UNIVERSIDAD PRIVADA DE TACNA

ESCUELA DE POSTGRADO

MAESTRÍA EN INGENIERÍA DE SOFTWARE

GESTIÓN DE RIESGOS Y CALIDAD DE SOFTWARE REALIZADO POR PROFESIONALES DEL CIP, TACNA – 2024

TESIS

Presentado por:

Bach. Enrique Waldo Condori Soles
ORCID: 0009-0001-0514-4287

Asesor:

Dr. Hugo Manuel Barraza Vizcarra
ORCID: 0000-0001-5000-0250

Para obtener el grado académico de:

MAESTRO EN INGENIERÍA DE SOFTWARE

TACNA – PERÚ

2026

UNIVERSIDAD PRIVADA DE TACNA

ESCUELA DE POSTGRADO

MAESTRÍA EN INGENIERÍA DE SOFTWARE

Tesis

"GESTIÓN DE RIESGOS Y CALIDAD DE SOFTWARE REALIZADO POR PROFESIONALES DEL CIP, TACNA – 2024"

Presentada por:

Bach. Enrique Waldo Condori Siles

Tesis sustentada y aprobada el 22 de mayo de 2026; ante el siguiente jurado examinador:

PRESIDENTE	:	DRA. MARTHA JUDITH PAREDES VIGNOLA
SECRETARIO	:	MTRO. TITO FERNANDO ALE NIETO
VOCAL	:	MAG. ELARD RICARDO RODRÍGUEZ MARCA
ASESOR	:	DR. HUGO MANUEL BARRAZA VIZCARRA

DECLARACIÓN JURADA DE ORIGINALIDAD

Yo Enrique Waldo Condori Siles, en calidad de: egresado de la Maestría de Ingeniería de Software de la Escuela de Postgrado de la Universidad Privada de Tacna, identificado con DNI 71818758.

Soy autor de la tesis titulada:

GESTIÓN DE RIESGOS Y CALIDAD DE SOFTWARE REALIZADO POR PROFESIONALES DEL CIP, TACNA – 2024, con asesor: Dr. Hugo Manuel Barraza Vizcarra.

DECLARO BAJO JURAMENTO

Ser el único autor del texto entregado para obtener el grado académico de Maestro en Ingeniería de Software, y que tal texto no ha sido entregado ni total ni parcialmente para obtención de un grado académico en ninguna otra universidad o instituto, ni ha sido publicado anteriormente para cualquier otro fin.

Así mismo, declaro no haber trasgredido ninguna norma universitaria con respecto al plagio ni a las leyes establecidas que protegen la propiedad intelectual.

Declaro, que después de la revisión de la tesis con el software Turnitin se declara 14% de similitud, además que el archivo entregado en formato PDF corresponde exactamente al texto digital que presento junto al mismo.

Por último, declaro que para la recopilación de datos se ha solicitado la autorización respectiva a la empresa u organización, evidenciándose que la información presentada es real y soy conocedor de las sanciones penales en caso de infringir las leyes del plagio y de falsa declaración, y que firmo la presente con pleno uso de mis facultades y asumiendo todas las responsabilidades de ella derivada.

Por lo expuesto, mediante la presente asumo frente a LA UNIVERSIDAD cualquier responsabilidad que pudiera derivarse por la autoría, originalidad y veracidad del contenido de la tesis, así como por los derechos sobre la obra o invención

presentada. En consecuencia, me hago responsable frente a LA UNIVERSIDAD y a terceros, de cualquier daño que pudiera ocasionar, por el incumplimiento de lo declarado o que pudiera encontrar como causa del trabajo presentado, asumiendo todas las cargas pecuniarias que pudieran derivarse de ello en favor de terceros con motivo de acciones, reclamaciones o conflictos derivados del incumplimiento de lo declarado o las que encontrasen causa en el contenido de la tesis, libro o invento.

De identificarse fraude, piratería, plagio, falsificación o que el trabajo de investigación haya sido publicado anteriormente; asumo las consecuencias y sanciones que de mi acción se deriven, sometiéndome a la normatividad vigente de la Universidad Privada de Tacna.

Lugar y fecha: Tacna, 22 de mayo de 2026

Enrique Waldo Condori Siles

DNI: 71818758

DEDICATORIA

A mi amada esposa, Claudia, por su infinita paciencia y comprensión durante el proceso de esta investigación. Tu apoyo silencioso pero constante fue fundamental. Gracias por todo tu amor y sacrificio.

AGRADECIMIENTOS

Con todo mi amor y profunda gratitud a mis padres, cuyo apoyo incondicional y amor infinito fueron el motor que impulsó este logro.

A mi asesor de tesis, Dr. Hugo Manuel Barraza Vizcarra, por su guía experta, su paciencia y por compartir su invaluable conocimiento.

A mis amigos y colegas que creyeron en mi y no dudaron en darme ánimos para realizar y culminar esta investigación.

