado Nota de Derechos de Propiedad Intelectual (Copyright) en la versión traducida del programa de estudio.

## Historial de revisiones

Versión	Fecha	Observaciones
V 01.00	25/07/2025	Entrega CT-GenAI V01.00

Historial de revisiones – Traducción al idioma español		
Versión	Fecha	Observaciones
V ES-01.00	27/11/2025	Entrega para distribución y publicación.
V ES-01.01	04/12/2025	Cambio en lista de agradecimientos de la versión en español.

## Tabla de Contenidos

Nota sobre derechos de propiedad intelectual .....	2
Historial de revisiones .....	3
Tabla de Contenidos .....	5
Agradecimientos .....	7
Notas de la versión en idioma español .....	8
Nota editorial sobre la traducción integral de programas de estudio ISTQB .....	9
Traducciones, abreviaturas y acrónimos .....	10
0 Introducción .....	12
0.1 Objetivo de este programa de estudio .....	12
0.2 Prueba de software con IA Generativa .....	12
0.3 Trayectoria profesional de los probadores .....	12
0.4 Resultados de negocio .....	12
0.5 Objetivos de aprendizaje evaluables, objetivos prácticos y nivel cognitivo de conocimiento 14	
0.6 El examen del certificado de nivel especialista, prueba con IA generativa (IAGen) .....	14
0.7 Acreditación .....	14
0.8 Tratamiento de los estándares .....	15
0.9 Nivel de detalle .....	15
0.10 Organización del programa de estudio .....	15
1 Introducción a la IA generativa para la prueba de software .....	17
1.1 Fundamentos y conceptos clave de la IA generativa .....	19
1.1.1 Espectro de la IA: IA simbólica, aprendizaje automático clásico, aprendizaje profundo e IA generativa .....	19
1.1.2 Conceptos básicos de la IA generativa y de modelos de lenguaje grande (MLG) .....	19
1.1.3 MLG fundacionales, ajustados por instrucciones y de razonamiento .....	21
1.1.4 MLG multimodales y modelos de visión y lenguaje .....	22
1.2 Aprovechamiento de la IA generativa en la prueba de software: Principios Fundamentales 22	
1.2.1 Capacidades clave de los modelos de lenguaje grande (MLG) para tareas de prueba ..	22
1.2.2 Aplicaciones para prueba de software potenciadas por bots IA de conversación y modelos de lenguaje grande MLG .....	23
2 Ingeniería de instrucciones para la prueba de software efectiva .....	25
2.1 Desarrollo eficaz de instrucciones .....	27
2.1.1 Estructura de instrucciones para la IA generativa en la prueba de software .....	27
2.1.2 Técnicas de instrucción básicas para la prueba de software .....	28
2.1.3 Instrucción de sistema e instrucción de usuario .....	29
2.2 Aplicación de técnicas de ingeniería de instrucciones a tareas de prueba de software .....	30
2.2.1 Análisis de la prueba con IA generativa .....	30
2.2.2 Diseño de la prueba e implementación de la prueba con IA generativa .....	31
2.2.3 Prueba de regresión automatizada con IA generativa .....	33
2.2.4 Monitorización de la prueba y control de la prueba con IA generativa .....	35
2.2.5 Selección de técnicas de formulación de instrucciones para la prueba de software .....	36
2.3 Evaluación de los resultados de la IA generativa y mejora de las instrucciones para las tareas de prueba de software .....	36
2.3.1 Métricas para evaluar los resultados de la IA generativa en tareas de prueba .....	37

2.3.2	Técnicas para evaluar y perfeccionar instrucciones de forma iterativa .....	38
3	Gestión de riesgos de la IA generativa en la prueba de software .....	39
3.1	Alucinaciones, errores de razonamiento y sesgos .....	41
3.1.1	Alucinaciones, errores de razonamiento y sesgos en la IA generativa. ....	41
3.1.2	Identificación de alucinaciones, errores de razonamiento y sesgos en los resultados de MLG .....	41
3.1.3	Técnicas de mitigación de alucinaciones, errores de razonamiento y sesgos de IAGen en tareas de prueba de software .....	42
3.1.4	Mitigación del comportamiento no determinista de los MLG .....	43
3.2	Privacidad de datos y riesgos de seguridad de la IA generativa en la prueba de software .....	43
3.2.1	Riesgos para la privacidad de datos y la seguridad asociados al uso de la IA generativa .....	44
3.2.2	Privacidad de datos y vulnerabilidades en la IA generativa para procesos y herramientas de prueba .....	44
3.2.3	Estrategias de mitigación para proteger la privacidad de los datos y mejorar la seguridad en la prueba con IA generativa .....	45
3.3	Consumo de energía e impacto medioambiental de la IA generativa en la prueba de software .....	46
3.3.1	Impacto del uso de IAGen en el consumo energético y las emisiones de CO2 .....	46
3.4	Normativas, estándares y marcos de trabajo de buenas prácticas en materia de IA .....	47
3.4.1	Normativas, estándares y marcos de trabajo relevantes para la IA generativa en la prueba de software .....	47
4	Infraestructura de prueba impulsada por MLG para la prueba de software .....	49
4.1	Enfoques arquitectónicos para la infraestructura de prueba impulsada por MLG .....	51
4.1.1	Componentes y conceptos arquitectónicos clave de la infraestructura de prueba impulsada por MLG .....	51
4.1.2	Generación aumentada por recuperación .....	52
4.1.3	El papel de los agentes impulsados por MLG en la automatización de los procesos de prueba .....	52
4.2	Ajuste fino y OpsMLG: puesta en marcha de la IA generativa para la prueba de software .....	54
4.2.1	Ajuste fino de MLG para tareas de prueba .....	54
4.2.2	Operaciones del modelo de lenguaje grande (OpsMLG) al implementar y gestionar MLG para la prueba de software .....	55
5	Despliegue e integración de IA generativa en organizaciones de prueba .....	56
5.1	Guía para la adopción de la IA generativa en la prueba de software .....	58
5.1.1	Riesgos de IA en la sombra .....	58
5.1.2	Aspectos clave de una estrategia de IA generativa en la prueba de software .....	58
5.1.3	Selección de MLG/MLP para tareas de prueba de software .....	58
5.1.4	Fases de la adopción de la IA generativa en la prueba de software .....	59
5.2	Gestión del cambio al adoptar la IA generativa en la prueba de software .....	60
5.2.1	Habilidades y conocimientos esenciales para probar con IA generativa .....	60
5.2.2	Desarrollo de capacidades de IA generativa en equipos de prueba .....	60
5.2.3	Procesos de prueba en evolución en organizaciones de prueba asistidas por IA .....	60
6	Referencias .....	62



## Agradecimientos

La Asamblea General del ISTQB® publicó formalmente este documento el 25 de julio de 2025.

Ha sido elaborado por un equipo de “International Software Testing Qualifications Board”: Abbas Ahmad (propietario de producto), Gualtiero Bazzana, Alessandro Collino, Olivier Denoo, and Bruno Legeard (director técnico).

El equipo da las gracias a Anne Kramer, Jędrzej Kwapinski, Samuel Ouko and Ina Schieferdecker por su revisión técnica y a los Comités Miembro por sus sugerencias y aportaciones.

Las siguientes personas participaron en la revisión, con comentarios y en la votación de este programa de estudio:

Albert Laura, Aneta Derkova, Anne Kramer, Arda Ender Torçuk, Baris Sarialioglu, Claire Van Der Meulen, Daniel van der Zwan, Derek Young, Dietmar Gehring, Francisca Cano Ortiz, Gary Mogyorodi, Gergely Ágnes, Horst Pohlmann, Ina Schieferdecker, Ingvar Nordström, Jan Sabak, Jaroslav Hryszko, Jędrzej Kwapinski, Joanna Kazun, Karol Frühauf, Katalin Balla, Koray Yitmen, Laura Albert, Linda Vreeswijk, Lucjan Stapp, Lukáš Piška, Mario Winter, Marton Siska, Mattijs Kemmink, Matthias Hamburg, Meile Posthuma, Michael Stahl, Márton Siska, Nele Van Asch, Nils Röttger, Nishan Portoyan, Piet de Roo, Piotr Wicherski, Péter Földházi, Péter Sótér, Radosław Smilgin, Ralf Pichler, Renzo Cerquozzi, Rik Marselis, Samuel Ouko, Stephanie Ulrich, Stuart Reid, Tal Pe'er, Tamás Gergely, Thomas Letzkus, Wim Decoutere, Zsolt Hargitai, Mark Rutz, Patrick Quilter, Earl Burba, Taz Daughtrey, Judy McKay, Randall Rice, Thomas Adams, Tom Van Ongeval, Sander Mol, Miroslav Renda, Geng Chen, Chai Afeng, Xinghan Li, Klaudia Dussa-Zieger, Arnd Pehl, Florian Fieber, Ray Gillespie, József Kreisz, Dénes Medzihradszky, Ferenc Hamori, Giorgio Pisani, Giancarlo Tomasig, Young jae Choi, Arnika Hryszko, Andrei Brovko, Ilia kulakov, Praveen, Kostas Pashalidis, Ferdinand Gramsamer, A. Berfin Öztaş, Abdullah Gök, Abdurrahman AKIN, Aleya Zuhail İŞİK, Anıl Şahin, Atakan Erdemgil, Aysel Bilici, Azmi YÜKSEL, Bilal Gelik, Bilge Yazıcı, Burak Gel, Burcu ÖZEL, Büşra İlayda Çevik Köken, Can Polat, Canan Ayten Dörtkol (Polat), Cansu Mercan Daldaban, Denizcan Orhun Karaca, Didem Çiçek Bay, Duygu Yalçinkaya, Efe Can Yemez, ELİF CERAV, Emine Tekiner, Emre Aman, Emre Can Akgül, Esra Küçük, Gençay GENÇ, Gül Çalışır Açı, Gül Nihal SİNGİL, Güler GÖK, Gulhanım Anulur, Hakan GÜVEZ, Haktan Bilgehan Dilber, Halil İbrahim Tasdemir, Hasan Küçükayar, Hatice Erdoğan, Hatice Kübra Daşdoğan, Hüseyin Sevki ARI, Hyulya Gyuler, İLKNUR NEŞE TUNCAL, Kaan Eminğlu, Kamil Isik, Koray Danışman, Melisa Canbaz, Merve Guleroglu, Müjde Ceylan, Mustafa Furkan CEYLAN, Nergiz Gençaslan, Nuh Soner Bozkurt, Omer Fatih Poyraz, Onur Ersoy, Özlem Körpe, Özgür Özdemir, Sedat YOLTAY, Selahattin Aliyazıcıoğlu, Sevan Lalikoğlu, Sebastian Malyska, Sevim Öykü Demirel, Tatsiana Beliai, Tayg.

## Notas de la versión en idioma español

Este “Programa de Estudio de Probador Certificado del ISTQB®”, Nivel Especialista, Prueba con IA generativa, Versión 1.0” ha sido traducido por Spanish Software Testing Qualifications Board (SSTQB).

La traducción y revisión de este programa de estudio ha estado a cargo de:

Rol	Nombre	Apellido/s	País
Revisor/a:	Aurelio	Gandarillas	(España)
Revisor/a:	Irma	Infante	(Perú)
Responsable de la traducción:	Gustavo	Márquez Sosa	(España)
Revisor/a:	Patricia	Osorio	(Colombia)
Revisor/a:	Daniel	Tolosa	(Chile)

El Comité Ejecutivo del SSTQB agradece toda aportación que permita mejorar esta traducción del programa de estudio.

En una siguiente versión se podrán incorporar aportaciones adicionales. El SSTQB considera conveniente mantener abierta la posibilidad de realizar cambios en el “Programa de Estudio”.

Madrid, 27 de noviembre de 2025



## Nota editorial sobre la traducción integral de programas de estudio ISTQB

Esta traducción al español del programa de estudio de ISTQB persigue maximizar la comprensión, la inclusión y la precisión terminológica para toda la comunidad hispanohablante.

### Principios

- El español/castellano es válido como lengua técnica. La ingeniería y la prueba de software se pueden enseñar, aprender y ejercer en español con rigor.
- SSTQB no pretende imponer el uso del español. La traducción convive con el original en inglés: siempre se identifica el término fuente entre paréntesis, en nota o en un apartado específico, para mantener la trazabilidad y facilitar la búsqueda de referencias bibliográficas.
- Términos significativos. Se priorizan equivalentes claros y estables en español, evitando anglicismos innecesarios y reduciendo ambigüedad.
- Accesibilidad. SSTQB, como asociación española sin fines de lucro, ofrece un texto que no requiere conocimientos de otra lengua para estudiar y prepararse.

### Por qué traducimos todo (términos, acrónimos y abreviaturas)

- **Equidad de acceso:** elimina barreras idiomáticas y sesgos en formación y examen.
- **Coherencia terminológica:** un vocabulario armonizado evita interpretaciones divergentes (p. ej., coverage, test oracle, stub, fixture).
- **Calidad pedagógica:** estudiar en lengua materna mejora comprensión, retención y aplicación práctica.
- **Contextos regulados y contractuales:** numerosos proyectos y pliegos en España y Latinoamérica exigen documentación técnica en español.
- **Interoperabilidad documental:** planes de prueba, criterios de aceptación, informes y métricas se integran con mayor claridad en sistemas de gestión de calidad y contratos redactados en español, manteniendo la forma inglesa como referencia.
- **Mantenimiento y versionado:** un glosario completo y normado facilita actualizaciones y alineación con nuevas ediciones del programa.
- **Uso académico y corporativo:** materiales reutilizables en docencia y formación interna sin pérdida de rigor, con puente explícito al inglés.
- **Siglas sin ambigüedad:** normalizar acrónimos/abreviaturas (y expandir su forma larga) evita colisiones semánticas.
- **Búsqueda eficaz:** incluir ES/EN mejora el hallazgo de información en repositorios, rastreadores de defectos y bases de conocimiento.
- **Buenas prácticas de traducción técnica:** fidelidad al original, respeto del tipo de término (acrónimo/abreviatura), notas justificativas cuando procede y registro de versiones.

### Síntesis

- **Traducir todo no sustituye al inglés:** amplía el acceso, mejora la precisión didáctica y preserva la trazabilidad al original, permitiendo estudiar y trabajar en español con el mismo nivel de exigencia profesional.

## Traducciones, abreviaturas y acrónimos

### Traducción de términos

Español	Inglés
capa de servicios	back-end
director de tecnología	chief technology officer
capa de presentación	front-end
interfaz gráfica de usuario	graphical user interface
espacio de alta dimensión	high-dimensional space
hiperparámetro	hyper-parameter
modelo de lenguaje	language model
MLG como servicio	LLM-as-a-Service
técnica de formulación de instrucciones	prompting technique
generación aumentada por recuperación	retrieval-augmented generation
implicado	stakeholder
toquen	token
toquenización	tokenization
toquenizador	tokenizer
tóquenes	tokens
interfaz de usuario	user interface
agente de prueba impulsado por IAGen	GenAI-powered test agent
herramienta de prueba impulsada por MLG	LLM-powered test tool

### Abreviaturas (ABR) y acrónimos (ACR)

Tipo de Término	Término en español	Tipo de término en español	Tipo de término en inglés	Término en inglés
ACR	inteligencia artificial	IA	AI	artificial intelligence
ACR	interfaz de programación de aplicación	IPA	API	application programming interface
ACR	director de seguridad de la información	DSI	CISO	chief information security officer
ACR	director de tecnología	DT	CTO	chief technology officer
ABR	inteligencia artificial generativa	IAGen	GenAI	generative artificial intelligence

Tipo de Término	Término en español	Tipo de término en español	Tipo de término en inglés	Término en inglés
ACR	interfaz gráfica de usuario	IGU	GUI	graphical user interface
ACR	modelo de lenguaje grande	MLG	LLM	large language model
ABR	operaciones del modelo de lenguaje grande	OpsMLG	LLMOps	large language model operations
ACR	modelo de lenguaje	ML	LM	language model
ACR	aprendizaje automático	AA	ML	machine learning
ACR	generación aumentada por recuperación	GAR	RAG	retrieval-augmented generation
ACR	modelo de lenguaje pequeño	MLP	SLM	small language model
ACR	interfaz de usuario	IU	UI	user interface

## 0 Introducción

### 0.1 Objetivo de este programa de estudio

Este programa de estudio constituye la base para la formación como Probador Certificado del ISTQB®, Nivel Especialista, Prueba con IA Generativa (IAGen). El ISTQB proporciona este programa de estudio en los siguientes términos:

1. A los comités miembro, para traducir a su idioma local y para acreditar a los proveedores de formación. Los Comités Miembro pueden adaptar el programa de estudio a sus necesidades lingüísticas particulares y añadir referencias para adaptarlo a sus publicaciones locales.
2. A los organismos de certificación, para elaborar las preguntas del examen en su lengua local adaptadas a los objetivos de aprendizaje de este programa de estudio.
3. A los proveedores de formación, para desarrollar material didáctico y determinar los métodos de enseñanza adecuados.
4. A los candidatos a la certificación, para que preparen el examen de certificación (ya sea como parte de un curso de formación o de forma independiente).
5. A la comunidad internacional de ingeniería de software y sistemas, para avanzar en la profesión de prueba del software y sistemas, y como fuente de libros y artículos.

### 0.2 Prueba de software con IA Generativa

La certificación “Prueba con IA generativa” está dirigida a cualquier persona involucrada en el uso de IA generativa (IAGen) para la prueba de software. Esto incluye a personas que desempeñan funciones como probadores, analistas de prueba, ingenieros de automatización de prueba, gestores de prueba, probadores especializados en prueba de aceptación de usuarios y desarrolladores de software. La certificación “Prueba con IAGen” también es adecuada para cualquier persona que desee adquirir conocimientos básicos sobre el uso de IAGen para la prueba de software, como gestores de proyectos, gestores de calidad, gestores de desarrollo de software, analistas de negocios, directores de TI y consultores de gestión.

### 0.3 Trayectoria profesional de los probadores

El programa ISTQB® ofrece apoyo a los profesionales de la prueba en todas las etapas de su carrera, proporcionándoles conocimientos amplios y profundos. Las personas que obtengan la certificación de Probador Certificado ISTQB® en Prueba con IA generativa también pueden estar interesadas en los niveles avanzados básicos (analista de prueba, analista de prueba técnica, gestor de prueba e ingeniería de prueba) y, posteriormente, en el nivel experto (gestión de prueba o mejora del proceso de prueba). Visitar [www.istqb.org](http://www.istqb.org) para obtener la información más reciente sobre el programa de probador certificado de ISTQB.

### 0.4 Resultados de negocio

En esta sección se enumeran los resultados de negocio que se esperan de un candidato que haya obtenido la certificación Prueba con IA generativa.

Un candidato que haya obtenido la certificación Prueba con IA generativa puede:

IAGEN-B01	Comprender los conceptos fundamentales, las capacidades y las limitaciones de la IA generativa.
IAGEN-B02	Desarrollar habilidades prácticas para formular instrucciones a modelos de lenguaje grande en la prueba de software.
IAGEN-B03	Adquirir conocimientos sobre los riesgos y las medidas de mitigación del uso de la IA generativa en la prueba de software.
IAGEN-B04	Adquirir conocimientos sobre las aplicaciones de las soluciones de IA generativa en la prueba de software.
IAGEN-B05	Contribuir de manera eficaz a la definición y aplicación de una estrategia y una hoja de ruta referentes al uso de IA generativa en la prueba de software dentro de una organización.



## 0.5 Objetivos de aprendizaje evaluables, objetivos prácticos y nivel cognitivo de conocimiento

Los objetivos de aprendizaje y prácticos respaldan los resultados empresariales y se utilizan para crear exámenes de certificación para Prueba con IA generativa.

En general, todos los contenidos de este programa son examinables en los niveles K1, K2 y K3, excepto la Introducción, los Objetivos prácticos y los Apéndices. Las preguntas del examen confirmarán el conocimiento de las palabras clave en el nivel K1 (véase más abajo) o los objetivos de aprendizaje en todos los niveles K.

Los niveles específicos de los objetivos de aprendizaje se muestran al principio de cada capítulo y se clasifican de la siguiente manera:

- **K1: Recordar**
- **K2: Comprender**
- **K3: Aplicar**

Los objetivos prácticos específicos (HO) se muestran al principio de cada capítulo. Cada HO está vinculado a un LO del nivel K2 o K3, con el objetivo de perfeccionar el aprendizaje a través de la práctica. El nivel de un HO se clasifica de la siguiente manera:

- **H0: Puede incluir una demostración en directo de un ejercicio o un vídeo grabado. Dado que no lo realiza el alumno, no se trata estrictamente de un ejercicio.**
- **H1: Ejercicio guiado. Los alumnos siguen una secuencia de pasos realizados por el formador.**
- **H2: Ejercicio con pistas. Se le da al alumno un ejercicio con pistas relevantes para que pueda resolverlo en el tiempo establecido.**

## 0.6 El examen del certificado de nivel especialista, prueba con IA generativa (IAGen)

El examen para obtener el certificado de **probador certificado en prueba con IA generativa** se basará en este programa. Las respuestas a las preguntas del examen pueden requerir el uso de material basado en más de una sección de este programa. Todas las secciones del programa son evaluables, excepto la introducción, los objetivos prácticos y los apéndices. Las normas, los libros y los artículos se incluyen como referencias, pero su contenido no es evaluable, más allá de lo que se resume en el propio programa de estudio.

Consulte el documento Estructuras y reglas del examen V1.0 para el examen **probador certificado en prueba con IA generativa** para obtener más detalles.

Nota sobre los requisitos de acceso: Se deberá obtener el certificado de nivel básico ISTQB® antes de presentarse al examen de certificación de **probador certificado en prueba con IA generativa**.

## 0.7 Acreditación

Un Comité Miembro de ISTQB® puede acreditar a proveedores de formación cuyos materiales didácticos sigan este programa. Los proveedores de formación deben obtener las directrices de acreditación del Comité Miembro o del organismo que realiza la acreditación. Un curso acreditado se reconoce como conforme a este programa y se le permite incluir un examen ISTQB® como parte del curso.