ÍNDICE DE CONTENIDOS

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

La investigación tuvo como finalidad determinar la relación entre la gestión de riesgos y la calidad del software desarrollado por profesionales del Colegio de Ingenieros del Perú – Consejo Departamental Tacna. Se aplicó un enfoque cuantitativo con diseño correlacional, no experimental y transversal. Se encuestó a 255 ingenieros colegiados, procesando los datos en SPSS. Los resultados evidencian que el 70,59 % de los encuestados presenta un nivel medio en gestión de riesgos y el 54,51 % un nivel alto en calidad de software. En cuanto a las dimensiones de calidad, el 45,88 % alcanza un nivel alto en funcionalidad, el 40,00 % en fiabilidad, el 40,78 % en rendimiento y el 29,41 % en mantenibilidad. La prueba de Spearman reportó una correlación significativa y positiva alta (ρ = 0,884; $p < 0,001$) entre ambas variables. A nivel de hipótesis específicas, se hallaron correlaciones significativas entre gestión de riesgos y funcionalidad (ρ = 0,820), fiabilidad (ρ = 0,686), rendimiento (ρ = 0,728) y mantenibilidad (ρ = 0,715). Se concluye que una adecuada gestión de riesgos impacta favorablemente en todos los aspectos clave de la calidad del software.

Palabras clave: Gestión de riesgos, calidad de software, ingeniería de software.

ABSTRACT

The objective of this study was to determine the relationship between risk management and software quality developed by professionals from the College of Engineers of Peru – Tacna Chapter. A quantitative, non-experimental, correlational and cross-sectional design was used. A total of 255 engineers were surveyed and the data were processed using SPSS. The results showed that 70.59% of participants had a medium level of risk management, while 54.51% reached a high level in software quality. Specifically, 45.88% scored high in functionality, 40.00% in reliability, 40.78% in performance, and 29.41% in maintainability. The Spearman correlation test revealed a strong and statistically significant positive correlation ($\rho = 0.884$; $p < 0.001$) between the two main variables. Additionally, strong correlations were found between risk management and functionality ($\rho = 0.820$), reliability ($\rho = 0.686$), performance ($\rho = 0.728$), and maintainability ($\rho = 0.715$). It is concluded that effective risk management contributes positively to all key dimensions of software quality.

Keywords: Risk management, software quality, software engineering.

INTRODUCCIÓN

En el contexto actual de transformación digital, el desarrollo de software enfrenta desafíos crecientes relacionados con la calidad de los productos entregados y la gestión de los riesgos inherentes a cada proyecto. La acelerada evolución tecnológica, la presión por reducir los tiempos de entrega y la complejidad de los entornos informáticos modernos exigen que los equipos de desarrollo adopten enfoques sistemáticos que garanticen no solo la funcionalidad del software, sino también su fiabilidad, rendimiento y mantenibilidad a lo largo del tiempo. En este panorama, la gestión de riesgos se consolida como una práctica estratégica que permite anticipar contingencias, optimizar recursos y asegurar productos tecnológicos de mayor valor agregado.

La gestión de riesgos, entendida como un proceso integral de identificación, evaluación, tratamiento y monitoreo de eventos potenciales que puedan afectar los objetivos del proyecto, es especialmente crítica en el ámbito del desarrollo de software. Su aplicación permite reducir la probabilidad de fallos graves, garantizar la continuidad operativa, mejorar la planificación y aumentar la confianza de los stakeholders. Por otro lado, la calidad del software no solo abarca el cumplimiento de los requisitos funcionales, sino también su desempeño en condiciones reales de uso, su adaptabilidad ante cambios y su facilidad de mantenimiento. La relación entre ambas dimensiones —gestión de riesgos y calidad del software— resulta particularmente relevante para asegurar resultados sostenibles y alineados con los estándares internacionales de ingeniería del software.

En ese sentido, la presente investigación tiene como propósito analizar la relación existente entre la gestión de riesgos y la calidad del software en proyectos

desarrollados por profesionales del Capítulo de Ingeniería de Sistemas del Colegio de Ingenieros del Perú – Consejo Departamental Tacna. Este grupo representa un sector estratégico para el desarrollo regional, dado su involucramiento en proyectos informáticos de impacto público y privado. El estudio centra su atención en cómo las prácticas de gestión de riesgos implementadas en estos entornos influyen en variables clave de calidad, tales como la funcionalidad, fiabilidad, rendimiento y mantenibilidad del software desarrollado.