Las directrices de acreditación para este programa se definen en el documento Directrices de acreditación ISTQB CT - GenAI.

## 0.8 Tratamiento de los estándares

Existen normas relacionadas con las características de calidad y la prueba de software, concretamente las que se mencionan en el programa de estudios del nivel básico, como las de IEEE e ISO. El objetivo de estas referencias es proporcionar un marco de referencia o una fuente de información adicional si el lector lo desea. Se debe tener en cuenta que los programas de estudios utilizan los documentos normativos como referencia. Los documentos normativos no están destinados a ser utilizados en los exámenes. Para obtener más información sobre los estándares, consulte el capítulo 6.

## 0.9 Nivel de detalle

Objetivos de aprendizaje para cada área de conocimiento, que describen los resultados cognitivos que se deben alcanzar.

Una descripción de los conceptos clave, incluyendo referencias a fuentes como literatura o normas aceptadas

Una descripción de cada objetivo práctico de la práctica recomendada para apoyar el aprendizaje

El contenido del programa no es una descripción de toda el área de conocimiento de la prueba con IAGen, sino que refleja el nivel de detalle que se cubrirá en los cursos de formación de Probador Certificado ISTQB® en prueba con IA generativa. Se centra en los conceptos y técnicas de prueba que pueden aplicarse a todos los proyectos de software cuando se utiliza la IA generativa para las pruebas.

El programa utiliza la terminología (es decir, el nombre y el significado) de los términos utilizados en las pruebas de software y el control de calidad según el glosario ISTQB®.

## 0.10 Organización del programa de estudio

Hay 5 capítulos con contenido evaluable. El título principal de cada capítulo especifica la duración del mismo; no se proporciona la duración por debajo del nivel del capítulo. Para los cursos de formación acreditados, el programa requiere un mínimo de 13,6 horas de instrucción, distribuidas entre los 5 capítulos de la siguiente manera:

- **Capítulo 1 (100 minutos):** Introducción a la IA generativa para la prueba de software
  - El probador aprende los fundamentos de los modelos de lenguaje grande (MLG), incluyendo la tokenización y las capacidades multimodales.
  - El probador explora las aplicaciones de la IA generativa (IAGen) en la prueba de software, distinguiendo el chatbot con IA de las herramientas de prueba basadas en MLG, y experimentando con la tokenización, las ventanas de contexto y las instrucciones multimodales.
- **Capítulo 2 (365 minutos):** Ingeniería de instrucciones para la prueba de software efectiva
  - El probador aprende a crear instrucciones efectivas y estructuradas para IAGen en la prueba de software.
  - El probador adquiere experiencia práctica con técnicas de ingeniería de instrucciones para tareas de prueba de software y las aplica.
- **Capítulo 3 (160 minutos):** Gestión de riesgos de la IA generativa en la prueba de software
  - El probador aprende a identificar y mitigar las alucinaciones, los errores de razonamiento y los sesgos al probar con IAGen.
  - El probador aprende a abordar los problemas de privacidad de datos y de seguridad de IAGen en la prueba de software.

- El probador aprende sobre el consumo de energía y el impacto medioambiental de IAGen en la prueba de software.
- El probador aprende las normativas, los estándares y las buenas prácticas de IA para un uso ético, transparente y seguro de IAGen en la prueba de software.
- **Capítulo 4 (110 minutos):** Infraestructura de prueba impulsada por MLG para la prueba de software
  - El probador explora la arquitectura IAGen, como la generación aumentada por recuperación y los agentes de IAGen.
  - El probador aprende el proceso para ajustar los MLG para las tareas de prueba de software.
  - El probador aprende los conceptos de operaciones de modelos de lenguaje grande (OpsMLG) para implementar y gestionar MLG en la prueba de software.
- **Capítulo 5 (80 minutos):** Despliegue e integración de IA generativa en organizaciones de prueba
  - El probador obtiene una guía estructurada para integrar la IA generativa en los procesos de prueba.
  - El probador aprende la transformación organizativa para la integración de la IA generativa en los procesos de prueba.

# 1 Introducción a la IA generativa para la prueba de software

**Duración: 100 minutos**

## Palabras Clave<sup>1</sup>

En la siguiente tabla se presentan las palabras clave del capítulo. En este documento, se identifican dos tipos de palabras clave:

- **ISTQB:** identificarán palabras clave del proceso de prueba
- **ESPDOM:** identificarán palabras clave específicas de dominio: **Inteligencia Artificial**

Tipo Palabra Clave	Español	Inglés
ESPDOM	chatbot con IA	AI chatbot
ESPDOM	ventana de contexto	context window
ESPDOM	aprendizaje profundo	deep learning
ESPDOM	incrustación	embedding
ESPDOM	característica	feature
ESPDOM	MLG fundacional	foundation LLM
ESPDOM	IA generativa	generative AI
ESPDOM	transformador generativo preentrenado	generative pre-trained transformer
ESPDOM	MLG ajustado por instrucciones	instruction-tuned LLM
ESPDOM	modelo de lenguaje grande	large language model
ESPDOM	aprendizaje automático	machine learning
ESPDOM	modelo multimodal	multimodal model
ESPDOM	MLG de razonamiento	reasoning LLM

<sup>1</sup> Las palabras clave se encuentran ordenadas por orden alfabético de los términos en inglés.

## Objetivos de aprendizaje y objetivos prácticos para el “Capítulo 1”

### 1.1 Fundamentos y conceptos clave de la IA generativa

- IAGEN-1.1.1 (K1)** Recordar los diferentes tipos de IA: IA simbólica, aprendizaje automático clásico, aprendizaje profundo e IA generativa.
- IAGEN-1.1.2 (K2)** Explicar los fundamentos de la IA generativa y los modelos de lenguaje grande.
- HO-1.1.2 (H1)** Practicar la segmentación de tóquenes y la evaluación del recuento de tóquenes al utilizar un modelo de lenguaje grande para una tarea de prueba de software
- IAGEN-1.1.3 (K2)** Distinguir entre MLG fundacionales, ajustados por instrucciones y de razonamiento.
- IAGEN-1.1.4 (K2)** Resumir los principios básicos de los modelos de lenguaje grande multimodal y modelos de visión y lenguaje
- HO-1.1.4 (H1)** Escribir y ejecutar una instrucción para un modelo de lenguaje grande multimodal que utilice entradas textuales y de imagen para una tarea de prueba de software.

### 1.2 Aprovechamiento de la IA generativa en la prueba de software: principios básicos

- IAGEN-1.2.1 (K2)** Aportar ejemplos de capacidades clave de los modelos de lenguaje grande para las tareas de prueba
- IAGEN-1.2.2 (K2)** Comparar los modelos de interacción al utilizar IAGen para probar software



## 1.1 Fundamentos y conceptos clave de la IA generativa

La inteligencia artificial generativa (IAGen) es una rama de la inteligencia artificial que utiliza modelos grandes preentrenados para generar salidas similares a las humanas, como texto, imágenes o código. Los modelos de lenguaje grande (MLG) son modelos de IAGen que están preentrenados en grandes conjuntos de datos textuales, lo que les permite determinar el contexto y producir respuestas pertinentes según las instrucciones del usuario.

Los conceptos clave incluyen la tokenización (es decir, dividir/segmentar el texto en unidades para un procesamiento eficiente), las ventanas de contexto (que limitan la cantidad de información que se tiene en cuenta a la vez para mantener la relevancia) y los modelos multimodales (que son capaces de procesar múltiples tipos de datos, como texto, imágenes y audio, para interacciones enriquecidas).

En la prueba de software, estos modelos de lenguaje grande pueden contribuir en diversas tareas, como la revisión y optimización de los criterios de aceptación, la generación de casos de prueba o guiones de prueba, la identificación de posibles defectos, el análisis de patrones de defectos, la creación de datos de prueba sintéticos o la asistencia en la generación de documentación a lo largo de todo el proceso de prueba.

### 1.1.1 Espectro de la IA: IA simbólica, aprendizaje automático clásico, aprendizaje profundo e IA generativa

La inteligencia artificial (IA) es un campo amplio que abarca diferentes tipos de tecnologías, cada una con su propia forma única de resolver problemas, como la IA simbólica, el aprendizaje automático clásico, el aprendizaje profundo y la IAGen (entre otras tecnologías que quedan fuera del alcance de este programa de estudio):

- La **IA simbólica** utiliza un sistema basado en reglas para imitar la toma de decisiones humana. Básicamente, la IA simbólica representa el conocimiento mediante símbolos y reglas lógicas.
- El **aprendizaje automático clásico** es un enfoque basado en datos que requiere la preparación de datos, la selección de características y el entrenamiento de modelos, y que puede utilizarse para tareas como la categorización de defectos y la predicción de problemas software.
- El **aprendizaje profundo** usa redes neuronales, que son estructuras de aprendizaje automático, para aprender automáticamente características a partir de datos. Los modelos de aprendizaje profundo pueden encontrar patrones en conjuntos de datos muy grandes y complejos. Estos pueden ser imágenes, vídeo, audio o texto. No es necesario que los usuarios definan manualmente las características. Aunque en la práctica puede seguir siendo necesaria la intervención humana. Esto puede ser en tareas como la anotación de datos, el ajuste de modelos o la validación de resultados.
- La **IA generativa** utiliza técnicas de aprendizaje profundo para crear nuevos contenidos (texto, imágenes, código) mediante el aprendizaje y la imitación de patrones a partir de sus datos de entrenamiento. Modelos como los modelos de lenguaje grande pueden generar texto, escribir código y simular razonamientos o resolución de problemas dentro del ámbito de su entrenamiento.

En resumen, el campo de la IA ha evolucionado en varias direcciones, cada una con diferentes fortalezas y limitaciones. La ventaja clave de utilizar IAGen para la prueba de software es que utiliza modelos preentrenados que pueden aplicarse directamente a las tareas de prueba sin necesidad de una fase de entrenamiento adicional, aunque esto conlleva algunos riesgos (véase la sección 3.1).

### 1.1.2 Conceptos básicos de la IA generativa y de modelos de lenguaje grande (MLG)

Los modelos de lenguaje grande (MLG), entrenados con conjuntos de datos muy grandes que incluyen libros, artículos y sitios web, se basan en el transformador generativo preentrenado de aprendizaje profundo. Los modelos de lenguaje pequeños (MLP) son modelos compactos con menos parámetros

en comparación con los modelos de lenguaje grande, diseñados para proporcionar soluciones IAGen ligeras y concentradas.

Los modelos de lenguaje grande pueden tratar los matices del lenguaje y generar contenido coherente. Dos conceptos clave que ayudan a los modelos de lenguaje grande a procesar y generar contenido son la tokenización e incrustaciones. La tokenización e incrustaciones convierten el lenguaje en una forma numérica que el modelo puede procesar de manera efectiva.

- La tokenización en los modelos de lenguaje es el proceso de dividir el texto en unidades más pequeñas llamadas tokens. Los tokens pueden ser tan pequeños como un carácter o tan grandes como una subpalabra o una palabra. Cuando un modelo de lenguaje grande (MLG) procesa una frase, primero segmenta la entrada en tokens para que cada uno de ellos pueda entenderse individualmente, al tiempo que se mantiene el contexto general.
- Las incrustaciones son representaciones numéricas de tokens que codifican sus relaciones semánticas, sintácticas y contextuales en un formato adecuado para el procesamiento por parte de modelos de IA generativa. Cada token se transforma en un vector en un espacio de alta dimensión, capturando información matizada sobre su significado y uso. Los tokens con significados o roles contextuales similares tienen incrustaciones que se posicionan muy cerca entre sí en este espacio. Esta proximidad permite a los modelos de lenguaje grande (MLG) comprender las relaciones entre las palabras, conservar el contexto y generar respuestas coherentes y adecuadas al contexto.

Los modelos de lenguaje grande (MLG) utilizan una arquitectura de red neuronal conocida como modelo transformador. Los modelos transformadores destacan en tareas lingüísticas al procesar el contexto de secuencias de texto extensas y aprender cómo se relacionan los tokens entre sí. Durante la inferencia, los MLG predicen el siguiente token de una secuencia, aprovechando estas relaciones aprendidas para generar un texto coherente y adecuado al contexto. El modelo transformador se puede utilizar para generar nuevo texto que sea estadísticamente plausible, basándose en los datos de entrenamiento y la instrucción. Pero lo plausible no es necesariamente correcto.

Los MLG muestran un comportamiento no determinista debido principalmente a la naturaleza probabilística de sus mecanismos de inferencia y a la configuración de los hiperparámetros. Esta aleatoriedad inherente puede dar lugar a variaciones en las salidas, incluso cuando se proporciona la misma entrada varias veces.

En el ámbito de los MLG, la ventana de contexto se refiere a la cantidad de texto precedente, medida en tokens, que el modelo puede tener en cuenta al generar respuestas. Una ventana de contexto más grande permite al modelo mantener la coherencia en pasajes más largos, por ejemplo, al analizar registros de prueba grande. Sin embargo, al aumentar el número de tokens en la ventana de contexto también aumenta la complejidad computacional y el tiempo de procesamiento necesarios para que el modelo funcione con eficacia.

#### Objetivo Práctico HO-1.1.2 (H1): Practicar la tokenización y la evaluación del recuento de tokens

Esta actividad práctica está diseñada para ayudar a los alumnos a desarrollar una comprensión práctica de la tokenización y sus implicaciones cuando se trabaja con MLG. El ejercicio se divide en dos partes principales:

Tokenización: Utilizar un segmentador para dividir un texto de muestra en tokens individuales. Examinar la salida para ver cómo se representan las palabras, la puntuación y las frases, e identificar patrones o matices en la segmentación.

- Tokenización: Utilizar un tokenizador para dividir un texto de muestra en tokens individuales. Examinar la salida para ver cómo se representan las palabras, la puntuación y las frases, e identificar patrones o matices en la tokenización.
- Evaluación del recuento de tokens: Medir el número de tokens generados a partir de varios textos de entrada. Analizar cómo el recuento de tokens influye en el rendimiento del modelo, especialmente en relación con los límites de la ventana de contexto del modelo y las consideraciones de eficiencia.

Al final de este ejercicio, los alumnos serán capaces de anticipar mejor cómo las diferentes estructuras de texto y longitudes de entrada pueden afectar a las interacciones con los MLG.

### 1.1.3 MLG fundacionales, ajustados por instrucciones y de razonamiento

Los modelos de lenguaje grande se desarrollan a través de etapas de entrenamiento progresivamente especializadas para mejorar su efectividad en una amplia gama de tareas. Estas etapas dan lugar a tres categorías principales: modelos fundacionales de lenguaje grande, modelos de lenguaje grande ajustados por instrucciones y modelos de lenguaje grande de razonamiento.

- **MLG fundacionales**

- **Modelos de lenguaje grande (MLG) fundacionales** son modelos de uso general entrenados con conjuntos de datos amplios y diversos que incluyen texto, código, imágenes y otras modalidades. Su extenso preentrenamiento les permite realizar diversas tareas en distintos dominios, como el procesamiento del lenguaje natural, la visión artificial y el reconocimiento de voz. Aunque son potentes y flexibles, los modelos básicos suelen requerir adaptación adicional para cumplir los requisitos específicos de cada tarea.

- **MLG ajustados por instrucciones**

- **Modelos de lenguaje grande (MLG) ajustados por instrucciones:** son obtenidos a partir de los modelos fundacionales, los modelos de lenguaje grande ajustados por instrucciones se ajustan mediante conjuntos de datos que emparejan instrucciones con respuestas esperadas. Esta etapa mejora su alineación con las instrucciones humanas, lo que mejora su usabilidad en aplicaciones del mundo real. El proceso de ajuste implica la optimización del cumplimiento de tareas, el seguimiento de instrucciones y la coherencia de las respuestas, lo que mejora la capacidad del modelo para interpretar y actuar de forma efectiva según la intención del usuario.

- **MLG de razonamiento**

- **Modelos de lenguaje grande de razonamiento u orientados al razonamiento:** los modelos de razonamiento amplían los modelos ajustados por instrucciones enfatizando las capacidades cognitivas estructuradas, como la inferencia lógica, la resolución de problemas en varios pasos y el razonamiento en cadena. Estos modelos se entrenan o ajustan aún más en tareas cuidadosamente seleccionadas que exigen comprensión contextual, pasos de razonamiento intermedios y síntesis de información compleja. Como resultado, son más adecuados para tareas de alta carga cognitiva, incluidas las de dominios técnicos.

En el contexto de las aplicaciones de IAGen para probar software, se utilizan tanto MLG ajustados por instrucciones (a veces denominados “sin razonamiento”) como MLG de razonamiento. La selección



depende de la complejidad y las exigencias de razonamiento de la tarea de prueba específica que se vaya a realizar.

#### 1.1.4 MLG multimodales y modelos de visión y lenguaje

Los MLG multimodales amplían el modelo transformador tradicional para procesar múltiples modalidades de datos, incluyendo texto, imágenes, sonido y vídeo. Estos modelos se entrenan con extensos y diversos conjuntos de datos que les permiten aprender las relaciones entre diferentes tipos de datos. Para el tratamiento de diversas modalidades, la tokenización se adapta a cada tipo de dato; por ejemplo, las imágenes se convierten en incrustaciones utilizando modelos de visión y lenguaje antes de ser procesadas en el modelo transformador.

Los modelos de visión y lenguaje, un subconjunto de los MLG multimodales, integran específicamente información visual y textual para realizar tareas como la creación de pies de foto, la respuesta a preguntas visuales y el análisis de la consistencia entre la información textual y visual.

En la prueba de software, los MLG multimodales, especialmente los MLG aumentados con modelos de visión-lenguaje, ofrecen importantes oportunidades. Pueden analizar elementos visuales de las aplicaciones, como capturas de pantalla y esquemas de interfaces gráficas de usuario (IGU) junto con descripciones textuales asociadas, como informes de defecto o historias de usuarios. Esta capacidad permite a los probadores identificar discrepancias entre los resultados esperados y los elementos visuales reales en una captura de pantalla. Además, los MLG aumentados con modelos de visión y lenguaje pueden generar casos de prueba ricos y realistas que incorporan tanto datos textuales como señales visuales, lo que aumenta la cobertura general.

**Objetivo práctico HO-1.1.4 (H1): Escribir y ejecutar una instrucción para un modelo de lenguaje grande multimodal que utilice entradas textuales y de imagen para una tarea de prueba de software.**

Este ejercicio consiste en **revisar** y **ejecutar** una instrucción dada para un modelo multimodal utilizando entradas de texto e imágenes para resolver una tarea de prueba en dos pasos:

- Revisar las entradas: Revisar la instrucción y los datos de entrada (texto e imagen).
- Ejecutar la instrucción y verificar el resultado: Utilizar un MLG multimodal para introducir tanto la imagen como el texto y comprobar la respuesta del MLG.

Este ejercicio muestra cómo utilizar MLG multimodales para una tarea que implica la introducción de texto e imágenes en casos de uso de pruebas de software, incluyendo el reconocimiento de las ventajas y los posibles retos que ello conlleva.

## 1.2 Aprovechamiento de la IA generativa en la prueba de software: Principios Fundamentales

IAGen proporciona capacidades transformadoras en diversas actividades de prueba. Los MLG destacan en el procesamiento del lenguaje natural y el código, la generación de texto y código coherentes, la respuesta a preguntas, el resumen de información, la traducción de idiomas y el análisis de imágenes en un contexto multimodal.

Los profesionales que se dedican a la prueba en los diversos roles pueden aprovechar IAGen de dos maneras complementarias: a través de los bots de conversación de IAGen, que proporcionan respuestas instantáneas a las consultas, y a través de aplicaciones basadas en MLG integradas en las herramientas de prueba.

### 1.2.1 Capacidades clave de los modelos de lenguaje grande (MLG) para tareas de prueba

Los MLG pueden interpretar requisitos, especificaciones, capturas de pantalla, código, casos de prueba e informes de defecto, lo que los convierte en herramientas para comprender y aclarar la información necesaria a lo largo del proceso de prueba y generar elementos de productos de prueba.

A continuación se presentan algunas de las capacidades clave de los MLG relevantes para la prueba de software:

- **Análisis y mejora de requisitos:** los MLG pueden ayudar a analizar los requisitos y otros elementos de las bases de prueba, identificando ambigüedades, inconsistencias o información faltante. Pueden generar preguntas significativas para ayudar a aclarar los requisitos durante las discusiones con los implicados.
- **Apoyo a la creación de casos de prueba:** los MLG pueden ayudar a generar casos de prueba y sugerir objetivos de prueba basados en los requisitos del sistema, las historias de los usuarios o cualquier otro elemento de la base de prueba.
- **Generación de oráculos de prueba:** los MLG pueden ayudar a generar los resultados esperados.
- **Generación de datos de prueba:** los MLG pueden generar conjuntos de datos, establecer valores límite y crear diferentes combinaciones de datos de prueba.
- **Soporte para la automatización de la prueba:** los MLG pueden ayudar a generar guiones de prueba a partir de la descripción del caso de prueba y mejorar los guiones de prueba existentes sugiriendo cambios e identificando las técnicas de diseño de pruebas adecuadas.
- **Análisis de resultados de la prueba:** los MLG pueden ayudar a analizar los resultados de la prueba creando resúmenes y clasificando las anomalías en función de su severidad y prioridad.
- **Creación de productos de prueba:** los MLG pueden ayudar a crear diversos documentos, incluidos planes de prueba, informes de prueba e informes de defecto, y mantenerlos actualizados a medida que avanza el proyecto.

Estas capacidades demuestran cómo los MLG pueden influir en diversos aspectos de la prueba de software a lo largo de todo el proceso de prueba.

### **1.2.2 Aplicaciones para prueba de software potenciadas por bots IA de conversación y modelos de lenguaje grande MLG**

Los chatbot con IA y las aplicaciones de prueba basadas en MLG pueden ayudar a los probadores, aunque difieren en cuanto a funcionalidad, flexibilidad y enfoques de integración.

Los chatbot con IA proporcionan una interfaz de conversación fácil de usar que permite a los probadores comunicarse directamente con los MLG. Esta interacción en lenguaje natural permite a los probadores introducir preguntas, comandos o instrucciones y recibir respuestas inmediatas y contextualmente conscientes. Mediante técnicas como el encadenamiento de instrucciones, los probadores pueden refinar iterativamente las salidas, lo que hace que los bot de conversación sean especialmente efectivos para tareas rutinarias, pruebas exploratorias e incluso para incorporar a nuevos probadores, ya que proporcionan un acceso rápido a los conocimientos y prácticas de prueba.

Estos bots IA de conversación son especialmente beneficiosos en situaciones que requieren una retroalimentación rápida, la aclaración de conceptos de prueba o la exploración dinámica de requisitos y posibles casos de prueba. Su interfaz intuitiva los hace accesibles incluso para las partes interesadas sin conocimientos técnicos, lo que amplía la base de usuarios potenciales y fomenta una mayor adopción.

Las aplicaciones de pruebas basadas en MLG, por el contrario, implican la integración de capacidades MLG a través de interfaz de programación de aplicación (IPA) para realizar tareas de prueba bien definidas y, a menudo, automatizadas. Estas aplicaciones ofrecen una mayor personalización y escalabilidad, lo que permite a las organizaciones y a los proveedores de herramientas integrar la IA generativa en los marcos de pruebas existentes. Esto permite la automatización de tareas repetitivas o complejas, como la generación de casos de prueba, el análisis de defectos o la síntesis de datos de prueba. En implementaciones más avanzadas, las organizaciones pueden crear agentes de IA diseñados específicamente para desempeñar determinados roles de prueba (véase el capítulo 4).



Independientemente de cómo interactúe el probador con los MLG, ya sea a través de bots IA de conversación o de aplicaciones dotadas de MLG integrados, la implementación satisfactoria de la IA generativa en la prueba requiere una sólida ingeniería de instrucciones (véase el capítulo 2). Las instrucciones cuidadosamente diseñadas, claras y específicas son esenciales para asegurar que las salidas generadas por los MLG sean precisas, relevantes y estén alineadas con los objetivos de la prueba. Esta práctica ayuda a maximizar el valor derivado de la IA generativa y garantiza un apoyo consistente y fiable para una amplia gama de actividades de prueba.

## 2 Ingeniería de instrucciones para la prueba de software efectiva

Duración: 365 minutos

### Palabras Clave<sup>2</sup>

En la siguiente tabla se presentan las palabras clave del capítulo. En este documento, se identifican dos tipos de palabras clave:

- **ISTQB:** identificarán palabras clave del proceso de prueba
- **ESPDOM:** identificarán palabras clave específicas de dominio: **Inteligencia Artificial**

Tipo Palabra Clave	Español	Inglés
ISTQB	criterios de aceptación	acceptance criteria
ISTQB	guion de prueba	test script
ISTQB	caso de prueba	test case
ISTQB	condición de prueba	test condition
ISTQB	datos de prueba	test data
ISTQB	diseño de prueba	test design
ISTQB	informe de prueba	test report
ESPDOM	formulación de instrucciones con pocos ejemplos	few-shot prompting
ESPDOM	metainstrucción	meta-prompting
ESPDOM	procesamiento del lenguaje natural (PLN)	natural language processing (NLP)
ESPDOM	formulación de instrucciones con un ejemplo	one-shot prompting
ESPDOM	instrucción	prompt
ESPDOM	encadenamiento de instrucciones	prompt chaining
ESPDOM	ingeniería de instrucciones	prompt engineering
ESPDOM	instrucción de sistema	system prompt
ESPDOM	instrucción de usuario	user prompt
ESPDOM	formulación de instrucciones sin ejemplos	zero-shot prompting

<sup>2</sup> Las palabras clave se encuentran ordenadas por orden alfabético de los términos en inglés.

## Objetivos de aprendizaje y objetivos prácticos para el “Capítulo 2”

### 2.1 Desarrollo eficaz de instrucciones

- IAGEN-2.1.1 (K2)** Aportar ejemplos de la estructura de las instrucciones utilizadas en la IA generativa para la prueba de software.
- HO-2.1.1 (H0)** Observar varias instrucciones dadas para tareas de prueba de software, identificando los componentes de roles, contexto, instrucción, datos de entrada, restricciones y formato de salida en cada una de ellas.
- IAGEN-2.1.2 (K2)** Diferenciar las técnicas básicas de instrucción para prueba de software.
- HO-2.1.2a (H0)** Observar demostraciones de encadenamiento de instrucciones, formulación de instrucciones con pocos ejemplos y metainstrucción aplicadas a tareas de prueba de software.
- HO-2.1.2b (H1)** Identificar qué técnicas de ingeniería de instrucciones se están utilizando en ejemplos concretos.
- IAGEN-2.1.3 (K2)** Distinguir entre instrucciones de sistema e instrucciones de usuario.

### 2.2 Aplicación de técnicas de ingeniería de instrucciones a tareas de prueba de software

- IAGEN-2.2.1 (K3)** Aplicar la IA generativa a las tareas de análisis de prueba.
- HO-2.2.1a (H2)** Llevar a cabo instrucción multimodal para generar criterios de aceptación para una historia de usuario basada en un esquema de interfaz gráfica de usuario.
- HO-2.2.1b (H2)** Llevar a cabo encadenamiento de instrucciones y verificación humana para analizar de forma progresiva una historia de usuario determinada y perfeccionar los criterios de aceptación
- IAGEN-2.2.2 (K3)** Aplicar la IA generativa al diseño de pruebas y a las tareas de implementación de pruebas
- HO-2.2.2a (H2)** Practicar la generación de casos de prueba funcionales a partir de historias de usuario con IA generativa utilizando encadenamiento de instrucciones, instrucciones estructuradas y meta-instrucciones
- HO-2.2.2b (H2)** Utilizar la técnica de formulación de instrucciones con pocos ejemplos para generar condiciones de prueba y casos de prueba al estilo Gherkin a partir de historias de usuario.
- HO-2.2.2c (H2)** Utilizar el encadenamiento de instrucciones para priorizar los casos de prueba dentro de un juego de prueba determinado, teniendo en cuenta sus prioridades y dependencias específicas.
- IAGEN-2.2.3 (K3)** Aplicar la IA generativa a la prueba de regresión automatizada.
- HO-2.2.3a (H2)** Practicar la formulación de instrucciones con pocos ejemplos para crear y gestionar guiones de prueba basados en palabras clave.
- HO-2.2.3b (H2)** Practicar la ingeniería de instrucciones estructuradas para el análisis de informes de prueba.
- IAGEN-2.2.4 (K3)** Aplicar la IA generativa a las tareas de control y supervisión de la prueba.

**HO-2.2.4 (H0)** Observar las métricas de supervisión de la prueba preparadas por la IA generativa a partir de los datos de prueba.

**IAGEN-2.2.5 (K3)** Seleccionar y aplicar técnicas de instrucción adecuadas para un contexto y una tarea de prueba determinados.

**HO-2.2.5 (H1)** Seleccionar y aplicar técnicas de instrucción adecuadas al contexto para una tarea de prueba determinada

### 2.3 Evaluación de los resultados de la IA generativa y mejora de las instrucciones para las tareas de prueba de software

**IAGEN-2.3.1 (K2)** Comprender las métricas para evaluar los resultados de la IA generativa en las tareas de prueba.

**HO-2.3.1 (H0)** Observar cómo se pueden utilizar las métricas para evaluar el resultado de la IA generativa en una tarea de prueba.

**IAGEN-2.3.2 (K2)** Aportar ejemplos de técnicas para evaluar y perfeccionar iterativamente las instrucciones.

**HO-2.3.2 (H1)** Evaluar y optimizar una instrucción para una tarea de prueba determinada.

## 2.1 Desarrollo eficaz de instrucciones

El diseño eficaz de instrucciones asegura que las herramientas IAGen realicen las tareas de prueba de software con precisión y eficiencia, y que los probadores obtengan resultados útiles de los MLG. Una instrucción estructurada incluye diferentes componentes (véase la sección 2.1.1). Cada uno de estos componentes contribuye a la claridad y precisión de una instrucción que comunica eficazmente los requisitos y expectativas a los MLG.

Diversas técnicas de ingeniería de instrucciones mejoran la efectividad de las instrucciones en la prueba de software. Técnicas como el encadenamiento de instrucciones, la formulación de instrucciones con pocos ejemplos y la metainstrucción ayudan a abordar retos de prueba complejos (véase la sección 2.1.2).

La combinación de instrucciones estructuradas (véase la sección 2.1.1) con técnicas básicas de instrucción tiene como objetivo obtener buenos resultados al consultar un modelo de lenguaje grande para tareas de prueba de software (véase la sección 2.1.3).

### 2.1.1 Estructura de instrucciones para la IA generativa en la prueba de software

Una instrucción estructurada para la prueba de software suele incluir seis componentes:

- **Rol:** El rol define la perspectiva o personalidad que debe adoptar el modelo IAGen al generar una respuesta. Especificar el rol ayuda al MLG a determinar sus responsabilidades y adoptar un tono o enfoque adecuado, como actuar como probador, jefe de prueba o ingeniero de automatización de la prueba.
- **Contexto:** El contexto proporciona los antecedentes (información de contexto) que el modelo IAGen necesita para determinar las condiciones de prueba. Esto incluye detalles sobre el objeto de prueba, la funcionalidad específica que se va a probar y cualquier información contextual relevante.
- **Instrucciones:** Las instrucciones son directivas que se dan a la IAGen y que describen la tarea específica que se debe realizar. Las instrucciones claras, imperativas y concisas incluyen una descripción de la tarea y cualquier requisito relevante para la misma.

- **Datos de entrada:** Los datos de entrada incluyen cualquier información necesaria para realizar la tarea, como historias de usuarios, criterios de aceptación, capturas de pantalla, código, casos de prueba existentes o ejemplos de salida. Proporcionar datos de entrada detallados y estructurados ayuda al MLG a generar resultados más precisos y sensibles al contexto.
- **Restricciones:** Las restricciones describen cualquier limitación o consideración especial que el MLG deba respetar. Las restricciones ayudan a especificar cómo deben aplicarse las instrucciones a los datos de entrada.
- **Formato de salida:** Las especificaciones de salida indican el formato, la estructura o las características esperadas de la respuesta. Estos indicadores ayudan a dar forma a la salida del MLG.

Estos componentes forman la estructura básica de la instrucción. Esta estructura debe combinarse con la implementación de técnicas de instrucción (véase la sección 2.1.2), dependiendo de la tarea que se vaya a realizar y del modelo de lenguaje grande que se vaya a utilizar.

**HO-2.1.1 - (H0) - Observar varias instrucciones dadas para tareas de prueba de software, identificando los componentes de roles, contexto, instrucción, datos de entrada, restricciones y formato de salida en cada una de ellas.**

En una demostración, se experimenta con varias instrucciones estructuradas en un chatbot con IA, cada una de ellas adaptada a tareas específicas de prueba de software. Estas instrucciones siguen un formato estructurado que consta de seis componentes clave: rol, contexto, instrucción, datos de entrada, restricciones y formato de salida. La demostración tiene por objeto facilitar la observación y el análisis de estas instrucciones estructuradas, destacando cómo cada componente contribuye a proporcionar información precisa, relevante y útil a un modelo de lenguaje grande utilizado para una tarea de prueba de software.

## 2.1.2 Técnicas de instrucción básicas para la prueba de software

En los últimos años, se han propuesto muchas técnicas de formulación de instrucciones para MLG para diferentes casos de uso de IAGen (Schulhoff 2024). Entre ellas, hay tres técnicas de formulación de instrucciones que se utilizan habitualmente para tareas de prueba con IAGen junto con la estructura de instrucciones de seis componentes descrita anteriormente (véase la sección 2.1.1): el encadenamiento de instrucciones, la formulación de instrucciones con pocos ejemplos y metainstrucción.

- El encadenamiento de instrucciones consiste en dividir una tarea en una serie de pasos intermedios (múltiples instrucciones). El resultado de cada paso se comprueba y se perfecciona de forma manual o automática antes de pasar al siguiente paso. Este enfoque conduce a una mayor precisión, ya que cada respuesta informa a la siguiente instrucción. El encadenamiento de instrucciones es especialmente útil en procesos de prueba en los que las tareas son complicadas y requieren la descomposición en subtareas y la comprobación sistemática de los resultados intermedios del MLG. También permite interacciones dinámicas en los procesos de prueba.
- La formulación de instrucciones con pocos ejemplos consiste en proporcionar al MLG ejemplos en la instrucción. Mientras que la formulación de instrucciones sin ejemplos se basa en el conocimiento preexistente del modelo para generar una respuesta, la formulación de instrucciones con un ejemplo proporciona un ejemplo para demostrar el resultado deseado para una entrada determinada. Las instrucciones de pocos ciclos contienen más de un ejemplo (unos pocos) para consolidar aún más el comportamiento de respuesta deseado del modelo.

Esta técnica ayuda a guiar el modelo proporcionando una referencia clara y garantizando que los resultados sean coherentes y se ajusten a las expectativas. La formulación de instrucciones con pocos ejemplos es especialmente eficaz para tareas en las que los ejemplos pueden ilustrar el comportamiento requerido, lo que permite al modelo generalizar de forma eficaz y producir resultados fiables.

- La metainstrucción aprovecha la capacidad de la IA para generar o perfeccionar sus propias instrucciones. En un ciclo iterativo, el MLG puede generar instrucciones que pueden ser evaluadas



y perfeccionadas por el probador. Este enfoque optimiza la calidad de las instrucciones al aprovechar el conocimiento del MLG sobre instrucciones optimizadas. La metainstrucción es especialmente beneficiosa cuando la eficiencia y la optimización de las instrucciones son fundamentales, ya que reduce el esfuerzo manual necesario para diseñar instrucciones eficaces.

Otra ventaja de las metainstrucciones es que, si el probador no está seguro de cómo crear una instrucción eficaz, puede colaborar con el MLG para crearla conjuntamente. Esto refleja una forma de trabajo en pareja con la herramienta IAGen en la que el probador y la IA trabajan juntos de forma interactiva para lograr un objetivo común. Este concepto de trabajo en pareja pone de relieve una nueva forma de colaborar con las herramientas de IA, mejorando tanto la productividad como el aprendizaje, no solo en la ingeniería de instrucciones, sino también en la programación en pareja y la prueba en pareja.

Estas técnicas de formulación de instrucciones pueden utilizarse de manera eficaz en combinación para mejorar los resultados del MLG (véase la sección 2.2.5).

**HO-2.1.2a - (H0) - Observar demostraciones de encadenamiento de instrucciones, la formulación de instrucciones con pocos ejemplos y metainstrucción aplicadas a tareas de prueba de software.**

Los participantes experimentarán con el encadenamiento de instrucciones, la formulación de instrucciones con pocos ejemplos y la metainstrucción en un chatbot con IA, cada uno aplicado a tareas específicas de prueba de software. La demostración tiene como objetivo explorar y debatir estas técnicas de formulación de instrucciones en el contexto de la prueba de software, haciendo énfasis en cómo cada técnica contribuye a la precisión y la integridad de los resultados de MLG.

**HO-2.1.2b - (H1) - Identificar qué técnicas de ingeniería de instrucciones se están utilizando en ejemplos concretos.**

Los participantes leerán una serie de ejemplos de instrucciones relacionadas con la prueba de software para identificar las técnicas de instrucción básicas aplicadas. Se hace énfasis en el reconocimiento de técnicas como el encadenamiento de instrucciones, la formulación de instrucciones con pocos ejemplos y la metainstrucción, al tiempo que se destacan sus características distintivas y sus aplicaciones prácticas.

Esta actividad tiene como objetivo profundizar en la comprensión de los participantes sobre cómo las diferentes técnicas de formulación de instrucciones mejoran el uso eficaz de IAGen en las pruebas de software.

### 2.1.3 Instrucción de sistema e instrucción de usuario

Las instrucciones del sistema y las instrucciones del usuario tienen diferentes propósitos en las interacciones con los MLG, y cada una desempeña un rol distinto en la configuración de la conversación. La instrucción de sistema suele ser definida por el desarrollador o el probador, para guiar el comportamiento general del MLG, y no es visible ni editable por el usuario del bot de chat en la mayoría de las interfaces.

Una instrucción de sistema actúa como un conjunto de comandos predefinidos que define el comportamiento, la personalidad y los parámetros operativos del MLG. Los parámetros operativos determinan cómo responde el MLG, por ejemplo, utilizando un tono formal, manteniendo respuestas concisas, respetando las reglas específicas del dominio o evitando ciertos comportamientos. La instrucción de sistema establece las reglas para toda la conversación. Puede contener partes de una instrucción estructurada, como el rol, el contexto y las restricciones.

La instrucción de sistema permanece constante durante toda la sesión de interacción y establece el marco de trabajo fundamental sobre cómo debe responder el MLG. Por ejemplo, una instrucción de sistema podría decir: “Usted es un asistente profesional de pruebas de software. Responda siempre

con claridad, utilice un lenguaje formal y concéntrese en las prácticas alineadas con ISTQB. Evite las especulaciones y cite los principios de la prueba cuando sea relevante”.

Por otro lado, la instrucción de usuario representa la entrada o pregunta real del usuario del chatbot con IA. Cambia con cada interacción y puede incluir instrucciones específicas, preguntas o tareas que el usuario del bot de conversación desea que el MLG aborde. A diferencia de la instrucción de sistema, las instrucciones de usuario son directamente visibles y forman el contexto inmediato de cada respuesta.

Por ejemplo, una instrucción de usuario podría ser: “Enumera las diferencias clave entre las pruebas de caja negra y caja blanca con ejemplos”.

El uso típico consiste en configurar la instrucción de sistema una vez al inicio de la conversación y, a continuación, enviar sucesivas instrucciones de usuario para cada interacción. El MLG genera respuestas teniendo en cuenta tanto la instrucción de sistema, que no cambia, como la instrucción de usuario actual. Para una implementación eficaz, las instrucciones del sistema deben ser claras y específicas sobre la función del MLG y las posibles restricciones. También pueden contener contexto e instrucciones generales, por ejemplo, sobre el resultado esperado.

Las instrucciones de usuario deben ser concisas y estar bien estructuradas, e incluir instrucciones explícitas, así como información adicional relevante sobre el contexto y el formato de salida.

## 2.2 Aplicación de técnicas de ingeniería de instrucciones a tareas de prueba de software

La aplicación de técnicas de ingeniería de instrucciones a la prueba de software permite a la IAGen apoyar tareas de prueba como el análisis de prueba, el diseño de prueba, la automatización de prueba, la priorización de casos de prueba, la detección de defectos, el análisis de cobertura y la supervisión y el control de la prueba. Mediante el uso y la combinación de técnicas como el encadenamiento de instrucciones, la formulación de instrucciones con pocos ejemplos y la metainstrucción, los equipos pueden adaptar las instrucciones de IA a los objetivos específicos de la prueba, lo que hace que los resultados sean más precisos, relevantes y eficaces. Una entrada de alta calidad es fundamental para obtener resultados significativos de la IA.

### 2.2.1 Análisis de la prueba con IA generativa

La IAGen puede apoyar las tareas de análisis de prueba generando y priorizando condiciones de prueba, identificando defectos en la base de prueba y proporcionando análisis de cobertura. Los datos de entrada incluyen requisitos, historias de usuario, especificaciones técnicas, esquemas de la interfaz gráfica de usuario y otra información relevante. El resultado consiste en productos típicos del trabajo de análisis de pruebas, como condiciones de prueba priorizadas (por ejemplo, criterios de aceptación).

A continuación se indican algunas tareas típicas del análisis de prueba que pueden ser asistidas por IAGen:

- **Identificar posibles defectos en la base de prueba:** IAGen puede ayudar a analizar la base de prueba en busca de inconsistencias, ambigüedades o información incompleta que podrían dar lugar a defectos. Al comparar patrones de requisitos similares o aplicar conocimientos de informes de defectos anteriores, el MLG puede señalar posibles anomalías y sugerir mejoras.
- **Generación de condiciones de prueba basadas en la base de la prueba,** por ejemplo, en requisitos/historias de usuario: los MLG pueden analizar los requisitos y las historias de usuario para generar condiciones de prueba. Mediante el procesamiento del lenguaje natural (PLN), pueden interpretar el significado de los requisitos y desglosarlos en enunciados medibles y comprobables. Esto puede ayudar a traducir los requisitos en condiciones de prueba específicas.
- **Priorizar las condiciones de prueba en función del nivel de riesgo:** con información sobre la probabilidad de riesgo y el impacto del riesgo de fallo para cada condición de prueba, un MLG puede ayudar a priorizar el esfuerzo de prueba. Teniendo en cuenta aspectos como el

cumplimiento normativo, las características orientadas al usuario (por ejemplo, la funcionalidad de inicio de sesión o el procesamiento de pagos) y los datos históricos de defectos, el MLG puede recomendar niveles de prioridad.

- **Soporte al análisis de cobertura:** Al asignar los requisitos y las historias de usuario a las condiciones de prueba, un MLG puede realizar un análisis de cobertura para determinar si se cubren todos los aspectos de la base de prueba. Esto resulta especialmente útil en proyectos con requisitos complejos, en los que las lagunas en la cobertura pueden dar lugar a defectos no detectados.
- **Sugerir técnicas de prueba:** IAGen puede sugerir técnicas de prueba relevantes (por ejemplo, análisis de valores límite, partición por equivalencia) en función del tipo de requisito o historia de usuario que se esté probando. Esto puede ayudar a los probadores a aplicar las técnicas de prueba más eficaces para condiciones de prueba específicas.

La calidad y la relevancia de los datos proporcionados al MLG en relación con la tarea que se debe completar influyen directamente en la exactitud y precisión de los resultados generados por el MLG.