Este trabajo se estructura en cinco capítulos. El Capítulo I desarrolla la problemática, justificación, objetivos y las hipótesis de investigación, así como la operacionalización de variables. El Capítulo II expone el marco teórico, abordando antecedentes, bases conceptuales y científicas vinculadas al tema. En el Capítulo III, se describe la metodología empleada, especificando el tipo y diseño de investigación, población, muestra, instrumentos y técnicas de recolección y análisis de datos. El Capítulo IV presenta los resultados del análisis estadístico, tanto descriptivo como inferencial, y discute los hallazgos en función de la literatura previa. Finalmente, el Capítulo V formula las conclusiones obtenidas y plantea recomendaciones orientadas a mejorar la gestión de riesgos en el desarrollo de software profesional.

CAPÍTULO I: EL PROBLEMA

1.1. Planteamiento del problema

El desarrollo de software enfrenta desafíos persistentes relacionados con la calidad del producto y la gestión de riesgos durante el ciclo de vida del proyecto, a nivel internacional, diversos estudios han señalado que más del 50% de los proyectos de tecnologías de la información (TI) fracasan en cumplir sus objetivos de calidad, costo o tiempo, siendo la gestión inadecuada de riesgos una de las principales causas de estos resultados deficientes (Arias, Ferro, & Abuchar, 2019). Organismos como el Project Management Institute (PMI) destacan que la ausencia de procesos sistemáticos de identificación, análisis y control de riesgos incrementa la probabilidad de fallos críticos, especialmente en entornos ágiles y altamente demandantes (Morales, 2018).

En el contexto latinoamericano, esta problemática se agrava debido a limitaciones estructurales, culturales y metodológicas, en países como Colombia, se ha evidenciado que la baja adopción de modelos como PMBOK o del enfoque PHVA conlleva a que muchos equipos de desarrollo minimicen o ignoren la gestión de riesgos, lo que redunda en software con deficiencias funcionales, de rendimiento o mantenibilidad (Morales, 2018). A pesar de las recomendaciones internacionales, aún persiste la percepción errónea de que la gestión de riesgos inhibe la innovación o retrasa los proyectos (Benites, 2022).

A nivel nacional, el avance de la transformación digital ha impulsado el interés por mejorar la calidad del software, instituciones públicas y privadas han comenzado a adoptar estándares como ISO/IEC 25010 para evaluar atributos de calidad, tales como confiabilidad, eficiencia y usabilidad (Paucar, Acho, & Peralta,

2021). Sin embargo, se identifica una brecha importante: la gestión de riesgos no se encuentra adecuadamente integrada como parte estratégica del aseguramiento de la calidad del software, en muchos casos, los riesgos se gestionan de forma reactiva, y no preventiva, lo que compromete la entrega de productos robustos, seguros y sostenibles (Bravo, 2017).

A nivel local, en la región de Tacna, los ingenieros colegiados del Capítulo de Sistemas del Colegio de Ingenieros del Perú (CIP) participan activamente en proyectos de desarrollo de software tanto para el sector público como privado, no obstante, muchos de estos profesionales enfrentan dificultades para implementar enfoques sistemáticos de gestión de riesgos en sus proyectos, lo cual se traduce en productos con limitaciones funcionales, fallas en producción o baja satisfacción del usuario final (Bravo, 2017). Esta situación se ve agravada por la escasa capacitación formal en gestión de riesgos y la limitada cultura organizacional orientada a la prevención (Morales, 2018).

Específicamente en el contexto de los profesionales adscritos al CIP Tacna, las causas fundamentales que originan esta deficiente gestión de riesgos y baja calidad de software son multifactoriales. En primer lugar, destaca la presión por entregas rápidas y los presupuestos ajustados del mercado local, lo que frecuentemente obliga a los ingenieros a priorizar la codificación directa en detrimento de las fases de planificación y aseguramiento. Como advierte Pressman (2010), ceder ante la presión del cronograma omitiendo prácticas formales de ingeniería de software es una de las causas principales que incrementan la vulnerabilidad del proyecto y degradan la calidad del producto final. En segundo lugar, existe una carencia de herramientas metodológicas escaladas a la realidad local, lo que lleva a percibir los marcos formales como burocráticos para equipos pequeños. Finalmente, la dinámica laboral en la región evidencia una ausencia de roles especializados en calidad (QA), provocando que la mitigación de riesgos recaiga de manera empírica sobre el mismo desarrollador; un patrón de acumulación de roles y falta de procesos preventivos que Paucar, Acho & Peralta

(2021) también identificaron como una causa limitante en su investigación con los profesionales del CIP en Lima.

La adopción de buenas prácticas internacionales, como las promovidas por ISO y el PMI, y el uso de herramientas como sistemas basados en casos (Case-Based Reasoning) para la gestión de riesgos, puede permitir una identificación y mitigación más efectiva de los factores que afectan la calidad del software (Ramirez, 2022). Además, fomentar una cultura organizacional que valore la gestión de riesgos como un proceso estratégico, y no como una carga administrativa, resulta crucial para elevar los estándares técnicos y funcionales de los productos desarrollados (Benites, 2022).