#### **HO-2.2.1a - (H2) - Llevar a cabo instrucción multimodal para generar criterios de aceptación para una historia de usuario basada en un esquema de interfaz gráfica de usuario.**

Este es un ejercicio para practicar la redacción de instrucciones estructuradas utilizando entradas multimodales (texto e imagen). El objetivo es generar criterios de aceptación de alta calidad (es decir, bien formados, claros y completos) a partir de una historia de usuario y un esquema de interfaz gráfica de usuario. Se pueden añadir otros elementos de texto para proporcionar contexto, como restricciones en los campos de entrada o reglas de negocio que se aplicarán al procesamiento de datos.

Los resultados obtenidos del MLG se comparan para evaluar el impacto de diferentes formulaciones de la instrucción estructurada (rol, contexto, instrucción, datos de entrada textuales y de imagen, restricciones y formato de salida) para una tarea de análisis de prueba.

Este ejercicio proporciona experiencia práctica sobre la importancia de la estructuración de las instrucciones, la contribución de las instrucciones precisas y la importancia de los datos contextuales tanto textuales como de imagen para obtener resultados precisos y relevantes del MLG.

#### **HO-2.2.1b - (H2) - Llevar a cabo encadenamiento de instrucciones y verificación humana para analizar de forma progresiva una historia de usuario determinada y perfeccionar los criterios de aceptación**

Este es un ejercicio para practicar el encadenamiento de instrucciones con el fin de analizar una historia de usuario dada y refinar los criterios de aceptación, primero identificando las ambigüedades, luego evaluando la capacidad de prueba y, finalmente, evaluando la integridad. Este ejercicio fomenta un enfoque paso a paso, refinando el análisis en cada paso para garantizar que los criterios de aceptación estén bien formulados y sean viables para alcanzar los objetivos de la prueba. En cada paso, los resultados proporcionados por el MLG se verifican manualmente y se corrigen, si es necesario, ya sea ajustando el resultado o mediante un proceso de encadenamiento de instrucciones con el MLG. De esta manera, la siguiente etapa utiliza un resultado limpio de la etapa anterior para abordar otro aspecto de la mejora de los criterios de aceptación.

Este ejercicio proporciona experiencia práctica sobre las ventajas de dividir una tarea compleja en subtareas, con verificación humana de los resultados de cada etapa.

## **2.2.2 Diseño de la prueba e implementación de la prueba con IA generativa**

Diseño de la prueba e implementación de la prueba con IA generativa

Tal y como se describe en [ISTQB\_CTFL\_SYL], el diseño de prueba implica la elaboración y el perfeccionamiento de las condiciones de prueba, que luego se traducen en casos de prueba y otros productos de prueba. La implementación de la prueba conlleva la creación o adquisición de los productos de prueba necesarios para realizar las pruebas.

Tanto las pruebas manuales como los guiones de prueba automatizados se pueden crear, priorizar y organizar dentro de un calendario de ejecución de pruebas con el apoyo de IAGen. IAGen puede

apoyar de manera significativa este amplio grupo de actividades de prueba, ayudando en la creación y evaluación de diversos productos de prueba, incluyendo casos de prueba, datos de prueba, guiones de prueba y entornos de prueba.

A continuación se muestran algunas tareas típicas del diseño de pruebas y la implementación de pruebas que pueden ser asistidas por IAGen:

- **Generación de casos de prueba:** el procesamiento del lenguaje natural (PLN) permite a IAGen crear borradores de casos de prueba basados en requisitos funcionales y no funcionales. Cuando se le proporciona la información adecuada, un MLG puede sugerir precondiciones y entradas, resultados esperados y criterios de cobertura, produciendo casos de prueba que cumplen diferentes objetivos de prueba, desde la verificación funcional básica hasta pruebas complejas de extremo a extremo.
- **Síntesis de datos de prueba:** IAGen puede crear datos de prueba sintéticos representativos que preservan la privacidad de los datos y se asemejan a los datos de producción, cubriendo situaciones extremas y diversas condiciones de prueba. Estos datos de prueba sintéticos se pueden utilizar para pruebas funcionales y no funcionales. Los datos de prueba generados por IA se pueden adaptar a los requisitos de la aplicación, simulando escenarios realistas sin exponer información confidencial.
- **Generación de guiones de prueba automatizados:** IAGen puede generar procedimientos de prueba manuales y guiones de prueba automatizados a partir de casos de prueba estructurados, interpretando los pasos de la prueba y traduciéndolos a código compatible con diversos marcos de automatización de prueba. Estos guiones de prueba pueden actualizarse o ampliarse en función de los nuevos requisitos.
- **Programación y priorización de la ejecución de la prueba:** IAGen puede analizar los casos de prueba y sus interdependencias, optimizando los programas de ejecución de las pruebas en función de la prioridad, los riesgos asociados, la disponibilidad de recursos y los objetivos de la prueba.

#### **HO-2.2.2a - (H2) - Practicar la generación de casos de prueba funcionales a partir de historias de usuario con IA generativa utilizando encadenamiento de instrucciones, instrucciones estructuradas y meta-instrucciones**

Este ejercicio se centra en desarrollar casos de prueba funcionales a partir de historias de usuarios con IAGen, utilizando técnicas de encadenamiento de instrucciones, instrucciones estructuradas y metainstrucciones para garantizar una cobertura profunda. El primer paso es crear una instrucción que indique a la IA que genere casos de prueba funcionales basados en criterios de aceptación dados, siguiendo un formato de salida específico. El segundo paso es verificar la integridad de los casos de prueba generados. En este caso, la instrucción verifica que se cubra cada criterio de aceptación haciendo que la IA genere una tabla que resuma la cobertura. Por último, el tercer paso es crear una metainstrucción para ayudar en la creación de procedimientos de prueba de extremo a extremo. Esta metainstrucción ayuda a refinar la instrucción para generar pruebas de extremo a extremo completas, lo que fomenta las mejoras iterativas para maximizar la eficacia.

Este ejercicio mejora la comprensión del uso de los MLG para la generación de casos de prueba, la validación de la cobertura y las pruebas de extremo a extremo.

#### **HO-2.2.2b - (H2) - Utilizar la técnica de formulación de instrucciones con pocos ejemplos para generar condiciones de prueba y casos de prueba al estilo Gherkin a partir de historias de usuario.**

Este ejercicio trata sobre el uso de instrucciones de pocos ciclos para generar casos de prueba al estilo Gherkin a partir de historias de usuario determinadas. Comenzando con una revisión de ejemplos predefinidos y la sintaxis Gherkin, el paso 1 consiste en seleccionar n ejemplos para incluir en la instrucción, cada uno con una historia de usuario, condiciones de prueba y casos de



prueba esperados del estilo “dado-cuando-entonces” para modelar el resultado deseado. A continuación, esta instrucción se aplica a una nueva historia de usuario, generando escenarios Gherkin que reflejan las condiciones de prueba originales. Si los resultados no son correctos, se debe perfeccionar la instrucción o los ejemplos.

Este ejercicio ayuda a adquirir experiencia en la aplicación de técnicas de formulación de instrucciones con pocos ejemplos a tareas de diseño e implementación de pruebas realistas.

**HO-2.2.2c - (H2) - Utilizar el encadenamiento de instrucciones para priorizar los casos de prueba dentro de un juego de prueba determinado, teniendo en cuenta sus prioridades y dependencias específicas.**

Este ejercicio se centra en el uso de IAGen para mejorar la priorización de casos de prueba dentro de un conjunto de pruebas determinado, con el análisis de riesgos asociado y las dependencias entre los casos de prueba. La sesión comienza con una breve descripción general de los diferentes enfoques de prueba, como los basados en el riesgo, la cobertura y los requisitos, y una revisión del conjunto de pruebas dado. A continuación, los participantes se dedicarán a crear instrucciones para generar planes de priorización viables para diversas estrategias de priorización de prueba. Los resultados del MLG basados en la instrucción y los datos de entrada dados deben verificarse manualmente para detectar cualquier error en el razonamiento del MLG.

El objetivo de este ejercicio es experimentar con IAGen en tareas de prueba que requieren capacidades de razonamiento multicriterio (en este caso, los diferentes riesgos y dependencias que deben tenerse en cuenta para la priorización de casos de prueba).

### 2.2.3 Prueba de regresión automatizada con IA generativa

A medida que se completa cada nueva iteración o lanzamiento, el número de casos de prueba de regresión que se deben ejecutar suele aumentar, lo que los convierte en candidatos ideales para la automatización, especialmente en los procesos de integración continua/entrega continua (IC/EC), debido a la alta frecuencia de ejecución de las pruebas. IAGen puede agilizar este proceso ayudando en la creación, el mantenimiento y la optimización de conjuntos de pruebas de regresión automatizadas. Al adaptarse dinámicamente a los cambios en el código base y realizar análisis de impacto, IAGen puede identificar qué áreas del software son más propensas a verse afectadas por las modificaciones recientes, centrando los esfuerzos de las pruebas de regresión donde más se necesitan.

A continuación se exponen algunas actividades frecuentes en la prueba de regresión automatizada y la elaboración de informes de prueba que pueden realizarse con la ayuda de IAGen:

- **Implementación de guiones de prueba automatizados con automatización basada en palabras clave:** los MLG se pueden utilizar para implementar guiones de prueba basados en marcos de automatización de pruebas basados en palabras clave, en los que las palabras clave predefinidas representan pasos de prueba comunes. IAGen puede asignar estas palabras clave a casos de prueba específicos, generar guiones de prueba y ayudar a los probadores y a los ingenieros de automatización de pruebas en su trabajo.
- **Análisis de impacto y optimización de la prueba:** IAGen se puede utilizar para analizar los cambios en el código con el fin de identificar áreas de alto riesgo, lo que permite realizar pruebas de regresión específicas donde más se necesitan.
- **Pruebas autorreparables y pruebas adaptativas:** IAGen se puede utilizar para ajustar automáticamente los guiones de prueba con el fin de gestionar cambios menores en la interfaz de usuario o la interfaz de programación de aplicación (IPA), lo que evita fallos innecesarios debidos a pequeñas modificaciones y garantiza que los conjuntos de pruebas se mantengan estables a lo largo del tiempo.
- **Suministro de información de prueba automatizado y perspectiva:** IAGen permite la generación de informes de prueba detallados y disponibles de forma oportuna con métricas de



éxito, fallos y perspectivas clave, proporcionando a los implicados paneles de control que resaltan las tendencias de la prueba y ofrecen perspectivas predictivas sobre posibles puntos de fallo.

- **Mejora del suministro de información de defectos y análisis de las causas raíz:** IAGen puede apoyar la recopilación automática de informes completos de defectos con registros de prueba, capturas de pantalla y datos del entorno de prueba.

Estas actividades pueden aplicarse a una variedad de pruebas de regresión, incluidas las pruebas de regresión funcional y no funcional. Sin embargo, los probadores deben ser conscientes de que IAGen puede cometer errores. Por lo tanto, el resultado generado debe verificarse cuidadosamente, dependiendo del riesgo asociado (véase el capítulo 3).

Además, IAGen puede asistir en la prueba de regresión extremo a extremo automatizada basada en interfaz gráfica de usuario (IGU) e interfaz de programación de aplicación (IPA), cada una con sus propios retos y soluciones. Las pruebas de IGU suelen volverse inestables debido a cambios recurrentes en la interfaz de usuario. IAGen puede adaptar automáticamente los guiones de prueba para manejar cambios como localizadores dinámicos e interacciones modificadas, reduciendo la necesidad de intervención manual. La prueba de regresión de interfaz de programación de aplicación (IPA) se enfrenta a retos como los cambios en los formatos de solicitud/respuesta, los puntos finales y la autenticación. IAGen puede adaptar automáticamente los guiones de prueba a las especificaciones cambiantes de interfaz de programación de aplicación (IPA) y generar diversos datos de prueba, manteniendo una cobertura completa y reduciendo la necesidad de actualizaciones manuales.

#### **HO-2.2.3a - (H2) - Practicar la formulación de instrucciones con pocos ejemplos para crear y gestionar guiones de prueba basados en palabras clave.**

Este ejercicio se centra en desarrollar y automatizar guiones de prueba para una aplicación web determinada utilizando un marco de automatización de prueba de IGU. El ejercicio se estructura en dos secciones principales: automatización de pruebas y depuración de guiones de prueba. La primera parte del ejercicio ofrece orientación sobre cómo crear documentación para una librería de palabras clave, generar guiones de prueba iniciales, hacer que la IA valide estos guiones de prueba y ampliar la cobertura con guiones de prueba adicionales. La segunda parte se centra en el soporte para la depuración, utilizando instrucciones del sistema para crear un asistente de IA que pueda comprobar y corregir los guiones de prueba.

Este ejercicio combina la automatización de pruebas tradicional con la validación asistida por IA, demostrando cómo las instrucciones de pocos ciclos pueden utilizarse de forma eficaz para crear, mantener y depurar guiones de prueba basados en palabras clave.

#### **HO-2.2.3b - (H2) - Practicar la ingeniería de instrucciones estructuradas para el análisis de informes de prueba.**

Este ejercicio ilustra un enfoque metódico para analizar informes de prueba de regresión, utilizando instrucciones estructuradas. El proceso comienza con un análisis de los resultados de la prueba proporcionados y una comparación con la especificación de la prueba. A continuación, se pasa a la agrupación de defectos similares, el mantenimiento de una lista de anomalías conocidas y la verificación cruzada de los resultados. Cada paso está vinculado al siguiente en una única conversación MLG.

El enfoque paso a paso demuestra cómo se pueden utilizar las instrucciones estructuradas para transformar los resultados de las pruebas de regresión y los registros de pruebas en información útil, lo que permite un análisis eficaz de los informes de prueba en el contexto de la prueba de regresión.

## 2.2.4 Monitorización de la prueba y control de la prueba con IA generativa

Las tareas de monitorización de la prueba requieren la recuperación de grandes cantidades de datos (a veces no estructurados), que a menudo ya están disponibles en herramientas de gestión de pruebas que IAGen puede ayudar a analizar y sintetizar.

IAGen facilita una serie de tareas de monitorización de la prueba y control de prueba, entre las que se incluyen:

- **Monitorización de la prueba y análisis de métricas:** IAGen puede facilitar la automatización de la monitorización de la prueba, así como el análisis de tendencias para predecir posibles riesgos y alertar a los equipos de cualquier desviación del plan. Esto permite a los equipos mantenerse informados y tomar medidas para mantener los estándares de calidad.
- **Control de la prueba:** IAGen puede asistir en el control de la prueba proporcionando información para reordenar las prioridades de las pruebas, ajustar los calendarios de prueba y reasignar los recursos según sea necesario. Esto garantiza que la prueba siga siendo flexible y se centren en las áreas de alta prioridad.
- **Perspectivas sobre la compleción de la prueba y aprendizaje continuo:** IAGen puede asistir en la generación de informes sobre la compleción de la prueba, destacando los éxitos y las lecciones aprendidas. Esto permite a los equipos perfeccionar las estrategias de prueba y mejorar los procesos de prueba futuros.
- **Mejora de la visualización de métricas y el suministro de información de la prueba:** IAGen puede asistir en la creación de cuadros de mando dinámicos y resúmenes en lenguaje natural, asegurando que todas las partes interesadas tengan acceso a las métricas relevantes. Esta asistencia proporciona la información necesaria para tomar decisiones rápidas y ofrece una visión clara del avance de la prueba.

### HO-2.2.4 - (H0) - Observar las métricas de supervisión de la prueba preparadas por la IA generativa a partir de los datos de prueba.

Esta demostración ilustra cómo IAGen puede asistir a los equipos de prueba transformando los datos de prueba en métricas de monitorización de la prueba procesables, lo que facilita la toma de decisiones informadas. A partir de los datos de prueba extraídos de las herramientas de prueba, un MLG los procesa para generar métricas clave como el progreso de la prueba, las tendencias de defectos o la cobertura, destacando los riesgos potenciales. Estas métricas generadas por IA pueden mostrarse en un cuadro de mando y resumirse en lenguaje natural para que todos los implicados las comprendan fácilmente.

Esta demostración ilustra cómo IAGen convierte los datos de prueba en información práctica, ayudando a los equipos de prueba a supervisar el avance de la prueba, gestionar la calidad y adaptarse rápidamente a los cambios.

## 2.2.5 Selección de técnicas de formulación de instrucciones para la prueba de software

La siguiente tabla presenta la adecuación de las tres técnicas de formulación de instrucciones mencionadas en la sección 2.1.2 según las características de la tarea de prueba.

Técnica de formulación de instrucciones	Casos de uso recomendados	Características y aplicaciones clave
Encadenamiento de instrucciones	Tareas complejas que requieren precisión con verificación humana en cada paso	Descompone las tareas en pasos más pequeños, lo que resulta útil para el análisis de prueba, diseño de prueba y automatización de prueba, donde se comprueba la precisión de cada paso de la prueba.
Formulación de instrucciones con pocos ejemplos	Tareas repetitivas o con formatos de salida específicos/restringidos	Aporta ejemplos a IAGen para la generación repetitiva con un patrón específico, por ejemplo, en casos de prueba con un formato Gherkin (por ejemplo, basados en escenarios), prueba guiada por palabras clave o informes de prueba con un formato de salida específico.
Metainstrucción	Tareas flexibles y dinámicas, útiles para crear instrucciones para nuevas tareas	Descripción general del objetivo y la tarea a realizar, que guía al MLG en la creación de la instrucción. Útil para todo tipo de tareas complejas, como el análisis de informes de prueba y la detección de anomalías.

Incluso es posible utilizar múltiples técnicas para un solo caso de uso. Por ejemplo, se puede utilizar la metainstrucción para crear una instrucción inicial. Esta instrucción generada puede contener ejemplos que deben adaptarse y que pueden mejorarse (formulación de instrucciones con pocos ejemplos). Por último, puede ser útil dividir la tarea en subtareas más pequeñas para permitir la validación de los pasos intermedios (encadenamiento de instrucciones).

### HO-2.2.5 - (H1) - Seleccionar y aplicar técnicas de instrucción adecuadas al contexto para una tarea de prueba determinada

Este ejercicio se centra en seleccionar la técnica de formulación de instrucciones adecuada para diferentes tareas de prueba. Se proporcionan a los participantes varias tareas de prueba con diferentes retos. Para cada tarea de prueba, los participantes deben evaluar la naturaleza de la tarea (si requiere precisión o una estructura repetitiva) y sugerir la técnica de formulación de instrucciones que mejor se adapte al contexto y satisfaga las necesidades específicas de la tarea. Las opciones se debaten en grupo.

Este ejercicio está diseñado para profundizar en la comprensión de cómo se pueden utilizar eficazmente las diferentes técnicas de formulación de instrucciones en las pruebas prácticas.

## 2.3 Evaluación de los resultados de la IA generativa y mejora de las instrucciones para las tareas de prueba de software

Para evaluar el rendimiento de IAGen en la prueba de software se requiere un conjunto claro de métricas que permitan valorar la calidad, la relevancia y la eficacia de los resultados generados (Li

2024). Estas métricas, ya sean generales o específicas para cada tarea, ayudan a optimizar la formulación de instrucciones que se dan a un MLG.

### 2.3.1 Métricas para evaluar los resultados de la IA generativa en tareas de prueba

Se pueden utilizar varias métricas para evaluar la calidad y la eficiencia de los resultados de la IA generativa en una tarea de prueba:

Métrica	Descripción	Ejemplo
Exactitud	Mide la corrección general de la salida generada en comparación con casos de prueba, requisitos u otras normas redactadas por expertos.	El grado en que los casos de prueba generados cubren todos los requisitos especificados.
Precisión	Evalúa la corrección de la salida generada con respecto a un objetivo específico.	El grado en que los casos de prueba generados identifican correctamente las anomalías.
Recuperación	Mide la capacidad de un modelo para identificar todas las instancias relevantes dentro de un conjunto de datos.	El grado en que los casos de prueba generados cubren la partición de equivalencia válida e inválida de una clase de datos.
Relevancia y adecuación contextual	Determina si el resultado generado es aplicable y adecuado para un contexto determinado.	El grado en que los casos de prueba generados son coherentes con la base de prueba e integran los requisitos específicos del dominio.
Diversidad	Asegura que se cubra una amplia gama de entradas y escenarios, evitando repeticiones.	El grado en que los casos de prueba generados cubren diversos comportamientos de los usuarios y en que exploran casos extremos.
Índice de éxito de ejecución	Mide la proporción de casos de prueba o guiones de prueba generados que se pueden ejecutar con éxito.	La determinación de cuántos de los guiones de prueba generados pueden ejecutarse sin errores de sintaxis ni problemas de formato de salida en un entorno de prueba que, por lo demás, funciona correctamente.
Eficiencia temporal	Evalúa el tiempo ahorrado en comparación con los esfuerzos de prueba manuales.	El tiempo que necesita la IA para generar casos de prueba frente al tiempo que tardaría un ser humano en crear manualmente pruebas equivalentes.

Además de estas métricas generales, se pueden adaptar métricas específicas para cada tarea con el fin de evaluar en qué medida IAGen soporta actividades de prueba concretas.

Para evaluar estas métricas de forma eficaz, los probadores pueden realizar revisiones manuales o automatizarlas, por ejemplo, comparando los resultados del MLG con una referencia predefinida. Dada la naturaleza no determinista de IAGen, las métricas deben basarse en datos estadísticamente relevantes.



#### HO-2.3.1 - (H0) - Observar cómo se pueden utilizar las métricas para evaluar el resultado de la IA generativa en una tarea de prueba.

Durante una demostración sobre una tarea de prueba determinada, se muestran métricas adaptadas a la tarea para evaluar los resultados de IAGen, así como su aplicación concreta a los resultados obtenidos con un MLG en esa tarea de prueba.

Esta demostración ilustra la importancia de las métricas de evaluación para proporcionar confianza en los resultados de la IA generativa para la prueba de software.