1.2. Formulación del problema

1.2.1. Interrogante principal

¿Cómo se relaciona la gestión de riesgos y la calidad del software realizado por profesionales del CIP, Tacna – 2024?

1.2.2. Interrogantes secundarias

¿Cuál es la relación de la gestión de riesgos y la funcionalidad del software realizado por profesionales del CIP, Tacna – 2024?

¿Cuál es la relación de la gestión de riesgos y la fiabilidad del software realizado por profesionales del CIP, Tacna – 2024?

¿Cuál es la relación de la gestión de riesgos y el rendimiento del software realizado por profesionales del CIP, Tacna – 2024?

¿Cuál es la relación de la gestión de riesgos y la mantenibilidad del software realizado por profesionales del CIP, Tacna – 2024?

1.3. Justificación de la investigación

La investigación tuvo como propósito ampliar el conocimiento sobre la relación entre la gestión de riesgos y la calidad del software, integrando dimensiones clave como los procesos, la gestión de recursos financieros y el talento

humano. Al analizar su influencia en aspectos como la funcionalidad, la fiabilidad, el rendimiento y la mantenibilidad del software, se buscó enriquecer la literatura existente y aportar elementos teóricos que sirvieran de base para futuras investigaciones orientadas a mejorar los modelos de gestión en ingeniería de software.

Desde el enfoque metodológico, se empleó un diseño no experimental y un muestreo probabilístico para el tamaño de muestra y un no probabilístico como técnica de selección, lo cual permitió garantizar la validez y confiabilidad de los datos obtenidos de los ingenieros colegiados del CIP de Tacna. Este enfoque permitió realizar análisis estadísticos rigurosos y contrastar de manera precisa las hipótesis planteadas.

En el ámbito práctico, el estudio ofreció indicadores relevantes para optimizar la calidad del software mediante una gestión de riesgos efectiva, reduciendo costos, mejorando procesos y fortaleciendo la satisfacción del usuario. Asimismo, proporcionó insumos valiosos para la formación continua de los profesionales del sector.

A nivel social, la investigación contribuyó a promover el desarrollo de aplicaciones tecnológicas más seguras y confiables, especialmente en sectores sensibles como la salud, la educación y la banca. Este aporte favoreció la inclusión digital y elevó los estándares de calidad, beneficiando a la comunidad y fortaleciendo una cultura profesional orientada a la excelencia y la ética en la ingeniería de software.

1.4. Objetivos de la investigación

1.4.1. Objetivo general

Determinar cómo se relaciona de la gestión del riesgo con la calidad del software desarrollado por profesionales del CIP Tacna.

1.4.2. Objetivos específicos

Analizar cómo se relaciona la gestión del riesgo con la funcionalidad del software desarrollado por profesionales del CIP Tacna.

Establecer cómo se relaciona la gestión del riesgo con la fiabilidad del software desarrollado por profesionales del CIP Tacna.

Señalar cómo se relaciona la gestión del riesgo con el rendimiento del software desarrollado por profesionales del CIP Tacna.

Especificar cómo se relaciona la gestión del riesgo con la mantenibilidad del software desarrollado por profesionales del CIP Tacna.

CAPÍTULO II: MARCO TEÓRICO

2.1. Antecedentes del problema

2.1.1. Antecedentes internacionales

Owoc & Stambulski (2025) en el artículo de investigación "Software quality management: Machine learning for recommendation of regression test suites", introducen una tendencia de vanguardia en la gestión de la calidad: la aplicación de Machine Learning (ML). El estudio tiene como objetivo demostrar cómo las técnicas de aprendizaje automático pueden mejorar la gestión de la calidad del desarrollo de software, centrándose específicamente en la optimización de las suites de pruebas de regresión. Esta investigación es crucial como antecedente internacional, ya que evidencia un cambio en el enfoque de la calidad del software, pasando de procesos manuales a la utilización de Inteligencia Artificial para automatizar la selección y priorización de pruebas, lo cual impacta directamente en la eficiencia, en la reducción de costos y en el aseguramiento de la calidad de los productos de software de manera inteligente.

Afshari & Javdani Gandomani (2022) en "A novel risk management model in the Scrum and extreme programming hybrid methodology" aborda una problemática crítica en el desarrollo ágil de software: la limitada atención al riesgo dentro de metodologías como Scrum y Extreme Programming (XP), debido a su énfasis en la documentación reducida. Para mitigar esta deficiencia, la investigación propuso un novedoso modelo de gestión de riesgos diseñado específicamente para una metodología híbrida que combina Scrum y XP. Mediante la aplicación de este modelo en un estudio de caso, los autores demostraron su éxito en la gestión de riesgos y su capacidad para mejorar el rendimiento del equipo, destacando así la

necesidad y la eficacia de integrar un marco de gestión de riesgos robusto en entornos de desarrollo ágil y sus combinaciones híbridas.