### 2.3.2 Técnicas para evaluar y perfeccionar instrucciones de forma iterativa

A partir de las métricas presentadas anteriormente, se utilizan técnicas específicas para la evaluación y el perfeccionamiento de instrucciones con el fin de mejorar los resultados de la IA:

- **Modificación iterativa de instrucciones:** comenzar con una instrucción básica y modificarla de forma iterativa en función de los resultados observados, añadiendo gradualmente más contexto o ajustando la redacción (por ejemplo, en lo que respecta a la terminología) para mejorar la especificidad y la relevancia.
- **Pruebas A/B de instrucciones:** crear varias versiones de instrucciones y evaluar cuál de ellas produce mejores resultados según métricas predefinidas. Este enfoque ayuda a determinar qué redacción o estructura de instrucciones produce los resultados más precisos y relevantes.
- **Análisis de resultados:** Examinar los resultados generados por la IA en busca de imprecisiones o incoherencias, por ejemplo, con respecto a la base de la prueba. Comprender los tipos de errores e incoherencias puede ayudar a perfeccionar las instrucciones para evitar defectos similares en futuras iteraciones.
- **Integrar retroalimentación de los usuarios:** Recopilar opiniones de los probadores sobre la utilidad y claridad de los resultados generados, por ejemplo, en lo que respecta al nivel de detalle de las pruebas generadas. Analizar sus opiniones y utilizarlas para perfeccionar las instrucciones con el fin de satisfacer mejor las necesidades de las pruebas que se realizan en el mundo real.
- **Ajustar la longitud y la especificidad de las instrucciones:** Experimentar con diferentes longitudes y niveles de detalle en las instrucciones. A veces, añadir más contexto puede mejorar la calidad de la respuesta. En otros casos, las instrucciones más breves pueden dar lugar a una mejor generalización.

Mediante el uso de estas técnicas, los equipos de prueba pueden organizar sesiones de evaluación y optimización de las instrucciones para garantizar la mejora continua de las instrucciones de IAGen. Compartir prácticas entre el equipo de prueba o la organización de prueba no solo ayuda a estandarizar las técnicas de instrucciones y a mantener una calidad constante, sino que también promueve una cultura de aprendizaje y mejora iterativa. Este enfoque colaborativo contribuye a la evolución de las metodologías de prueba de IAGen al permitir a los equipos de prueba aprovechar los conocimientos colectivos, evitar la repetición de errores y perfeccionar el uso de las herramientas de IAGen de forma más eficaz a lo largo del tiempo, por ejemplo, compartiendo librerías de instrucciones.

#### HO-2.3.2 - (H1) - Evaluar y optimizar una instrucción para una tarea de prueba determinada.

Este ejercicio se centra en la aplicación de técnicas de optimización de instrucciones a una tarea de prueba determinada. Los participantes comenzarán con una instrucción inicial y la perfeccionarán de forma iterativa para mejorar los resultados generados por la IA. Utilizarán técnicas como las pruebas A/B y la verificación humana para evaluar y mejorar la calidad de las instrucciones. El objetivo es que los participantes experimenten cómo el perfeccionamiento iterativo conduce a una generación de casos de prueba más eficaz y relevante desde el punto de vista contextual.

Al final del ejercicio, los participantes habrán realizado varias iteraciones de refinamiento de instrucciones y habrán evaluado cada iteración utilizando las métricas discutidas para mejorar la calidad de los resultados de la IA.



### 3 Gestión de riesgos de la IA generativa en la prueba de software

Duración: 160 minutos

#### Palabras Clave<sup>3</sup>

En la siguiente tabla se presentan las palabras clave del capítulo. En este documento, se identifican dos tipos de palabras clave:

- **ISTQB:** identificarán palabras clave del proceso de prueba
- **ESPDOM:** identificarán palabras clave específicas de dominio: **Inteligencia Artificial**

Tipo Palabra Clave	Español	Inglés
ESPDOM	sesgo	bias
ISTQB	privacidad de datos	data privacy
ESPDOM	alucinación	hallucination
ESPDOM	error de razonamiento	reasoning error
ISTQB	seguridad	security
ESPDOM	temperatura	temperature
ISTQB	vulnerabilidad	vulnerability

<sup>3</sup> Las palabras clave se encuentran ordenadas por orden alfabético de los términos en inglés.

## Objetivos de aprendizaje y objetivos prácticos para el “Capítulo 3”

### 3.1 Alucinaciones, errores de razonamiento y sesgos

- |                    |             |   |
|--------------------|-------------|---|
| <b>IAGEN-3.1.1</b> | <b>(K1)</b> | Recordar las definiciones de alucinaciones, errores de razonamiento y sesgos en los sistemas de IA generativa.                                |
| <b>IAGEN-3.1.2</b> | <b>(K3)</b> | Identificar alucinaciones, errores de razonamiento y sesgos en los resultados de MLG.   |
| <b>HO-3.1.2a</b>   | <b>(H1)</b> | Experimentar con alucinaciones en la realización de pruebas con IAGen.  |
| <b>HO-3.1.2b</b>   | <b>(H1)</b> | Experimentar con errores de razonamiento en la realización de pruebas con IAGen.  |
| <b>IAGEN-3.1.3</b> | <b>(K2)</b> | Resumir las técnicas de mitigación para las alucinaciones, los errores de razonamiento y los sesgos de IAGen en tareas de prueba de software. |
| <b>IAGEN-3.1.4</b> | <b>(K1)</b> | Recordar las técnicas de mitigación para el comportamiento no determinista de los MLG.  |

### 3.2 Privacidad de datos y riesgos de seguridad de la IA generativa en la prueba de software

- |                    |             |   |
|--------------------|-------------|---|
| <b>IAGEN-3.2.1</b> | <b>(K2)</b> | Explicar los principales riesgos para la privacidad de datos y el riesgo de seguridad asociados al uso de la IA generativa en la prueba de software.  |
| <b>IAGEN-3.2.2</b> | <b>(K2)</b> | Aportar ejemplos de privacidad de datos y vulnerabilidades en el uso de la IA generativa en la prueba de software.                                    |
| <b>IAGEN-3.2.3</b> | <b>(K2)</b> | Resumir las estrategias de mitigación para proteger la privacidad de los datos y mejorar la seguridad en la IA generativa para la prueba de software. |
| <b>HO-3.2.3</b>    | <b>(H0)</b> | Reconocer los riesgos para la privacidad de los datos y la seguridad en una IA generativa determinada para el caso de estudio de prueba.              |

### 3.3 Consumo energético e impacto medioambiental de la IA generativa para la prueba de software

- |                    |             |   |
|--------------------|-------------|---|
| <b>IAGEN-3.3.1</b> | <b>(K2)</b> | Explicar el impacto de las características de las tareas y el uso de modelos en el consumo energético de la IA generativa en la prueba de software. |
| <b>HO-3.3.1</b>    | <b>(H1)</b> | Utilizar un simulador para calcular la energía y las emisiones de CO <sub>2</sub> para determinadas tareas de prueba con IA generativa.             |

### 3.4 Normativa, estándares y marcos de trabajo para buenas prácticas en materia de IA

- |                    |             |  |
|--------------------|-------------|--|
| <b>IAGEN-3.4.1</b> | <b>(K3)</b> | Recordar ejemplos de normativas, estándares y marcos de buenas prácticas relacionados con la IA generativa en la prueba de software. |
|--------------------|-------------|--|

## 3.1 Alucinaciones, errores de razonamiento y sesgos

Los sistemas IAGen, especialmente los MLG, son propensos a ciertos defectos, entre los que se incluyen alucinaciones, errores de razonamiento y sesgos. Estos defectos reducen la calidad de los resultados de IAGen en las tareas de prueba, lo que da lugar a productos de prueba que no cumplen con las expectativas de los probadores. Los probadores deben identificar estas alucinaciones, errores de razonamiento y sesgos en los resultados de MLG, y se deben tomar medidas para mitigar estos riesgos.

El comportamiento no determinista de los MLG (véase la sección 1.1.2) dificulta la corrección de este tipo de defectos, que pueden parecer corregidos en una salida del MLG, pero reaparecer en otra conversación con el mismo MLG.

### 3.1.1 Alucinaciones, errores de razonamiento y sesgos en la IA generativa.

Las alucinaciones se producen cuando un MLG genera resultados que parecen incorrectos desde el punto de vista factual o irrelevantes para una tarea determinada. En la prueba de software, las alucinaciones pueden manifestarse cuando los MLG crean casos de prueba ficticios o irrelevantes, generan guiones de prueba incorrectos o que no funcionan, o sugieren casos de prueba que verifican criterios de aceptación inexistentes. Esto puede confundir a los probadores y comprometer la validez de los resultados de la prueba.

Los errores de razonamiento se producen cuando los MLG interpretan erróneamente estructuras lógicas, como las relaciones de causa y efecto, la lógica condicional o los procesos de resolución de problemas paso a paso, lo que conduce a conclusiones incorrectas. A diferencia de los seres humanos, los MLG carecen de un verdadero razonamiento lógico y se basan en la comparación de patrones, lo que puede dar lugar a una lógica defectuosa al realizar tareas como el razonamiento matemático (Mirzadeh 2024). La planificación de la prueba y la priorización de los casos de prueba son ejemplos de tareas de prueba que requieren razonamiento lógico y en las que los MLG pueden cometer errores de razonamiento.

Los sesgos de los MLG (Gallegos 2024) provienen de los datos con los que se entrenó el modelo. Estos sesgos pueden dar lugar a resultados que favorecen ciertos tipos de información, enfoques o suposiciones. Por ejemplo, los MLG entrenados principalmente con datos en inglés pueden subrepresentar las perspectivas no anglosajonas. En la prueba de software, los sesgos pueden influir en las respuestas de los MLG cuando, por ejemplo, se generan datos de prueba o se refinan los criterios de aceptación para los casos de prueba.

Las alucinaciones, los errores de razonamiento y los sesgos en los resultados de IAGen son consecuencia de la naturaleza de sus datos de entrenamiento y de las limitaciones inherentes al modelo transformador (véase el capítulo 1). Reconocer y abordar estos retos aumenta la calidad de los resultados de la IA generativa en los procesos de prueba.

### 3.1.2 Identificación de alucinaciones, errores de razonamiento y sesgos en los resultados de MLG

La integración eficaz de los sistemas IAGen en la prueba de software requiere la capacidad de detectar alucinaciones, errores de razonamiento y sesgos en los resultados de MLG. Dependiendo del tipo de problema, se pueden aplicar diferentes enfoques de detección. Los siguientes son enfoques comunes que se aplican mediante revisión o una combinación de revisión y verificación automatizada:

Detección de alucinaciones:

- se comparan los resultados generados por la IA con la documentación existente, los requisitos y el comportamiento conocido del sistema. Las herramientas automatizadas pueden ayudar a cotejar los resultados con fuentes de datos establecidas para señalar las discrepancias.

- Consulta a expertos en el dominio: se recurre a expertos en la materia para validar la exactitud del contenido generado. Su experiencia es esencial para captar matices que los sistemas automatizados podrían pasar por alto.
- Comprobaciones de consistencia: se verifica que los resultados generados sean coherentes entre sí y con la información conocida. Los sistemas automatizados pueden ayudar a identificar patrones y señalar inconsistencias.

Detección de errores de razonamiento:

- Validación lógica: se evalúa el flujo lógico (por ejemplo, la consistencia, la coherencia y el razonamiento estructurado dentro del texto generado) del contenido generado por IA para verificar su coherencia y corrección a través de ciclos de revisión. Las herramientas automatizadas pueden ayudar, pero los casos complejos pueden requerir el juicio humano.
- Prueba de la salida: por ejemplo, ejecutar los casos de prueba o guiones de prueba generados contra los objetos de prueba para verificar los resultados de la prueba. Esto puede automatizarse parcial o totalmente, dependiendo del tipo de producto de prueba que se genere.

Detección del sesgo:

- Se revisa cómo el producto de prueba generado, como los datos de prueba sintéticos, se representa de manera justa y precisa en relación con la estrategia de prueba.
- Se evalúan los sesgos relacionados con los tipos de prueba, como por ejemplo, las pruebas no funcionales infrarrepresentadas en el resultado generado por el MLG.

La implementación concreta de estos métodos de detección dependerá del nivel de riesgo estimado de alucinaciones, errores de razonamiento o sesgos en la tarea de prueba que se realice con IAGen.

#### **HO-3.1.2a - (H1) - Experimentar con alucinaciones en la realización de pruebas con IAGen.**

Este ejercicio se centra en experimentar con ejemplos de alucinaciones de IA generativa en relación con el conjunto de conocimientos sobre la prueba de software. Los participantes intentarán confrontar al menos dos MLG con una situación en la que los MLG inventen elementos irrelevantes, por ejemplo, añadiendo criterios no relacionados que no existen en los datos contextuales proporcionados. Se prueban variaciones en las instrucciones para examinar la influencia de estas en las alucinaciones.

Este ejercicio aumenta la comprensión de la identificación de alucinaciones de IAGen en las pruebas de software.

#### **HO-3.1.2b - (H1) - Experimentar con errores de razonamiento en la realización de pruebas con IAGen.**

Este ejercicio se centra en presentar un ejemplo de error de razonamiento de la IAGen. Un ejemplo de un problema que hay que resolver en el ámbito de la planificación de pruebas, como la estimación del esfuerzo de prueba y la priorización de los casos de prueba (véase [ISTQB\_CTFL] - Capítulo 5). El ejercicio está diseñado con una cierta complejidad de los datos de entrada, lo que requiere habilidades para la resolución de problemas y pone de relieve las limitaciones de los MLG para este fin. El resultado del MLG se comparará con el resultado exacto que se debería obtener. Se probarán tres tipos diferentes de MLG (MLG, MLP y modelo de razonamiento) y se utilizarán variaciones de la instrucción para intentar mejorar los resultados.

Este ejercicio aumenta la comprensión de cómo identificar los errores de razonamiento de IAGen en tareas de prueba de software que requieren habilidades de resolución de problemas lógicos.

### **3.1.3 Técnicas de mitigación de alucinaciones, errores de razonamiento y sesgos de IAGen en tareas de prueba de software**

Para minimizar los resultados indeseables de la IAGen en la prueba de software, se pueden emplear varias estrategias para reducir las alucinaciones, los errores de razonamiento y los sesgos. Estos problemas son más probables cuando las instrucciones no están diseñadas adecuadamente (véase el capítulo 2) o cuando faltan datos contextuales relevantes para una tarea de prueba determinada. Las técnicas clave para mitigar los riesgos asociados con las alucinaciones, los errores de razonamiento y los sesgos de la IA incluyen:



- Aportar un contexto completo: Es necesario asegurar que la instrucción contenga toda la información relevante (véase la sección 2.1.1), ofreciendo un contexto completo que guíe a la IA en la generación de resultados precisos.
- Dividir las instrucciones en segmentos manejables: dividir las instrucciones complejas en pasos más pequeños utilizando técnicas de encadenamiento de instrucciones (véase la sección 2.1.2), verificando sistemáticamente cada resultado antes de pasar al siguiente. Este enfoque paso a paso puede ayudar a detectar errores de razonamiento en una fase temprana del proceso de generación.
- Utilizar formatos de datos claros e interpretables: Evitar formatos que puedan resultar ambiguos o difíciles de interpretar para la IAGen. Los formatos estructurados y sencillos ayudan al modelo a centrarse en los aspectos esenciales de la tarea.
- Seleccionar el modelo IAGen adecuado para la tarea: Utilizar un MLG específicamente entrenado para la tarea en cuestión (véase la sección 5.1.3).
- Comparar resultados entre modelos: Cuando resulte adecuado, evaluar la instrucción con varios MLG y comparar los resultados ayuda a detectar errores y seleccionar los resultados más fiables.

El capítulo 4 presenta dos técnicas complementarias para mejorar los resultados del MLG: la generación aumentada por recuperación y el ajuste fino.

### 3.1.4 Mitigación del comportamiento no determinista de los MLG

El comportamiento no determinista inherente a los MLG (Shuyin 2023) puede dar lugar a variaciones en los resultados, incluso cuando se proporciona la misma entrada. Esto se debe a los procesos de muestreo probabilístico utilizados durante la inferencia. En consecuencia, lograr resultados consistentes y reproducibles al utilizar MLG puede resultar difícil, especialmente en el caso de resultados largos, lo que aumenta el riesgo de variabilidad.

Aunque no se puede garantizar una reproducibilidad completa, ciertas estrategias pueden ayudar a reducir la variabilidad:

- Ajustar la configuración del parámetro de temperatura de MLG: reducir la temperatura durante la generación de respuestas (inferencia) estrecha la distribución de probabilidad, lo que reduce la aleatoriedad y da lugar a resultados más consistentes. Sin embargo, esto también limitará la creatividad y la diversidad de las respuestas, haciendo que los resultados sean más repetitivos o excesivamente deterministas.
- Configuración de semillas aleatorias: algunas implementaciones de MLG permiten configurar un valor de semilla para el generador de números aleatorios, lo que garantiza que se utilice la misma secuencia pseudoaleatoria (es decir, valores aleatorios deterministas), lo que mejora la reproducibilidad.

Para reducir el riesgo de alucinaciones y errores de razonamiento en los resultados de MLG, es necesario abordar este comportamiento no determinista, por ejemplo, automatizando algunos aspectos de la verificación de resultados para garantizar un proceso de evaluación estructurado y consistente.

## 3.2 Privacidad de datos y riesgos de seguridad de la IA generativa en la prueba de software

El uso de IAGen en las pruebas introduce riesgos relacionados con la privacidad de los datos y la seguridad debido al manejo de información confidencial y a las posibles vulnerabilidades de la infraestructura de pruebas dotada de MLG. Es esencial contar con una protección de datos sólida para evitar violaciones, accesos no autorizados y la exposición de datos confidenciales.



### 3.2.1 Riesgos para la privacidad de datos y la seguridad asociados al uso de la IA generativa

La IAGen puede procesar grandes cantidades de datos que pueden contener información confidencial o de identificación personal. Esto plantea las siguientes preocupaciones en materia de privacidad de datos:

- Exposición involuntaria de datos: los modelos de IAGen pueden generar resultados que revelen accidentalmente información confidencial.
- Falta de control sobre el uso de los datos: las herramientas de IAGen pueden almacenar y procesar datos confidenciales sin el consentimiento explícito del usuario ni su control. Esto puede dar lugar a un posible uso indebido o acceso no autorizado.
- Riesgos de cumplimiento: el uso de herramientas de IAGen sin cumplir con las normativas de protección de datos, como el Reglamento General de Protección de Datos (RGPD, Reglamento (UE) 2016/679), podría dar lugar a disputas legales.

Además, al realizar pruebas con IAGen surgen riesgos de seguridad específicos, como:

- La infraestructura de pruebas dotada de MLG puede ser vulnerable a ataques de seguridad, como violaciones de datos o accesos no autorizados.
- Los actores maliciosos pueden explotar las vulnerabilidades de los MLG, como los ataques de manipulación (véase la sección 3.2.2), para alterar su comportamiento o extraer información confidencial.
- Los atacantes pueden introducir intencionadamente datos maliciosos para engañar a los MLG y comprometer su precisión o seguridad.

### 3.2.2 Privacidad de datos y vulnerabilidades en la IA generativa para procesos y herramientas de prueba

La siguiente tabla ofrece algunos ejemplos de vectores de ataque en los procesos y herramientas de prueba de IAGen.

Vector de ataque	Descripción	Ejemplo
Exfiltración de datos (también fuga de datos)	Enviar solicitudes diseñadas para extraer datos de entrenamiento confidenciales.	Superar la ventana contextual del MLG con instrucciones extensas para sobrecargar la memoria de la IA podría llevarla a revelar fragmentos aleatorios de sus datos de entrenamiento y exponer potencialmente información confidencial.
Manipulación de solicitudes	Introducir datos que alteren los resultados de la IA.	Imágenes que atraen a la IA a un contexto diferente, provocando así alucinaciones sobre, por ejemplo, los criterios de aceptación.
Envenenamiento de datos (también contaminación de datos)	Manipular los datos de entrenamiento.	Aportar evaluaciones falsas al calificar los resultados de un informe de prueba generado por IA.

Vector de ataque	Descripción	Ejemplo
Generación de código malicioso	Manipular un MLG para generar puertas traseras (por ejemplo, llamadas a comandos externos) durante su uso.	Generación de código para abrir un canal de comunicación con una IP específica y maliciosa.

### 3.2.3 Estrategias de mitigación para proteger la privacidad de los datos y mejorar la seguridad en la prueba con IA generativa

A medida que la IA genérica se generaliza, y con los riesgos inherentes que ello conlleva, surgen normativas y estándares para mitigarlos (véase la sección 3.4.1).

Las normativas de protección de datos, como el RGPD, no restringen explícitamente las aplicaciones de la IA genérica, pero sí proporcionan salvaguardias que pueden limitar lo que se puede hacer, en particular en lo que respecta a la legalidad y las limitaciones en cuanto a los fines de recopilación, tratamiento y almacenamiento de datos.

Para mitigar estos riesgos, las organizaciones deben implementar medidas sólidas de privacidad de datos, entre las que se incluyen:

- Minimización de datos: evitar el procesamiento de datos sensibles a menos que esté permitido legalmente y utilizar solo la cantidad necesaria de datos no sensibles en la prueba de IA para reducir los riesgos de privacidad de datos.
- Anonimización y seudonimización de datos: enmascarar o sustituir la información sensible por datos no identificables.
- Almacenamiento y transmisión seguros de datos: implementar controles sólidos de cifrado y acceso.
- Formación de los recursos: Las organizaciones deben establecer programas y políticas de formación claros para garantizar el uso responsable de las herramientas de IAGen, promover prácticas éticas y mitigar los riesgos potenciales.

Se pueden considerar estrategias de mitigación adicionales al implementar IAGen para la prueba:

- Revisión sistemática de los resultados generados: la evaluación humana es esencial para garantizar la calidad y la precisión de las tareas de prueba impulsadas por IAGen.
- Evaluación mediante comparación con otro MLG: Esto implica utilizar varios MLG en una tarea determinada para evaluar los resultados comparando sus respuestas.
- Elección de un entorno operativo seguro: Dependiendo del nivel de confidencialidad requerido, las organizaciones pueden optar por diferentes soluciones seguras: utilizar una oferta comercial y segura de un proveedor de MLG, operar el MLG en una nube segura o instalar el MLG en la infraestructura de la organización.
- Auditorías periódicas de seguridad y evaluaciones de vulnerabilidad: Identificar y abordar las debilidades de los sistemas IAGen.
- Mantenerse al día con las mejores prácticas de seguridad: Mantenerse al día con las últimas directrices y tecnologías de seguridad.