Ramírez (2022), "Proceso de Gestión de Riesgos para el Desarrollo de Aplicaciones Informáticas en la UCI". El objetivo fue diseñar un proceso de gestión de riesgos para optimizar la capacidad en proyectos de software. Se aplicaron herramientas como SCAMPI y el modelo CMMI para evaluar el estado de los proyectos CICPC, CPNB, SIGEPOL, y AuditBD antes y después de implementar el proceso. Los resultados mostraron que la implantación del proceso mejoró el cumplimiento de las metas generales (GG) y metas específicas (SP) del CMMI en un alto porcentaje, permitiendo a los equipos tomar decisiones más acertadas y mejorar la mitigación de riesgos. Además, se identificaron áreas de mejora, como la falta de capacitación en la plantilla, aunque se lograron mejoras sustanciales en los productos de trabajo generados. Se concluyó que la implementación del proceso elevó la capacidad de gestión de riesgos en la UCI al nivel 5 del CMMI.

Piñero et al. (2021), "Buenas prácticas para prevenir los riesgos de la eficiencia del desempeño en productos de software". El objetivo fue identificar las características de la eficiencia del desempeño en software, centrándose en cómo prevenir los riesgos asociados. La metodología incluyó encuestas a 37 especialistas de varias entidades desarrolladoras de software en Cuba. Los resultados mostraron que el 95% de los encuestados consideraron la eficiencia del desempeño como relevante, y que las buenas prácticas propuestas mejoraron la prevención de riesgos y la satisfacción del usuario. Además, se resaltó la importancia de incorporar dichas prácticas desde el inicio del desarrollo para mitigar riesgos. Los expertos coincidieron en que estas prácticas deben integrarse desde las primeras fases del ciclo de vida del software. La conclusión fue que la gestión de riesgos es fundamental para asegurar la calidad del software y evitar insatisfacciones del usuario final.

Masso et al. (2020) en "Risk management in the software life cycle: A systematic literature review", buscó establecer el estado del arte y las brechas en el

campo de la gestión de riesgos en el ciclo de vida del software. Los autores concluyeron que el foco de la investigación ha cambiado de la definición de un proceso de gestión de riesgos integrado a concentrarse en actividades específicas de este proceso. Además, la RSL señaló una falta de rigor científico en los procesos de validación de los estudios y una deficiencia en el uso de estándares o modelos de facto para definir prácticas de gestión de riesgos. Un hallazgo crucial es la identificación de una brecha significativa con respecto al manejo del riesgo en los enfoques ágiles, lo que subraya una necesidad vital de investigar y proponer nuevas estrategias que integren y soporten la gestión de riesgos de manera coherente en metodologías ágiles como Scrum o XP.

2.1.2. Antecedentes nacionales

Betetta Soler & Cordova Matos (2025) en su investigación “Aplicación de ISTQB para la gestión de aseguramiento de calidad en una empresa de desarrollo de software, Lima 2024” demostraron la efectividad de implementar un marco formal de pruebas (basado en ISTQB) para el aseguramiento de la calidad (QA) en empresas peruanas. El principal aporte de este estudio a la presente investigación radica en que una gestión de pruebas rigurosa y documentada se traduce directamente en una mejora de la calidad del producto de software, específicamente en la identificación y corrección temprana de defectos, lo que a su vez mitiga los riesgos técnicos y financieros asociados al mantenimiento y la fiabilidad del software en producción.

Sernaque (2022), "Modelo para la gestión de riesgos de desarrollo de software bajo la perspectiva de la gestión de proyectos". El objetivo fue desarrollar un modelo de gestión de riesgos en proyectos de software utilizando la metodología Magerit y las normas ISO/IEC 27000. La investigación fue aplicada, descriptiva, propositiva y no experimental, y se utilizó un enfoque cualitativo para la construcción del modelo. Los resultados mostraron que el modelo propuesto permite identificar y gestionar eficazmente los riesgos en el desarrollo de software, mejorando la toma de decisiones y reduciendo los costos asociados a los riesgos. El modelo fue validado por expertos con un nivel de suficiencia y coherencia alto, lo

que indica que es viable para su implementación en la práctica. La conclusión es que el modelo proporciona un marco integral para la gestión de riesgos, ayudando a las empresas de software a minimizar los problemas derivados de una inadecuada gestión de riesgos.

Valverde Flores (2022) aunque en su investigación “Implementación de una gestión de riesgos de TI para mejorar la seguridad de la información de una empresa de agencia publicitaria - 2021” el estudio se centra en la seguridad, su principal contribución radica en la formalización del proceso de gestión de riesgos, una dimensión clave en la presente investigación. Valverde Flores demostró que la aplicación de metodologías rigurosas (como las basadas en las normas ISO) permite identificar, analizar y tratar las amenazas, lo cual es un fundamento directo para la efectividad de los procesos de gestión de riesgos. Además, el estudio implícitamente subraya la necesidad de asignar recursos (tanto humanos como financieros) para la implementación de las salvaguardas y planes de contingencia propuestos, lo que refuerza la relevancia de la dimensión de Gestión de Recursos Financieros en la mitigación de riesgos de TI en el contexto peruano.

Paucar et al. (2021), "Relación de la gestión de riesgos y calidad de software realizados por los profesionales del Colegio de Ingenieros del Perú del Consejo Departamental de Lima". El objetivo fue determinar la relación entre la gestión de riesgos y la calidad del software en proyectos de TI. Se utilizó un diseño correlacional no experimental con una encuesta virtual a 120 profesionales del CIP. Los resultados muestran una correlación significativa (Rho = 0,634) entre la gestión de riesgos y la calidad del software, destacando que los procesos de gestión de riesgos (Rho = 0,656) y la planificación financiera (Rho = 0,651) influyen significativamente en la calidad. Solo el 9 % de los proyectos alcanzaron todos los objetivos de calidad. La conclusión es que una gestión adecuada de riesgos mejora la calidad del software, pero los proyectos evaluados mostraron deficiencias en la aplicación de buenas prácticas.

2.2. Bases teóricas

2.2.1. Gestión de riesgos

2.2.1.1. Definición según autores

El análisis y la administración del riesgo son acciones que ayudan al equipo de software a entender y manejar la incertidumbre. Muchos problemas pueden plagar un proyecto de software. Un riesgo es un problema potencial: puede ocurrir, puede no ocurrir. Pero, sin importar el resultado, realmente es una buena idea identificarlo, valorar su probabilidad de ocurrencia, estimar su impacto y establecer un plan de contingencia para el caso de que el problema realmente ocurra. Reconocer qué puede salir mal es el primer paso, llamado "identificación de riesgos". A continuación, cada riesgo se analiza para determinar la probabilidad de que ocurra y el daño que causará si ocurre. Una vez establecida esta información se clasifican los riesgos, por probabilidad e impacto. Finalmente, se desarrolla un plan para manejar aquellos que tengan alta probabilidad y alto impacto. Se produce un plan para mitigar, monitorear y manejar el riesgo (MMMR) o un conjunto de hojas de información de riesgo (Pressman, 2010, pág. 640)

La gestión de riesgos es definida por Hillson como un proceso sistemático cuyo propósito es identificar, analizar y mitigar las amenazas que podrían comprometer el logro de los objetivos organizacionales. Este enfoque trasciende la simple reacción ante problemas, pues implica una planificación estratégica y anticipada frente a escenarios de incertidumbre (Hillson, 2020, pág. 5). Hillson subraya que los riesgos no deben concebirse únicamente como factores negativos a evitar, sino como catalizadores de innovación y mejora continua. Bajo este paradigma, los riesgos ofrecen la oportunidad de rediseñar procesos, optimizar recursos y generar ventajas competitivas. De esta manera, la gestión adecuada de los riesgos no solo minimiza impactos indeseados, sino que puede transformar amenazas potenciales en oportunidades de crecimiento y diferenciación en mercados cada vez más exigentes (Hillson, 2020, pág. 9).

Kerzner aporta una visión centrada en el ámbito de la gestión de proyectos, enfatizando la relevancia de la gestión de riesgos en contextos altamente dinámicos

como el desarrollo de software. Por tanto, la gestión de riesgos constituye una herramienta indispensable para anticipar escenarios adversos y optimizar la asignación de recursos. Al aplicar una evaluación continua de los riesgos, las organizaciones pueden adelantarse a la ocurrencia de problemas críticos, evitando que estos afecten el avance del proyecto o comprometan sus entregables. La anticipación no solo permite reducir pérdidas, sino que también mejora la capacidad de respuesta frente a imprevistos (Kerzner, 2021, pág. 604).

Por su parte, Crouhy et al. destacan que la gestión de riesgos debe concebirse como un proceso integral y dinámico, que no se limita únicamente a la identificación y evaluación de amenazas, por tanto, es imprescindible que las organizaciones implementen controles efectivos y establezcan mecanismos de monitoreo continuo, con el fin de verificar que las medidas preventivas adoptadas sean realmente eficaces en el tiempo. Además, recalcan que este proceso debe ser lo suficientemente flexible para adaptarse a los cambios constantes del entorno externo e interno, ya sean tecnológicos, económicos o regulatorios. Desde esta perspectiva, la gestión de riesgos debe abarcar no solo los riesgos operativos, sino también los financieros y estratégicos, consolidando un marco integral que apoye la sostenibilidad organizacional y la resiliencia a largo plazo (Crouhy, Galai, & Mark, 2021, pág. 4).