Las estrategias suelen ser complementarias entre sí y es necesario combinarlas para garantizar la seguridad de los datos al utilizar IAGen. Se recomienda especialmente contar con la participación de ingenieros de seguridad sénior, asesores jurídicos, el director de tecnología (DT) o el director de seguridad de la información (DSI), si los hay en la organización.

**HO-3.2.3 - (H0) - Reconocer los riesgos para la privacidad de los datos y la seguridad en una IA generativa determinada para el caso de estudio de prueba.**

Esta demostración ilustra cómo pueden surgir riesgos para la privacidad y la seguridad de los datos al utilizar IAGen en las pruebas de software. Los participantes analizarán casos prácticos para identificar posibles amenazas, como vulnerabilidades de los modelos, acceso no autorizado a los datos o uso malintencionado de los resultados generados. Explorarán estrategias de mitigación, como el manejo seguro de los datos, controles de acceso robustos y prácticas de supervisión de la IA, al tiempo que reflexionarán sobre las implicaciones éticas y prácticas.

Al final, los participantes comprenderán los principios de la privacidad de datos y aprenderán a reconocer y abordar los riesgos de seguridad en las condiciones de prueba de IAGen.

### **3.3 Consumo de energía e impacto medioambiental de la IA generativa en la prueba de software**

Estudios como el de Luccioni (2024a) muestran que el entrenamiento y el procesamiento de los MLG requieren un uso intensivo de una gran cantidad de recursos informáticos especializados. Los MLG están disponibles como servicios basados en la web, y su uso aumenta la carga sobre los dispositivos, las redes y los centros de datos, lo que conduce a un mayor consumo de energía.

#### **3.3.1 Impacto del uso de IAGen en el consumo energético y las emisiones de CO<sub>2</sub>**

No se debe subestimar el impacto medioambiental de la IAGen, ya que el consumo de energía aumenta considerablemente a medida que aumenta su uso. La complejidad de la tarea y los recursos de cálculo necesarios influyen en el consumo de energía. Por ejemplo, generar una sola imagen utilizando un potente modelo de IA puede consumir tanta energía como cargar completamente un smartphone, mientras que generar texto solo consume un pequeño porcentaje de la carga de un smartphone (Heikkilä 2023).

Aunque es difícil obtener datos precisos sobre el impacto medioambiental de la IAGen (Luccioni 2024b), está claro que estas operaciones, que consumen mucha energía, contribuyen colectivamente a unas emisiones significativas de CO<sub>2</sub> (Berthelot 2024). Si bien una sola búsqueda o tarea de generación de texto puede parecer insignificante, su efecto acumulativo en millones de usuarios de todo el mundo supone una carga medioambiental considerable.

Adoptar las mejores prácticas, como limitar las interacciones innecesarias con los modelos, es fundamental para mitigar los riesgos medioambientales que plantea la IAGen.

**HO-3.3.1 - (H1) - Utilizar un simulador para calcular la energía y las emisiones de CO<sub>2</sub> para determinadas tareas de prueba con IA generativa**

Este ejercicio se centra en evaluar el consumo de energía y las emisiones de CO<sub>2</sub> asociadas a diversas tareas de IA generativa dentro de la prueba de software. Los participantes utilizarán simulaciones para calcular estas métricas y examinar cómo las diferentes características de las tareas y el uso de los modelos afectan al impacto medioambiental.

Al observar cómo diferentes factores afectan al consumo de energía y a las emisiones, los participantes comprenden los factores que impulsan el consumo de energía con los MLG.

### 3.4 Normativas, estándares y marcos de trabajo de buenas prácticas en materia de IA

La IAGen está transformando la prueba de software al ayudar a los probadores en una variedad de tareas de prueba (véase el capítulo 2). Sin embargo, estas oportunidades también conllevan riesgos importantes, como errores de razonamiento, privacidad de datos, vulnerabilidades e impactos ambientales (véanse las secciones 3.1, 3.2 y 3.3). Para abordar estos riesgos, se deben tener en cuenta las normativas generales, las normas y los marcos de buenas prácticas para la IA.

#### 3.4.1 Normativas, estándares y marcos de trabajo relevantes para la IA generativa en la prueba de software

A continuación se ofrece una descripción general de las directrices clave relevantes para el uso de la IA generativa en la prueba de software:

Código	Nombre / Tipo	Descripción	Aplicación en la prueba de software
ISO/IEC 42001:2023	Information technology - Artificial intelligence - Management system	Especifica los requisitos para gestionar los sistemas de IA dentro de una organización.	Asegura que la IAGen en la prueba se adhiera a las prácticas recomendadas, promoviendo la consistencia y la fiabilidad.
	Tecnología de la información. Inteligencia artificial. Sistema de gestión		
	TIPO: NORMA		
ISO/IEC 23053:2022	Framework for Artificial Intelligence (AI) Systems Using Machine Learning	Proporciona un marco para los procesos del ciclo de vida de la IA, haciendo énfasis en la tolerancia a fallos y la transparencia.	Proporciona un marco para la calidad de los datos, la transparencia y la tolerancia a fallos cuando se utiliza la IAGen para la prueba.
	Marco para sistemas de inteligencia artificial (IA) que utilizan aprendizaje automático		
	TIPO: NORMA		
AI Act 2024	EU AI Act	Establece un marco legal que aborda los riesgos de la IA, clasificando las aplicaciones por nivel de riesgo.  Fuente: (AI Act 2024)	Exige el cumplimiento de la transparencia, la responsabilidad y la mitigación del sesgo para la IAGen utilizada en la prueba.
	Ley de IA de la UE		
	TIPO: REGLAMENTO		
NIST AI RMF 1.0	NIST AI Risk Management Framework (US)	Ofrece directrices para la gestión de los riesgos de la IA,	Garantiza la equidad y mitiga los riesgos en la IAGen,

Código	Nombre / Tipo	Descripción	Aplicación en la prueba de software
	<b>Marco de gestión de riesgos de IA del NIST (EE. UU.)</b>	centrándose en la equidad, la transparencia y la seguridad.	evitando resultados sesgados en las pruebas.
	<b>TIPO: MARCO</b>	Fuente: (NIST AI RMF 1.0)	

A medida que las tecnologías de IA y su marco regulatorio continúan evolucionando, es imprescindible que las organizaciones de prueba se mantengan al día en lo que respecta al desarrollo de reglamentos, normas, leyes nacionales y marcos de buenas prácticas, como los que se incluyen en esta tabla.



## 4 Infraestructura de prueba impulsada por MLG para la prueba de software

Duración: 110 minutos

### Palabras Clave<sup>4</sup>

En la siguiente tabla se presentan las palabras clave del capítulo. En este documento, se identifican dos tipos de palabras clave:

- **ISTQB**: identificarán palabras clave del proceso de prueba
- **ESPDOM**: identificarán palabras clave específicas de dominio: **Inteligencia Artificial**

Tipo Palabra Clave	Español	Inglés
ISTQB	infraestructura de prueba	test infrastructure
ESPDOM	ajuste fino	fine-tuning
ESPDOM	agente impulsado por MLG	LLM-powered agent
ESPDOM	operaciones de modelos de lenguaje grande	large language model operations
ESPDOM	generación aumentada por recuperación	retrieval-augmented generation
ESPDOM	base de datos vectorial	vector database

<sup>4</sup> Las palabras clave se encuentran ordenadas por orden alfabético de los términos en inglés.

## Objetivos de aprendizaje y objetivos prácticos para el “Capítulo 4”

### 4.1 Enfoques arquitectónicos para la infraestructura de prueba impulsada por MLG

- |                    |             |   |
|--------------------|-------------|---|
| <b>IAGen-4.1.1</b> | <b>(K2)</b> | Explicar los componentes y conceptos arquitectónicos clave de la infraestructura de prueba impulsada por MLG.         |
| <b>IAGen-4.1.2</b> | <b>(K2)</b> | Resumir la generación aumentada por recuperación.   |
| <b>HO-4.1.2</b>    | <b>(H1)</b> | Experimentar con la generación aumentada por recuperación para una tarea de prueba determinada.                       |
| <b>IAGen-4.1.3</b> | <b>(K2)</b> | Explicar la función y la aplicación de los agentes impulsados por MLG en la automatización de los procesos de prueba. |
| <b>HO-4.1.3</b>    | <b>(H0)</b> | Observar cómo un agente impulsado por MLG ayuda a automatizar una tarea de prueba repetitiva                          |

### 4.2 Ajuste fino y OpsMLG: puesta en marcha de la IA generativa para la prueba de software

- |                    |             |  |
|--------------------|-------------|--|
| <b>IAGen-4.2.1</b> | <b>(K2)</b> | Explicar el ajuste fino de los modelos de lenguaje para tareas de prueba específicas.                        |
| <b>HO-4.2.1</b>    | <b>(H0)</b> | Observar un ejemplo de proceso de ajuste fino para una tarea de prueba y un modelo de lenguaje determinados. |
| <b>IAGen-4.2.2</b> | <b>(K2)</b> | Explicar OpsMLG y su función en la implementación y gestión de MLG para tareas de prueba.                    |

## 4.1 Enfoques arquitectónicos para la infraestructura de prueba impulsada por MLG

Los bots IA de conversación y las herramientas de prueba basadas en MLG son dos tipos de infraestructura de prueba que utilizan MLG (véase la sección 1.2.2).

Más allá de la arquitectura básica de una infraestructura de prueba impulsada por MLG (véase la sección 4.1.1), la generación aumentada por recuperación (véase la sección 4.1.2) y las arquitecturas de agentes impulsados por MLG (véase la sección 4.1.3) amplían la funcionalidad y la utilidad del uso de MLG en la prueba de software.

### 4.1.1 Componentes y conceptos arquitectónicos clave de la infraestructura de prueba impulsada por MLG

Una infraestructura de prueba impulsada por MLG se refiere a un sistema que integra un MLG en el proceso de prueba de software para mejorar la automatización, el razonamiento y la toma de decisiones. A diferencia de un chatbot con IA tradicional, que se centra principalmente en las interacciones en forma de conversación, una herramienta de prueba dotada de MLG está diseñada para apoyar la prueba de software mediante el procesamiento de consultas relacionadas con la prueba, el análisis de requisitos, la generación de casos de prueba y la evaluación de resultados.

La arquitectura típica de una infraestructura de prueba impulsada por MLG sigue un diseño multicomponente que facilita la interacción segura y eficiente con el MLG. La arquitectura consta de los componentes correspondientes a la capa de presentación y la capa de servicios, junto con fuentes de datos externas y un MLG integrado:

- La capa de presentación (en inglés, front-end) sirve como interfaz de usuario, donde los probadores interactúan con el sistema introduciendo consultas o comandos.
- La capa de servicios (en inglés, back-end) procesa la información introducida por el usuario y gestiona funciones críticas como la autenticación, la recuperación de datos, la preparación de instrucciones y la interacción con el MLG.
- El MLG, que puede alojarse como un servicio de terceros (al que se accede a través de interfaz de programación de aplicación (IPA) o como un modelo interno personalizado, genera respuestas basadas en instrucciones estructuradas.

Esta arquitectura va más allá del modelo cliente-servidor tradicional al incorporar componentes de procesamiento inteligente, como MLG y capas de servicio multifuentes:

- 1 Esta arquitectura va más allá del modelo cliente-servidor tradicional al incorporar componentes de procesamiento inteligente, como MLG y capa de servicios multifuentes:
- 2 El MLG no es solo un servidor, sino un componente de procesamiento inteligente que interpreta y razona basándose en productos de prueba.
- 3 La capa de servicios integra múltiples fuentes de datos, tales como:
  - Bases de datos relacionales (para datos estructurados utilizados en pruebas, como casos de prueba).
  - bases de datos vectoriales (para la recuperación semántica de contenido relacionado mediante incrustaciones; véase la sección 4.1.2).
- 4 La capa de servicios mejora la salida sin procesar del MLG mediante el posprocesamiento, lo que garantiza que sus respuestas se ajusten a las condiciones de prueba del proceso de prueba antes de presentarlas a la interfaz de usuario.

#### 4.1.2 Generación aumentada por recuperación

La generación aumentada por recuperación (GAR) mejora los MLG al incorporar fuentes de datos adicionales en su proceso de generación de respuestas (Zhao 2024), lo que aumenta la relevancia y la precisión de sus resultados.

GAR combina sistemas de recuperación con modelos de lenguaje para generar respuestas sensibles al contexto. Durante el preprocesamiento, los documentos grandes se dividen en fragmentos más pequeños (por ejemplo, 256-512 tokens) para garantizar una recuperación centrada y la compatibilidad con la ventana de contexto del modelo. Cada fragmento se limpia, se procesa y se codifica en un vector de alta dimensión (incrustación) utilizando modelos preentrenados. Estas incrustaciones, que pueden almacenarse en bases de datos vectoriales, permiten una recuperación eficiente basada en la similitud en tiempo de ejecución (inferencia). Se codifica la consulta del usuario, se recuperan los fragmentos relevantes en función de la similitud semántica y estos fragmentos se utilizan como contexto para que el modelo lingüístico genere una respuesta fundamentada.

Una respuesta relevante es, esencialmente, un resultado generado por el modelo de lenguaje que se basa en información relevante, precisa y contextualmente adecuada recopilada durante el proceso de recuperación. Esto garantiza que la respuesta no solo se base en el entrenamiento preexistente del modelo, sino que también se enriquezca con datos precisos pertinentes a la instrucción. Esta sinergia entre la recuperación y la generación mejora la precisión y la relevancia de las respuestas, haciéndolas más fiables e informativas para el usuario.

En la fase de procesamiento de la instrucción de usuario, un sistema GAR funciona mediante un proceso de dos pasos:

- 1 Recuperación: Dada una consulta del usuario, el sistema recupera la información relevante de las bases de datos vectoriales creadas previamente. Esta recuperación se basa normalmente en la similitud semántica entre las incrustaciones de la instrucción y las de los fragmentos.
- 2 Generación: La información recuperada se envía al MLG, que genera una respuesta que combina sus conocimientos existentes con los datos recién adquiridos, lo que da como resultado una salida más precisa y adecuada al contexto.

La GAR en la prueba de software permite que la infraestructura de prueba impulsada por MLG acceda a las fuentes de datos corporativas de la empresa, como bases de datos, documentación y repositorios, para recuperar información contextual en tiempo real, lo que garantiza que las tareas de prueba, como el análisis o el diseño de pruebas, se ajusten a las últimas especificaciones, requisitos y datos de prueba existentes.

##### HO-4.1.2 - (H1) - Experimentar con la generación aumentada por recuperación para una tarea de prueba determinada.

Este ejercicio práctico se centra en la aplicación de técnicas GAR para una tarea de prueba determinada. Los participantes experimentarán con un sistema GAR incorporando documentos y observarán cómo genera respuestas más o menos precisas basadas en información compleja. Los participantes compararán los resultados del MLG con y sin GAR en la tarea de prueba determinada. El objetivo de este ejercicio es identificar los puntos fuertes y las limitaciones del sistema GAR en el manejo de diferentes tipos de tareas de prueba.

Al examinar los datos recuperados y los resultados generados, los participantes obtendrán información sobre el papel de GAR en la mejora de los procesos de prueba dotados de MLG.

#### 4.1.3 El papel de los agentes impulsados por MLG en la automatización de los procesos de prueba

Los agentes impulsados por MLG (Wang 2024) son aplicaciones IAGen especializadas que funcionan con MLG y están diseñadas para el procesamiento semiautónomo o autónomo de tareas definidas. En



esencia, estos agentes se basan en MLG para la comprensión y generación del lenguaje natural, complementadas con la posibilidad de procesar instrucciones, recuperar contexto y realizar acciones inteligentes.

Los agentes impulsados por MLG (Wang 2024) son aplicaciones IAGen especializadas que funcionan con MLG y están diseñadas para el procesamiento semiautónomo o autónomo de tareas definidas. En esencia, estos agentes se basan en MLG para la comprensión y generación del lenguaje natural, complementadas con la posibilidad de procesar instrucciones, recuperar contexto y realizar acciones inteligentes. A diferencia de los bots IA de conversación tradicionales, que se centran únicamente en las interacciones de pregunta-respuesta, los agentes impulsados por MLG pueden realizar tareas o “actuar” invocando un conjunto predefinido de funciones, comúnmente denominadas “herramientas”. Esta capacidad les permite interactuar con sistemas externos y manipularlos, lo que los hace muy versátiles en la ejecución de tareas. El grado de autonomía de los agentes impulsados por MLG puede variar:

Los agentes autónomos operan de forma independiente, realizando tareas con una intervención humana mínima mediante reglas predefinidas, aprendizaje por refuerzo y bucles de retroalimentación adaptativos.

Los agentes semiautónomos realizan tareas con supervisión humana periódica para garantizar que el resultado cumpla los objetivos definidos por el usuario.

Las arquitecturas multiagente implican un sistema colaborativo en el que varios agentes, cada uno con funciones especializadas, se comunican y coordinan para resolver problemas complejos de forma más eficiente que un solo agente. Este esfuerzo coordinado entre múltiples agentes de IA se conoce como “orquestración”.

#### **Objetivo práctico 4.1.3 (H0): Observar cómo un agente impulsado por MLG ayuda a automatizar una tarea de prueba repetitiva**

La demostración se centra en una tarea de prueba realizada por un agente impulsado por MLG. Se mostrarán los datos de entrada que se pasan al agente, su comportamiento y los resultados de sus acciones para ilustrar los diversos aspectos de la integración de soluciones basadas en agentes en un proceso de prueba.

Esta demostración muestra un ejemplo concreto de un agente impulsado por MLG en el contexto de una tarea de prueba.

## 4.2 Ajuste fino y OpsMLG: puesta en marcha de la IA generativa para la prueba de software

Dos prácticas clave para poner en funcionamiento una infraestructura de prueba impulsada por MLG para probar son el ajuste fino de los MLG y la gestión de la canalización operativa a través de OpsMLG (Mailach 2024).

### 4.2.1 Ajuste fino de MLG para tareas de prueba

El ajuste fino adapta un modelo de lenguaje (ML) preentrenado, como un MLG o un modelo de lenguaje pequeño (MLP, véase la sección 1.1.2), para realizar tareas específicas o adaptarlo a dominios concretos (Parthasarathy 2024). Esto implica seguir entrenando el modelo con un conjunto de datos específico, lo que le permite adquirir conocimientos y matices propios del dominio. Mediante el ajuste fino, se mejora el rendimiento del modelo para aplicaciones especializadas, lo que lo hace más preciso y relevante para el caso de uso previsto.

En la práctica, el ajuste fino es adecuado para dotar a los MLG genéricos de capacidades de razonamiento especializadas relevantes para un dominio específico o para adoptar un vocabulario exclusivo de ese campo. El ajuste fino también se puede aplicar a modelos más pequeños, conocidos como modelos de lenguaje pequeño (MLP), que consumen menos recursos. Mediante el ajuste fino de un modelo de lenguaje pequeño (MLP), se pueden alcanzar niveles de rendimiento más altos para tareas específicas sin la misma sobrecarga de recursos de proceso que requieren los MLG. Esta comparación pone de relieve la flexibilidad y la eficiencia del uso tanto de MLG como de MLP en función de los requisitos específicos de la tarea.

Por ejemplo, en la prueba de software, el ajuste fino puede permitir que un MLG o MLP genere casos de prueba a partir de historias de usuarios en un formato de salida específico para el contexto de la organización. Al entrenar el modelo con las historias de usuarios de la organización y los casos de prueba correspondientes, el modelo se alinea con el proceso de prueba y la terminología específicos de la organización.

El ajuste fino de un modelo IAGen para la prueba de software presenta varios retos:

- Evitar resultados sesgados o inexactos garantizando el uso de conjuntos de datos de entrenamiento de alta calidad y específicos para cada tarea.
- Mitigar el sobreajuste (el modelo se vuelve demasiado especializado en los datos de entrenamiento, lo que afecta negativamente a su rendimiento con datos nuevos y desconocidos) para mantener la generalización en diferentes escenarios.
- Abordar la opacidad (falta de transparencia en la forma en que un MLG toma sus decisiones o produce sus resultados) en el razonamiento del modelo, lo que complica la depuración y la validación.
- Gestionar los importantes recursos de cálculo necesarios para el proceso de ajuste fino (para los MLG).

#### **HO-4.2.1 - (H0) - Observar un ejemplo de proceso de ajuste fino para una tarea de prueba y un modelo de lenguaje determinados.**

Esta demostración expone los distintos pasos que intervienen en el ajuste fino de un MLG para una tarea de prueba determinada. Comienza con la selección de un MLG o MLP adecuado. A continuación, se presenta un conjunto de datos adaptado a la tarea de prueba determinada. Luego se muestra una solución de ejemplo para el proceso de ajuste fino (por ejemplo, un marco de aprendizaje automático). Por último, se envía una instrucción al modelo ajustado y se analiza la calidad del resultado generado.

Esta demostración del proceso de ajuste fino de MLG/MLP para una tarea de prueba muestra varios aspectos clave de este proceso y aborda, en particular, la calidad de los datos de entrenamiento.

### **4.2.2 Operaciones del modelo de lenguaje grande (OpsMLG) al implementar y gestionar MLG para la prueba de software**

Las operaciones del modelo de lenguaje grande (OpsMLG) se refieren al conjunto de prácticas, herramientas y procesos diseñados para optimizar el desarrollo, la implementación y el mantenimiento de los MLG en entornos de producción (Sinha, 2024).