2.2.1.2. Importancia de la gestión de riesgos en ingeniería de software

La gestión de riesgos es fundamental para el éxito de los proyectos de software modernos. Numerosos estudios recientes confirman que una adecuada gestión de riesgos aumenta significativamente las probabilidades de lograr los objetivos del proyecto (Nikolaenko & Sidorov, 2023). La ausencia o deficiencia en este proceso se ha identificado como una de las causas de fracaso en proyectos de desarrollo, especialmente bajo presiones de presupuesto y tiempo (Luthfiansyah, Prasetyo, & Raharjo, 2024).

Por ejemplo, un análisis de Standish Group en 50,000 proyectos TI indicó que cada riesgo negativo materializado ocasionó en promedio pérdidas de $1000,

pérdidas que podrían haberse evitado invirtiendo menos de $1 en medidas preventivas. Estas evidencias destacan que anticipar y manejar riesgos no es un "lujo" sino una necesidad para evitar retrasos costosos, sobrecostos y fallos catastróficos en proyectos de software (Nikolaenko & Sidorov, 2023).

Una gestión de riesgos eficaz aporta múltiples beneficios a la ingeniería de software, en primer lugar, brinda estabilidad al proyecto al anticipar problemas potenciales y mitigar su impacto antes de que afecten el cronograma, el presupuesto o el rendimiento del producto. En segundo lugar, mejora la toma de decisiones al proporcionar un marco estructurado para evaluar opciones: los gerentes pueden sopesar cursos de acción considerando los riesgos asociados a cada uno. Además, un enfoque proactivo en riesgos incrementa la confianza de los interesados (stakeholders) al demostrar compromiso con la calidad y la confiabilidad del proyecto. Esto fomenta una cultura de transparencia donde se abordan abiertamente las incertidumbres, resultando en productos finales de mayor calidad debido a la reducción de fallos post-despliegue (Tasnim, 2024).

Actualmente existe amplio consenso en la industria sobre la importancia de gestionar riesgos desde el inicio de los proyectos, en estudios recientes, la gran mayoría de profesionales encuestados reconoce que la gestión de riesgos es crítica en sus proyectos de software; incluso en entornos ágiles –donde tradicionalmente se hacía menos énfasis formal en la gestión de riesgos– se ha visto necesaria la adopción de prácticas de riesgo para prevenir problemas en ciclos de desarrollo rápidos. La falta de atención a los riesgos en metodologías ágiles puede conducir a fallos, por lo que hoy se desarrollan herramientas específicas para incorporar la gestión de riesgos en Agile (Gontijo, Keil, Sanches, & Dinis, 2020).

En esencia, abordar activamente los riesgos salvaguarda el cumplimiento de plazos y objetivos del proyecto y evita sorpresas desagradables, convirtiéndose en un factor clave para el éxito sostenido en ingeniería de software (Nikolaenko & Sidorov, 2023).

2.2.1.3. Tipos de riesgos en proyectos de software

Los proyectos de desarrollo de software enfrentan diversos tipos de riesgos, que pueden clasificarse según distintas taxonomías reconocidas internacionalmente. La guía PMBOK del Project Management Institute señala cuatro categorías amplias de riesgo: técnicos, externos, organizacionales y de gestión del proyecto (Chernish & Yemets, 2024).

Por su parte, estudios académicos recientes proponen clasificaciones más detalladas: por ejemplo, Nikolaenko et al. (2023) identificaron hasta 105 riesgos "universales" en proyectos TI, desglosándolos en 5 riesgos comerciales, 45 riesgos de cumplimiento normativo (regulatorios) y 55 riesgos propios del proyecto. Estas perspectivas reflejan que los riesgos abarcan desde aspectos tecnológicos internos hasta factores externos del entorno de negocio y regulación. En cualquier caso, es esencial categorizar los riesgos para asegurar que se aborden todas las áreas vulnerables de un proyecto.

Dentro de los riesgos técnicos se incluyen aquellos inherentes a la tecnología, al proceso de desarrollo y a la calidad del producto software. Estos riesgos abarcan desafíos como la complejidad del software, posibles defectos de programación, fallas de integración con terceros, problemas de rendimiento, seguridad y más (Tasnim, 2024).

Por ejemplo, una arquitectura inadecuada o tecnologías mal escogidas pueden resultar en dificultades de escalabilidad o vulnerabilidades de seguridad, asimismo, la falta de pruebas rigurosas puede dejar errores no detectados que afecten la calidad final (Chernish & Yemets, 2024). Este tipo de riesgos técnicos, si no se gestionan, amenazan directamente la funcionalidad y confiabilidad del software. Mitigarlos requiere atención experta durante todo el ciclo de desarrollo – desde un buen análisis de requisitos técnicos, pasando por design reviews, hasta pruebas continuas– para garantizar que la solución final cumpla con los estándares esperados de desempeño y seguridad.