El uso de la IA generativa en los procesos de prueba de una organización puede llevarse a cabo de varias maneras diferentes, lo que influirá en las decisiones que se tomen en materia de OpsMLG. A continuación se presentan tres posibles enfoques:

- **Uso de un chatbot con IA:** Las principales consideraciones para este enfoque incluyen la gestión de los riesgos de privacidad y seguridad de los datos, al tiempo que se optimizan los costes. Las organizaciones pueden utilizar plataformas MLG como servicio si se ofrecen las garantías necesarias, o implementar una infraestructura interna utilizando MLG con licencia de código abierto para un mayor control. Es fundamental realizar una evaluación rigurosa de las garantías de los proveedores o de las capacidades internas para mitigar los riesgos de privacidad de datos y seguridad (véase la sección 3.2) y garantizar la eficiencia operativa.
- **Uso de una herramienta de prueba con capacidades de IA generativa:** este enfoque tiene consideraciones similares a las de los bots IA de conversación, como la privacidad de los datos, la seguridad y los costes operativos. Además, las organizaciones deben evaluar la seguridad de los datos y las garantías de rendimiento que ofrece el proveedor de la herramienta de prueba. Estas herramientas de prueba suelen complementar los procesos de prueba existentes, lo que requiere un análisis exhaustivo de la relación coste-beneficio y una evaluación de los riesgos.
- **Desarrollo interno de una herramienta de prueba basada en IA generativa:** este enfoque hace énfasis en el control integral de los riesgos de privacidad y seguridad de los datos, así como en la planificación cuidadosa de los costos operativos de la IA, tales como los recursos de procesamiento, el almacenamiento de datos y la capacitación del personal. Las organizaciones también deben establecer procesos estructurados para validar y mantener los desarrollos específicos de IAGen. El desarrollo de soluciones internas requiere experiencia en la implementación y el despliegue de una infraestructura de pruebas impulsada por MLG.

Estos enfoques no son mutuamente excluyentes, ya que una organización puede utilizar un chatbot con IA para algunas tareas y desarrollar herramientas personalizadas para otras. Por lo tanto, pueden implementarse simultáneamente en función de las actividades de prueba específicas que se realicen. Además, pueden incorporar tecnologías adicionales, como GAR y el ajuste fino de MLG/MLP, para mejorar la eficacia y la adaptabilidad de los procesos de prueba con IAGen.

## 5 Despliegue e integración de IA generativa en organizaciones de prueba

Duración: 80 minutos

### Palabras Clave<sup>5</sup>

En la siguiente tabla se presentan las palabras clave del capítulo. En este documento, se identifican dos tipos de palabras clave:

- **ISTQB:** identificarán palabras clave del proceso de prueba
- **ESPDOM:** identificarán palabras clave específicas de dominio: **Inteligencia Artificial**

Tipo Palabra Clave	Español	Inglés
ESPDOM	IA en la sombra	shadow AI

<sup>5</sup> Las palabras clave se encuentran ordenadas por orden alfabético de los términos en inglés.



## Objetivos de aprendizaje y objetivos prácticos para el “Capítulo 5”

### 5.1 Guía para la adopción de IA generativa en la prueba de software

- IAGEN-5.1.1 (K1)** Recordar los riesgos de la IA en la sombra.
- IAGEN-5.1.2 (K2)** Explicar los aspectos clave a tener en cuenta al definir una estrategia de IA generativa para la prueba de software.
- IAGEN-5.1.3 (K2)** Resumir los criterios clave para seleccionar MLG/MLP para tareas de prueba de software en un contexto determinado.
- HO-5.1.3 (H1)** Estimar los costes recurrentes del uso de la IA generativa para una tarea de prueba determinada.
- IAGEN-5.1.4 (K1)** Recordar las fases clave en la adopción de la IA generativa en una organización de prueba.

### 5.2 Gestión del cambio al adoptar la IA generativa en la prueba de software

- IAGen-5.2.1 (K2)** Explicar las habilidades y áreas de conocimiento esenciales que necesitan los probadores para trabajar eficazmente con la IA generativa en los procesos de prueba.
- IAGen-5.2.2 (K1)** Recordar estrategias para cultivar habilidades de IA dentro de los equipos de pruebas para apoyar la adopción de IA generativa en actividades de prueba.
- IAGen-5.2.3 (K1)** Reconocer cómo cambian los procesos y responsabilidades de prueba dentro de una organización de pruebas al adoptar IA generativa.

## 5.1 Guía para la adopción de la IA generativa en la prueba de software

Una estrategia de prueba con IAGen debe tener en cuenta cuidadosamente aspectos clave como los objetivos de la prueba que se desean alcanzar, la selección adecuada del MLG, los problemas con los datos de entrada utilizados para las instrucciones y el cumplimiento de las normas y regulaciones de IA. Basándose en esta estrategia, la organización puede establecer un plan de acción y supervisar el avance en la integración de IAGen en los procesos de prueba.

### 5.1.1 Riesgos de IA en la sombra

La IA en la sombra puede conllevar riesgos en materia de seguridad, cumplimiento de normativas y privacidad de datos:

- Debilidades en seguridad y privacidad de datos: las herramientas de IA personales pueden carecer de una seguridad sólida, lo que puede dar lugar a posibles violaciones de datos.
- Cuestiones normativas y de cumplimiento: el uso de herramientas de IA no aprobadas puede dar lugar al incumplimiento de estándares y normativas del sector (véase la sección 3.4.1), lo que podría acarrear consecuencias legales.
- Propiedad intelectual difusa: el uso de herramientas de IA con acuerdos de licencia poco claros puede exponer a los usuarios de MLG a disputas sobre propiedad intelectual, especialmente si se procesan datos protegidos por derechos de autor sin la debida autorización.

Una estrategia y unos pasos para integrar e implementar IAGen pueden ayudar a las organizaciones de pruebas a evitar el riesgo de la IA en la sombra.

### 5.1.2 Aspectos clave de una estrategia de IA generativa en la prueba de software

Para implementar con éxito una estrategia de IAGen en la prueba, las organizaciones deben considerar cuidadosamente varios factores clave para garantizar una integración fluida y resultados óptimos. Esto comienza con la definición de objetivos de prueba medibles para IAGen, como aumentar la productividad de la prueba, acortar los ciclos de prueba y mejorar la calidad de la prueba. La selección de los MLG adecuados es fundamental (véase la sección 5.1.3) y debe estar en consonancia con estos objetivos de prueba, al tiempo que se garantiza la compatibilidad con la infraestructura de prueba existente y se cumplan los requisitos de escalabilidad del sistema.

La calidad de los datos desempeña un papel fundamental, ya que la efectividad de la prueba basada en MLG depende de datos de entrada precisos y relevantes, protegidos por procedimientos de seguridad robustos. Manteniendo datos de entrada de alta calidad es, por lo tanto, clave para lograr resultados en los que se puede confiar que son correctos.

Se deben proporcionar programas de formación integrales para garantizar que los equipos de prueba cuenten con las habilidades técnicas y éticas necesarias para utilizar las herramientas de IAGen de manera eficaz. Además de la formación, se deben recopilar métricas específicas para medir la eficacia de los resultados de IAGen (véase la sección 2.3.1).

Para asegurar el cumplimiento de los estándares regulatorios y la adhesión a las directrices éticas, las organizaciones deben establecer directrices de proceso para el uso de IAGen, incluyendo reglas para el uso de datos sensibles, obligaciones de transparencia (por ejemplo, qué se generó utilizando IAGen) y controles de calidad con revisión del producto de prueba generado.

### 5.1.3 Selección de MLG/MLP para tareas de prueba de software

Existe una amplia gama de MLG/MLP, cada uno con diferentes capacidades funcionales (por ejemplo, entrada multimodal, capacidades de razonamiento), características técnicas (por ejemplo, tamaño de

la ventana de contexto) y tipos de licencia (por ejemplo, comercial frente a código abierto). Aunque existen muchos puntos de referencia para evaluar los MLG/MLP en tareas como el procesamiento del lenguaje natural (PLN), la generación de código o el análisis de imágenes, solo unos pocos se centran específicamente en tareas de prueba de software (Wenhan 2024). Por lo tanto, la selección de MLG/MLP para tareas de prueba requiere una cuidadosa consideración de varios criterios clave:

- Rendimiento del modelo: evaluar el rendimiento del modelo para las tareas de prueba específicas en comparación con los puntos de referencia de la organización utilizando métricas como las presentadas en la sección 2.3.1.
- Potencial de ajuste fino: evaluar si es posible y útil ajustar el modelo de lenguaje (MLG o MLP) con datos específicos del dominio para mejorar el rendimiento en un caso de uso determinado, aumentando la precisión y la relevancia en contextos especializados.
- Coste recurrente: Es necesario tener en cuenta los costes recurrentes del uso del MLG/MLP, incluidas las tasas de licencia y los gastos operativos, para garantizar que se ajusta al presupuesto de la organización para las tareas de prueba previstas.
- Comunidad y soporte: Elegir modelos con soporte activo de la comunidad y documentación detallada para ayudar en la implementación y la resolución de problemas.

Al evaluar cuidadosamente estos criterios, las organizaciones de prueba pueden seleccionar uno o más MLG/MLP que satisfagan sus necesidades específicas y limitaciones organizativas.

HO-5.1.3 - (H1) - Estimar los costes recurrentes del uso de la IA generativa para una tarea de prueba determinada.

Este ejercicio se centra en estimar los costes recurrentes del uso de la IA generativa para una tarea de prueba específica basándose en diversas hipótesis. Entre ellas se incluyen factores como el número de tokens en los datos de entrada y salida, las instrucciones utilizadas y la frecuencia de la tarea. Se explorarán y compararán los modelos de precios de varios proveedores de MLG/MLP, incluyendo al menos una solución comercial y un modelo con licencia de código abierto.

Este ejercicio ofrece la oportunidad de calcular y experimentar con los costes recurrentes de la IA generativa utilizando condiciones de prueba prácticas, lo que ayuda a comprender las implicaciones financieras de los diferentes enfoques y proveedores.

### 5.1.4 Fases de la adopción de la IA generativa en la prueba de software

La adopción de la IA generativa en una organización de prueba consta de tres fases clave:

1. Descubrimiento: la primera fase se centra en la concienciación y el desarrollo de capacidades. Las actividades incluyen la formación de los equipos de pruebas en los conceptos de la IA generativa, el acceso a MLG/MLP y la experimentación con casos de uso iniciales para familiarizar a los probadores con la IA generativa y generar confianza.
2. Inicio y definición del uso: una vez establecida la comprensión básica, la segunda fase se centra en identificar y priorizar casos de uso prácticos para la IA generativa en la prueba de software. Esta fase incluye la evaluación de la infraestructura de prueba impulsada por MLG, el desarrollo de conocimientos especializados y el aseguramiento de su adecuación a las necesidades de la organización (véase [ISTQB\_CTFL\_SYL], sección 6).
3. Uso e iteración: en esta fase avanzada, las organizaciones integran completamente IAGen en sus procesos de prueba. Se lleva a cabo un seguimiento continuo del progreso de IAGen para la prueba de software y las herramientas relacionadas, así como la medición y la gestión de la transformación para garantizar beneficios sostenibles y escalabilidad.

Estas fases pueden ejecutarse en paralelo para diferentes casos de uso. Por ejemplo, el análisis de informes de pruebas puede estar más avanzado en la hoja de ruta, mientras que la automatización de la prueba se encuentra en sus primeras fases. También es importante reconocer y abordar las

preocupaciones iniciales, como el temor a la pérdida de puestos de trabajo, que pueden afectar a la adopción y a la moral del equipo.

## 5.2 Gestión del cambio al adoptar la IA generativa en la prueba de software

La implementación exitosa de IAGen en una organización de prueba requiere un enfoque estructurado de los procesos de gestión del cambio. Los aspectos clave incluyen el desarrollo de habilidades esenciales en IAGen y la evolución de las funciones tradicionales de pruebas para adoptar procesos de pruebas basados en IA. La transformación implica tanto habilidades técnicas como aspectos organizativos.

### 5.2.1 Habilidades y conocimientos esenciales para probar con IA generativa

La integración exitosa de IAGen en la prueba requiere dominar las técnicas de ingeniería de instrucciones, comprender las ventanas de contexto de los modelos y desarrollar métodos de revisión de la prueba. Los probadores deben combinar su experiencia en el dominio y en prueba con habilidades de IA para evaluar pruebas guiadas por MLG en tareas tales como la generación de casos de prueba, el análisis de informes de defectos y la generación de datos de prueba.

Las competencias clave incluyen la evaluación de las capacidades de los MLG, la comprensión de las técnicas de refinamiento de las instrucciones y la evaluación de los productos de prueba generados por IA. Los conocimientos esenciales incluyen la comprensión de los riesgos inherentes a la IAGen, junto con el conocimiento de las estrategias de mitigación habituales. Los probadores deben comprender las implicaciones de seguridad de los datos que conlleva compartir productos de prueba con MLG, implementar una limpieza de datos adecuada (eliminando u ocultando información sensible, personal o confidencial) y seguir prácticas de ingeniería de instrucciones que preserven la privacidad de los datos. Las consideraciones medioambientales incluyen la optimización de la selección de modelos y los patrones de uso para reducir la sobrecarga computacional, la selección de modelos del tamaño adecuado para las tareas de prueba y el equilibrio entre los beneficios de la automatización de la IA generativa y el impacto en el coste y el consumo de energía.

### 5.2.2 Desarrollo de capacidades de IA generativa en equipos de prueba

Es esencial adoptar un enfoque práctico para formar estratégicamente a los equipos de prueba en IAGen para llevar a cabo la prueba. Esto incluye practicar con varios MLG/MLP, seguir itinerarios de aprendizaje estructurados y desarrollar gradualmente los conocimientos técnicos mediante el intercambio dentro de la organización. La formación se centra en el desarrollo de habilidades prácticas a través de ejercicios guiados, el aprendizaje entre compañeros y la integración gradual de la IA en las tareas diarias de prueba.

Los miembros del equipo de prueba pasan de dominar la creación básica de instrucciones a utilizar técnicas más específicas, como las instrucciones específicas para la prueba. Un patrón de instrucción es una plantilla reutilizable para crear instrucciones eficaces que guíen a IAGen hacia resultados consistentes y fiables. Las comunidades internas de práctica apoyan el intercambio continuo de conocimientos, con reuniones periódicas para destacar las experiencias exitosas de IAGen, debatir los retos y perfeccionar las mejores prácticas. Estas comunidades promueven la mejora continua mediante el intercambio de librerías de patrones de instrucciones y la documentación de las lecciones aprendidas de IAGen para la implementación de pruebas en todos los proyectos y dominios.

### 5.2.3 Procesos de prueba en evolución en organizaciones de prueba asistidas por IA

La integración de IAGen transforma los procesos de prueba tradicionales de los probadores y gestores de pruebas dentro de las organizaciones de prueba.



Los probadores evolucionan de especialistas en diseño de pruebas y ejecución de pruebas a especialistas en pruebas asistidas por IA, combinando su experiencia en técnicas de prueba con habilidades para guiar y verificar los productos de prueba generados por IA. Sus tareas de prueba se amplían para incluir la revisión de los resultados generales basados en IA, el perfeccionamiento de las instrucciones y el mantenimiento de librerías de instrucciones específicas para la prueba.

Las responsabilidades de los gestores de prueba se actualizan para incluir el desarrollo de una estrategia de prueba basada en IA, la gestión de riesgos basada en IA y la supervisión y el control de los procesos de pruebas basados en IA. Los gestores de prueba se centran en equilibrar las capacidades humanas y de la IA, establecer marcos de gobernanza de la IA para casos de uso y garantizar que sus equipos de pruebas mantengan tanto las competencias de prueba tradicionales como los conocimientos de IA. Los gestores de prueba no solo dirigirán a los probadores humanos, sino que también coordinarán con los agentes de prueba impulsados por IAGen, lo que requiere nuevas habilidades de gestión para supervisar equipos híbridos de personas y herramientas IAGen.

## 6 Referencias

### 6.1 Estándares

- ISO/IEC 42001:2023 - (2023), Information technology — Artificial intelligence — Management system
- ISO/IEC 23053:2022 - (2022), Framework for Artificial Intelligence (AI) Systems Using Machine Learning (ML)

### 6.2 Documentos de ISTQB®

- [ISTQB\_CTFL\_SYL], ISTQB® Foundation Level Syllabus v4.0, 2023.

### 6.3 Glosario de ISTQB®

- ISTQB® Glossary, <https://glossary.istqb.org/>

### 6.4 Libros

- Winteringham M. (2024) Software Testing with Generative AI, Manning Publications (5 Mar. 2025), ISBN-13 : 978-1633437364, 10 Dec. 2024 - 304 pages.

### 6.5 Artículos

- (Berthelot 2024) Berthelot, Adrien, et al. "Estimating the environmental impact of Generative-AI services using an LCA-based methodology." *Procedia CIRP* 122 (2024): 707-712.
- (Gallegos 2024) Gallegos, Isabel O., et al. "Bias and fairness in large language models: A survey." *Computational Linguistics* (2024): 1-79.
- (Li 2024) Yihao Li, Pan Liu, Haiyang Wang, Jie Chu, W. Eric Wong, Evaluating Large Language Models for Software Testing, Computer Standards & Interfaces (2024), doi: <https://doi.org/10.1016/j.csi.2024.103942>
- (Luccioni 2024a) Luccioni, Sasha, Yacine Jernite, and Emma Strubell. "Power hungry processing: Watts driving the cost of AI deployment?." *The 2024 ACM Conference on Fairness, Accountability, and Transparency*. 2024.
- (Mailach 2024) Mailach, Alina, et al. "Practitioners' Discussions on Building LLM-based Applications for Production." *arXiv preprint arXiv:2411.08574* (2024).
- (Mirzadeh 2024) Mirzadeh, Iman et al. "GSM-Symbolic: Understanding the Limitations of Mathematical Reasoning in Large Language Models." *ArXiv abs/2410.05229* (2024)

- (NIST AI RMF 1.0) National Institute of Standards and Technology. Artificial Intelligence Risk Management Framework (AI RMF 1.0). NIST AI 100-1, U.S. Department of Commerce, 2023, <https://doi.org/10.6028/NIST.AI.100-1>.
- (Parthasarathy 2024) Parthasarathy, Venkatesh Balavadhani, et al. "The ultimate guide to fine-tuning LLM's from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities." arXiv preprint arXiv:2408.13296 (2024).
- (Schulhoff 2024) Schulhoff, S., "The Prompt Report: A Systematic Survey of Prompting Techniques", Art. no. arXiv:2406.06608, 2024. doi:10.48550/arXiv.2406.06608.
- (Shuyin 2023) Ouyang, Shuyin, et al. "LLM is Like a Box of Chocolates: the Non-determinism of ChatGPT in Code Generation." arXiv preprint arXiv:2308.02828 (2023).
- (Sinha 2024) Sinha, Megha, Sreekanth Menon, and Ram Sagar. "LLMOps: Definitions, Framework and Best Practices." 2024 International Conference on Electrical, Computer and Energy Technologies (ICECET. IEEE, 2024).
- (Wang 2024) Wang, Yanlin, et al. "Agents in Software Engineering: Survey, Landscape, and Vision." arXiv preprint arXiv:2409.09030 (2024).
- (Wenhan 2024) Wang, Wenhan, et al. "TESTEVAL: Benchmarking Large Language Models for Test Case Generation." arXiv preprint arXiv:2406.04531 (2024).
- (Zhao 2024) Zhao, Penghao, et al. "Retrieval-augmented generation for ai-generated content: A survey." arXiv preprint arXiv:2402.19473 (2024).

## 6.6 Páginas Web

- (AI Act 2024) European Commission. "European Approach to Artificial Intelligence." *Shaping Europe's Digital Future*, European Commission, <https://digital-strategy.ec.europa.eu/en/policies/european-approach-artificial-intelligence>. Accessed 24 Nov. 2024.
- (Heikkilä 2023) Heikkilä, M. (2023, December 1). Making an image with generative AI uses as much energy as charging your phone. MIT Technology Review. Retrieved from <https://www.technologyreview.com/2023/12/01/1084189/making-an-image-with-generative-ai-uses-as-much-energy-as-charging-your-phone/>
- (Luccioni 2024b) Luccioni, S. (2024, February 22). Generative AI's environmental costs are soaring. Nature. Retrieved from <https://www.nature.com/articles/d41586-024-00478-x>
- (Google Dev Glossary 2024) Google Developers. (n.d.). Machine learning glossary: Generative AI. Retrieved November 24, 2024, from <https://developers.google.com/machine-learning/glossary/generative>
- (MIT 2024) "Glossary of Terms: Generative AI Basics." \*MIT Sloan Teaching & Learning Technologies\*, MIT Sloan School of Management, <https://mitsloanedtech.mit.edu/ai/basics/glossary>. Accessed 24 Nov. 2024.

Las referencias anteriores apuntan a información disponible en Internet y en otros lugares. Aunque esas referencias fueron comprobadas en el momento de la publicación de este programa de estudio, el ISTQB® no se hace responsable de la disponibilidad de dichas referencias.





## 7 Anexo A - Objetivos de aprendizaje/nivel cognitivo de conocimiento

Los objetivos de aprendizaje específicos de este programa de estudio figuran al principio de cada capítulo. Cada tema del programa de estudio se examinará de acuerdo con su objetivo de aprendizaje.

Los objetivos de aprendizaje comienzan con un verbo de acción correspondiente a su nivel cognitivo de conocimiento, como se indica a continuación.

### Nivel 1: Recordar (K1)

El candidato recordará, reconocerá y rememorará un término o concepto.

**Verbos de acción:** Recordar, reconocer.

Ejemplos
Recordar los conceptos de la pirámide de prueba.
Reconocer los objetivos más habituales de la prueba.

### Nivel 2: Comprender (K2)

El candidato puede seleccionar las razones o explicaciones de las sentencias relacionadas con el tema y puede resumir, comparar, clasificar y dar ejemplos para el concepto de prueba.

**Verbos de acción:** Clasificar, comparar, diferenciar, distinguir, explicar, dar ejemplos, interpretar, resumir

Ejemplos	Observaciones
Clasificar las herramientas de prueba según su finalidad y las actividades de prueba que admiten.	Sin observaciones.
Comparar los diferentes niveles de prueba.	Se puede utilizar para buscar similitudes, diferencias o ambas cosas.
Diferenciar entre prueba y depuración.	Buscar diferencias entre conceptos.
Distinguir entre riesgos de proyecto y riesgos de producto.	Permite clasificar por separado dos (o más) conceptos.
Explicar el impacto del contexto en el proceso de prueba.	Sin observaciones.
Aportar ejemplos de por qué es necesario probar.	Sin observaciones.
Resumir las actividades del proceso de revisión del producto de trabajo.	Sin observaciones.

### Nivel 3: Aplicar (K3)

El candidato puede llevar a cabo un procedimiento cuando se enfrenta a una tarea conocida o seleccionar el procedimiento correcto y aplicarlo a un contexto determinado.

**Verbos de acción:** Aplicar, implementar, preparar, utilizar

Ejemplos	Observaciones
Aplicar el análisis del valor frontera para extraer casos de prueba a partir de requisitos dados.	Debería hacer referencia a un procedimiento, técnica, proceso, etc.
Implementar métodos de recopilación de métricas para apoyar los requisitos técnicos y de gestión.	
Preparar pruebas de instalabilidad para aplicaciones móviles.	
Utilizar la trazabilidad para monitorizar el avance de las pruebas en cuanto a su completitud y consistencia con los objetivos, la estrategia y el plan de pruebas.	Podría utilizarse en un objetivo de aprendizaje que quiera que el candidato sea capaz de utilizar una técnica o procedimiento. Similar a «aplicar».

### Referencia

(Para los niveles cognitivos de los objetivos de aprendizaje)

Anderson, L. W. and Krathwohl, D. R. (eds) (2001) A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives, Allyn & Bacon

## 8 Anexo B - Matriz de trazabilidad de los resultados de negocio con respecto a los objetivos de aprendizaje

En esta sección se enumeran los vínculos entre los resultados empresariales y los objetivos de aprendizaje de Probador Certificado en Prueba con IA generativa. Los objetivos prácticos no se mencionan en esta tabla, ya que cada HO está asociado a un único LO. Las relaciones entre un HO y un BO se establecen a través del LO al que está asociado el HO.

Resultados de Negocio		BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
<b>IAGEN-B01</b>	Comprender los conceptos fundamentales, las capacidades y las limitaciones de la IA generativa.	8				
<b>IAGEN-B02</b>	Desarrollar habilidades prácticas para formular instrucciones a modelos de lenguaje grande para la prueba de software.		10			
<b>IAGEN-B03</b>	Adquirir conocimientos sobre los riesgos y las medidas de mitigación del uso de la IA generativa para la prueba de software.			11		
<b>IAGEN-B04</b>	Adquirir conocimientos sobre las aplicaciones de las soluciones de IA generativa para la prueba de software.				19	
<b>IAGEN-B05</b>	Contribuir de manera eficaz a la definición e implementación de una estrategia y un plan de acción de IA generativa para la prueba de software dentro de una organización.					13

Capítulo/ sección/ subsección	Objetivos de Aprendizaje	Nivel - K	BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
<b>1</b>	<b>Introducción a la IA generativa para la prueba de software</b>						
<b>1.1</b>	<b>Fundamentos y conceptos clave de la IA generativa</b>						
<b>IAGEN-1.1.1</b>	Recordar los diferentes tipos de IA: IA simbólica, aprendizaje automático clásico, aprendizaje profundo e IA generativa.	<b>(K1)</b>	X				
<b>IAGEN-1.1.2</b>	Explicar los fundamentos de la IA generativa y los modelos de lenguaje grande.	<b>(K2)</b>	X				
<b>IAGEN-1.1.3</b>	Distinguir entre MLG fundacionales, ajustados por instrucciones y de razonamiento.	<b>(K2)</b>	X				
<b>IAGEN-1.1.4</b>	Resumir los principios básicos de los modelos de lenguaje grande multimodal y modelos de visión y lenguaje.	<b>(K2)</b>	X				
<b>1.2</b>	<b>Aprovechamiento de la IA generativa en la prueba de software: principios básicos</b>						
<b>IAGEN-1.2.1</b>	Aportar ejemplos de capacidades clave de los modelos de lenguaje grande para las tareas de prueba.	<b>(K2)</b>	X			X	
<b>IAGEN-1.2.2</b>	Comparar los modelos de interacción al utilizar IAGen para probar software.	<b>(K2)</b>	X			X	
<b>2</b>	<b>Ingeniería de instrucciones para la prueba de software efectiva</b>						



Capítulo/ sección/ subsección	Objetivos de Aprendizaje	Nivel - K	BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
<b>2.1</b>	<b>Desarrollo eficaz de instrucciones</b>						
<b>IAGEN-2.1.1</b>	Aportar ejemplos de la estructura de las instrucciones utilizadas en la IA generativa para la prueba de software.	(K2)		X			
<b>IAGEN-2.1.2</b>	Diferenciar las técnicas básicas de instrucción para prueba de software.	(K2)		X			
<b>IAGEN-2.1.3</b>	Distinguir entre instrucciones de sistema e instrucciones de usuario.	(K2)		X			
<b>2.2</b>	<b>Aplicación de técnicas de ingeniería de instrucciones a tareas de prueba de software</b>						
<b>IAGEN-2.2.1</b>	Aplicar la IA generativa a las tareas de análisis de prueba.	(K3)		X			
<b>IAGEN-2.2.2</b>	Aplicar la IA generativa al diseño de pruebas y a las tareas de implementación de pruebas.	(K3)		X			
<b>IAGEN-2.2.3</b>	Aplicar la IA generativa a la prueba de regresión automatizada.	(K3)		X			
<b>IAGEN-2.2.4</b>	Aplicar la IA generativa a las tareas de control y supervisión de la prueba.	(K3)		X			
<b>IAGEN-2.2.5</b>	Seleccionar y aplicar técnicas de instrucción adecuadas para un contexto y una tarea de prueba determinados.	(K3)		X		X	

Capítulo/ sección/ subsección	Objetivos de Aprendizaje	Nivel - K	BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
2.3	<b>Evaluación de los resultados de la IA generativa y mejora de las instrucciones para las tareas de prueba de software</b>						
IAGEN-2.3.1	Comprender las métricas para evaluar los resultados de la IA generativa en las tareas de prueba.	(K2)		X	X	X	
IAGEN-2.3.2	Aportar ejemplos de técnicas para evaluar y perfeccionar iterativamente las instrucciones.	(K2)		X	X	X	
3	<b>Gestión de riesgos de la IA generativa en la prueba de software</b>						
3.1	<b>3.1 Alucinaciones, errores de razonamiento y sesgos</b>						
IAGEN-3.1.1	Recordar las definiciones de alucinaciones, errores de razonamiento y sesgos en los sistemas de IA generativa.	(K1)	X		X	X	
IAGEN-3.1.2	Identificar alucinaciones, errores de razonamiento y sesgos en los resultados de MLG.	(K3)			X	X	
IAGEN-3.1.3	Resumir las técnicas de mitigación para las alucinaciones, los errores de razonamiento y los sesgos de IAGen en tareas de prueba de software.	(K2)			X	X	
IAGEN-3.1.4	Recordar las técnicas de mitigación para el comportamiento no determinista de los MLG.	(K1)	X		X	X	
3.2	<b>Privacidad de datos y riesgos de seguridad de la IA generativa en la prueba de software</b>						

Capítulo/ sección/ subsección	Objetivos de Aprendizaje	Nivel - K	BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
<b>IAGEN-3.2.1</b>	Explicar los principales riesgos para la privacidad de datos y el riesgo de seguridad asociados al uso de la IA generativa en la prueba de software.	(K2)			X	X	
<b>IAGEN-3.2.2</b>	Aportar ejemplos de privacidad de datos y vulnerabilidades en el uso de la IA generativa en la prueba de software.	(K2)			X	X	
<b>IAGEN-3.2.3</b>	Resumir las estrategias de mitigación para proteger la privacidad de los datos y mejorar la seguridad en la IA generativa para la prueba de software.	(K2)			X	X	
<b>3.3</b>	<b>Consumo de energía e impacto medioambiental de la IA generativa en la prueba de software</b>						
<b>IAGEN-3.3.1</b>	Explicar el impacto de las características de las tareas y el uso de modelos en el consumo energético de la IA generativa en la prueba de software.	(K2)			X	X	
<b>3.4</b>	<b>Normativas, estándares y marcos de trabajo de buenas prácticas en materia de IA</b>						
<b>IAGEN-3.4.1</b>	Recordar ejemplos de normativas, estándares y marcos de buenas prácticas relacionados con la IA generativa en la prueba de software.	(K3)			X	X	X
<b>4</b>	<b>Infraestructura de prueba impulsada por MLG para la prueba de software</b>						
<b>4.1</b>	<b>Enfoques arquitectónicos para la infraestructura de prueba impulsada por MLG</b>						
<b>IAGEN-4.1.1</b>	Explicar los componentes y conceptos arquitectónicos clave de la infraestructura de prueba impulsada por MLG.	(K2)				X	X

Capítulo/ sección/ subsección	Objetivos de Aprendizaje	Nivel - K	BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
IAGEN-4.1.2	Resumir la generación aumentada por recuperación.	(K2)				X	X
IAGEN-4.1.3	Explicar la función y la aplicación de los agentes impulsados por MLG en la automatización de los procesos de prueba.	(K2)				X	X
4.2	<b>Ajuste fino y OpsMLG: puesta en marcha de la IA generativa para la prueba de software</b>						
IAGEN-4.2.1	Explicar el ajuste fino de los modelos de lenguaje para tareas de prueba específicas.	(K2)				X	X
IAGEN-4.2.2	Explicar OpsMLG y su función en la implementación y gestión de MLG para tareas de prueba.	(K2)				X	X
5	<b>Despliegue e integración de IA generativa en organizaciones de prueba</b>						
5.1	<b>Guía para la adopción de la IA generativa en la prueba de software</b>						
IAGEN-5.1.1	Recordar los riesgos de la IA en la sombra.	(K1)					X
IAGEN-5.1.2	Explicar los aspectos clave a tener en cuenta al definir una estrategia de IA generativa para la prueba de software.	(K2)					X
IAGEN-5.1.3	Resumir los criterios clave para seleccionar MLG/MLP para tareas de prueba de software en un contexto determinado.	(K2)					X



Capítulo/ sección/ subsección	Objetivos de Aprendizaje	Nivel - K	BO - 01	BO - 02	BO - 03	BO - 04	BO - 05
IAGEN-5.1.4	Recordar las fases clave en la adopción de la IA generativa en una organización de prueba.	(K1)					X
5.2	Gestión del cambio al adoptar la IA generativa en la prueba de software						
IAGEN-5.2.1	Explicar las habilidades y áreas de conocimiento esenciales que necesitan los probadores para trabajar eficazmente con la IA generativa en los procesos de prueba.	(K2)					X
IAGEN-5.2.2	Recordar estrategias para cultivar habilidades de IA dentro de los equipos de pruebas para apoyar la adopción de IA generativa en actividades de prueba.	(K1)					X
IAGEN-5.2.3	Reconocer cómo cambian los procesos y responsabilidades de prueba dentro de una organización de pruebas al adoptar IA generativa.	(K1)					X

## **9 Anexo C - Notas de la versión**

Esta versión es la V1.0. No hay notas de entrega para esta primera versión.



## 10 Anexo D - Términos específicos de la IA generativa

Descripción en inglés	Término en inglés	Término en español	Descripción en español
A conversational agent that uses LLMs to process queries and generate human-like text responses, enabling interactive communication with users.	AI chatbot	chatbot con IA	Un agente de conversación que utiliza MLG para procesar consultas y generar respuestas de texto similares a las humanas, lo que permite una comunicación interactiva con los usuarios.
The span of text, measured in tokens, that a language model considers when generating responses, influencing the relevance and coherence of its outputs.	context window	ventana de contexto	La extensión del texto, medida en tóquenes, que un modelo de lenguaje tiene en cuenta al generar respuestas, de modo que influye en la relevancia y la coherencia de sus resultados.
ML using neural networks with multiple layers.	deep learning	aprendizaje profundo	Aprendizaje automático (AA) que utiliza redes neuronales con múltiples capas.
A technique used to represent tokens as dense vectors in a continuous space, learned during training to capture semantic, syntactic, and contextual relationships.	embedding	incrustar	Técnica utilizada para representar tóquenes como vectores densos en un espacio continuo, aprendida durante el entrenamiento para capturar relaciones semánticas, sintácticas y contextuales.
An individual measurable attribute of the input data used for training by an ML algorithm and for prediction by an ML model.	feature	característica	Atributo individual medible de los datos de entrada utilizados para el entrenamiento por un algoritmo de aprendizaje automático y para la predicción por un modelo de aprendizaje automático.
A technique where a model is given a few examples within the prompt to guide it in generating appropriate responses.	few-shot prompting	formulación de instrucciones con pocos ejemplos	Técnica en la que se proporcionan al modelo pocos ejemplos dentro de la instrucción para guiarlo en la generación de respuestas adecuadas.
A supervised learning process using a dataset of labeled examples to update LLM weights and adapt them for specific tasks or domains.	fine-tuning	ajuste fino	Proceso de aprendizaje supervisado que utiliza un conjunto de datos de ejemplos etiquetados para actualizar los pesos del modelo de lenguaje grande y adaptarlos a tareas o dominios específicos.

Descripción en inglés	Término en inglés	Término en español	Descripción en español
General-purpose models pre-trained on a wide range of text data, capable of predicting the next word based on learned linguistic patterns. Synonym: Base LLM	<b>foundation LLM</b>	<b>MLG fundacional</b>	Modelos de uso general preentrenados con una amplia gama de datos de texto, capaces de predecir la siguiente palabra basándose en patrones lingüísticos aprendidos. Sinónimo: MLG base.
A type of artificial intelligence system that uses machine learning models to generate (new) intellectual content that resembles human-created content.	<b>generative AI (GenAI)</b>	<b>IA generativa (IAGen)</b>	Un tipo de sistema de inteligencia artificial que utiliza modelos de aprendizaje automático para generar (nuevo) contenido intelectual que se asemeja al contenido creado por humanos.
A type of transformer-based deep learning model pre-trained on vast amounts of text data to understand and generate human-like text.	<b>generative pre-trained transformer (GPT)</b>	<b>transformador generativo preentrenado (TGP)</b>	Un tipo de modelo de aprendizaje profundo basado en transformadores, preentrenado con grandes cantidades de datos de texto para comprender y generar texto similar al humano.
Wrong information created by an LLM.	<b>hallucination</b>	<b>alucinación</b>	Información errónea creada por un MLG.
A foundation LLM trained to follow instructions, often reinforced by feedback to encourage correct answers.	<b>Instruction-tuned LLM</b>	<b>MLG ajustado por instrucciones</b>	Un MLG básico entrenado para seguir instrucciones, a menudo reforzado con comentarios para fomentar las respuestas correctas.
		<b>modelo de lenguaje grande ajustado por instrucciones</b>	
A computer program that uses very large collections of language data in order to understand and produce text in a way that is similar to the way humans do.	<b>large language model (LLM)</b>	<b>modelo de lenguaje grande (MLG)</b>	Un programa informático que utiliza grandes colecciones de datos lingüísticos para comprender y producir texto de una manera similar a como lo hacen los seres humanos.
An application that integrates LLM reasoning, decision-making, and memory, using tools to perform tasks.	<b>LLM-powered agent</b>	<b>agente impulsado por MLG</b>	Una aplicación que integra el razonamiento, la toma de decisiones y la memoria de los MLG, utilizando herramientas para realizar tareas.
Practices and tools focused on deploying, monitoring, and maintaining LLMs in production environments.	<b>LLMOps</b>	<b>OpsMLG</b>	Prácticas y herramientas centradas en la implementación, supervisión y mantenimiento de los MLG en entornos de producción.
The process using computational techniques to enable systems to learn from data or experience (ISO/IEC TR 29119-11).	<b>machine learning (ML)</b>	<b>aprendizaje automático (AA)</b>	Proceso que utiliza técnicas de cómputo para permitir que los sistemas aprendan a partir de datos o experiencias (ISO/IEC TR 29119-11).



Descripción en inglés	Término en inglés	Término en español	Descripción en español
The crafting of higher-level instructions that generate specific prompts for exploring or automating capabilities.	<b>meta-prompting</b>	<b>metainstrucción</b>	Creación de instrucciones de alto nivel que generan instrucciones específicas para explorar o automatizar capacidades.
GenAI models that are capable of processing and generating content across multiple data types, such as text, images, and audio.	<b>multimodal model</b>	<b>modelo multimodal</b>	Modelos IAGen capaces de procesar y generar contenido en múltiples tipos de datos, como texto, imágenes y audio.
The processing of data encoded in natural language by computers to retrieve information and for knowledge representation.	<b>natural language processing (NLP)</b>	<b>procesamiento del lenguaje natural (PLN)</b>	El procesamiento de datos codificados en lenguaje natural por ordenadores para recuperar información y representar conocimientos.
A prompt writing technique where the prompt contains one example to guide the LLM's response.	<b>one-shot prompting</b>	<b>instrucción de un ciclo</b>	Una técnica de redacción de instrucciones en la que la instrucción contiene un ejemplo para guiar la respuesta del MLG.
A natural language input provided to elicit specific response in Generative AI and large language models.	<b>prompt</b>	<b>instrucción</b>	Entrada de lenguaje natural proporcionada para obtener una respuesta específica en la IA generativa y los modelos de lenguaje grande.
A prompting technique that involves using the output of one prompt as the input for another, creating a sequence of prompts.	<b>prompt chaining</b>	<b>encadenamiento de instrucciones</b>	Técnica de formulación de instrucciones que consiste en utilizar el resultado de una instrucción como entrada para otra, creando una secuencia de instrucciones.
The process of designing and refining input prompts to guide LLMs toward producing desired outputs.	<b>prompt engineering</b>	<b>ingeniería de instrucciones</b>	El proceso de diseñar y perfeccionar las instrucciones de entrada para guiar a los MLG hacia la producción de los resultados deseados.
An LLM building upon instruction-tuned models by refining their ability to emulate human-like reasoning processes	<b>reasoning LLM</b>	<b>MLG de razonamiento</b>	Un MLG desarrollado a partir de modelos ajustados por instrucciones, que refina su capacidad para emular procesos de razonamiento similares a los humanos.
A technique combining LLM capabilities with a retriever to fetch relevant data for generating accurate, contextually relevant responses.	<b>retrieval-augmented generation (RAG)</b>	<b>generación aumentada por recuperación (GAR)</b>	Una técnica que combina las capacidades de MLG con un recuperador para obtener datos relevantes con el fin de generar respuestas precisas y contextualmente relevantes.

Descripción en inglés	Término en inglés	Término en español	Descripción en español
The use of GenAI tools or systems within an organization without formal approval or oversight.	<b>shadow AI</b>	<b>IA en la sombra</b>	El uso de herramientas o sistemas IAGen dentro de una organización sin aprobación formal ni supervisión.
Language models that are intentionally designed and trained to be small, offering a balance between efficiency and task-specific language understanding.	<b>small language model (SML)</b>	<b>modelo de lenguaje pequeño (MLP)</b>	Modelos de lenguaje diseñados y entrenados intencionadamente para ser pequeños, que ofrecen un equilibrio entre la eficiencia y la comprensión del lenguaje específica para cada tarea.
An AI approach that uses symbols, rules, and structured knowledge to model reasoning.	<b>symbolic AI</b>	<b>IA simbólica</b>	Un enfoque de IA que utiliza símbolos, reglas y conocimientos estructurados para modelar el razonamiento.
A predefined instruction set, typically hidden from the chatbot's users, that consistently establishes the context, tone, and boundaries for an LLM's responses and guides its behavior throughout interactions.	<b>system prompt</b>	<b>instrucción de sistema</b>	Un conjunto de instrucciones predefinidas, normalmente ocultas a los usuarios del bot de conversación, que establece de forma coherente el contexto, el tono y los límites de las respuestas de un MLG y guía su comportamiento a lo largo de las interacciones.
A parameter that controls the randomness or creativity of a LLM's outputs.	<b>temperature</b>	<b>temperatura</b>	Un parámetro que controla la aleatoriedad o la creatividad de los resultados de un MLG.
The process of breaking down text into smaller units for processing by language models.	<b>tokenization</b>	<b>toquenización</b>	Proceso de descomposición del texto en unidades más pequeñas para su procesamiento por modelos de lenguaje.
A deep learning model architecture that utilizes self-attention mechanisms to capture long-range dependencies in input sequences.	<b>transformer</b>	<b>transformador</b>	Una arquitectura de modelo de aprendizaje profundo que utiliza mecanismos de autoatención para capturar dependencias de largo alcance en secuencias de entrada.
An instruction or query entered by a user into a Large Language Model (LLM) that directs the model's response to fulfil specific tasks or provide desired information.	<b>user prompt</b>	<b>instrucción de usuario</b>	Una instrucción o consulta introducida por un usuario en un modelo de lenguaje grande (MLG) que dirige la respuesta del modelo para realizar tareas específicas o proporcionar la información deseada.

Descripción en inglés	Término en inglés	Término en español	Descripción en español
A database optimized for storing and querying high-dimensional vector representations of data	<b>vector database</b>	<b>base de datos vectorial</b>	Una base de datos optimizada para almacenar y consultar representaciones vectoriales de datos de alta dimensión.
A GenAI system that jointly processes visual and textual data to perform tasks by linking and generating content across both modalities.	<b>vision-language model</b>	<b>modelo de visión y lenguaje</b>	Un sistema IAGen que procesa conjuntamente datos visuales y textuales para realizar tareas mediante la vinculación y la generación de contenido en ambas modalidades.
A prompt writing technique where the prompt contains no examples, relying on the model's pre-existing knowledge to generate a response.	<b>zero-shot prompting</b>	<b>formulación de instrucciones sin ejemplos</b>	Una técnica de escritura de instrucciones en la que la instrucción no contiene ejemplos, sino que se basa en el conocimiento preexistente del modelo para generar una respuesta.

## 11 Anexo E - Marcas registradas

ISTQB® es una marca registrada de International Software Testing Qualifications Board.