Por otro lado, los riesgos de gestión de proyecto y organizacionales involucran factores humanos, de planificación y de entorno empresarial. En esta categoría figuran riesgos de cronograma (por ejemplo, subestimar el tiempo de desarrollo y sufrir retrasos), riesgos de presupuesto (sobrecostos inesperados), y riesgos asociados al alcance y requerimientos (cambios frecuentes o mal definiciones que desestabilizan el proyecto) (Tasnim, 2024).

También se consideran riesgos relativos al equipo de trabajo –como rotación de personal clave o falta de habilidades– y a la comunicación y gestión de interesados (stakeholders), adicionalmente, existen riesgos externos o de negocio, como cambios en el mercado, nuevas regulaciones legales o fallos de proveedores externos, que pueden impactar al proyecto; un estudio resaltó que prácticamente todos los proyectos, independientemente de su tamaño o duración, están expuestos a riesgos regulatorios (compliance) y comerciales además de los técnicos y de gestión (Nikolaenko & Sidorov, 2023).

Por ejemplo, incumplir con una norma de seguridad podría implicar sanciones significativas (riesgo de cumplimiento), o un cambio abrupto en las preferencias de los usuarios podría dejar obsoleto un producto en desarrollo (riesgo de mercado), reconocer estos tipos de riesgo más allá de lo técnico permite a los gestores tomar acciones preventivas más allá del código –como asegurar respaldo de la alta dirección, ajustar contratos con clientes/proveedores, o establecer reservas de contingencia en tiempo y presupuesto– para afrontar imprevistos del contexto organizacional y del entorno.

2.2.1.4. Ciclo de gestión de riesgos

La gestión de riesgos se lleva a cabo mediante un ciclo de procesos iterativos que aseguran la identificación y control continuo de las amenazas durante todo el proyecto. En metodologías modernas, este ciclo de gestión de riesgos suele desglosarse en cinco etapas fundamentales: planificación, identificación, análisis/evaluación, planificación de respuestas y monitorización/control (Chernish & Yemets, 2024).

En primer lugar, se planifica cómo se gestionarán los riesgos (definiendo metodologías, responsabilidades y herramientas), a continuación, se identifican los riesgos potenciales; luego se analizan para evaluar su severidad. Con base en ese análisis, se planifican las respuestas o mitigaciones apropiadas para los riesgos priorizados. Finalmente, se monitorean los riesgos a lo largo del tiempo, revaluándolos y ajustando las estrategias según sea necesario. Este proceso no es lineal de una sola vez, sino cíclico: se repite regularmente durante el ciclo de vida del proyecto, incorporando nuevos riesgos que surjan y verificando el estado de los riesgos conocidos, de esta manera, la gestión de riesgos se integra de forma continua en la gestión del proyecto, en lugar de ser una actividad aislada o puntual (Davydov, 2024).

En la fase de identificación de riesgos, el equipo recopila todos los riesgos posibles que puedan afectar el proyecto. Se revisan documentos, lecciones aprendidas y se consulta a expertos y miembros del equipo para elaborar una lista exhaustiva de riesgos potenciales (Brown, 2025). Es útil preguntarse: "¿Qué podría salir mal?" en cada aspecto del proyecto (requerimientos, tecnología, personal, proveedores, etc.). Por ejemplo, se exploran vulnerabilidades técnicas (¿fallos en la arquitectura o seguridad?), incertidumbres en requisitos (¿cambios del cliente?), y factores externos (¿nuevas normativas o dependencias de terceros?) (Tasnim, 2024).

Tras identificar los riesgos, sigue la evaluación o análisis de cada riesgo, en esta etapa, para cada riesgo se estima la probabilidad de que ocurra y el impacto que tendría sobre los objetivos del proyecto. Estas dos dimensiones (probabilidad e impacto) permiten calcular la criticidad o nivel de riesgo, frecuentemente se emplea una matriz de probabilidad-impacto u otras herramientas de puntaje para visualizar y priorizar los riesgos identificados: aquellos con alta probabilidad y/o alto impacto se clasifican como los más críticos (Brown, 2025). El propósito del análisis es enfocar la atención y recursos en los riesgos más significativos, con recursos limitados, es esencial "separar la señal del ruido", atendiendo primero los riesgos cuyo efecto potencial justifica acciones inmediatas (Tasnim, 2024).

Tras el análisis, el ciclo continúa con la planificación e implementación de las respuestas a los riesgos priorizados, en esta fase se definen las estrategias específicas para cada riesgo clave, con el fin de reducir su probabilidad de ocurrencia o su impacto en caso de materializarse. Las estrategias típicas de respuesta incluyen: evitar el riesgo (eliminar completamente la causa o condición de riesgo, por ejemplo descartando un requisito de alta incertidumbre), mitigar el riesgo (tomar acciones para disminuir su probabilidad o impacto, p. ej. prototipar una tecnología nueva para reducir incertidumbre), transferir el riesgo (asignar el riesgo a un tercero, como contratar un seguro o subcontratar una parte riesgosa del proyecto), y aceptar el riesgo (reconocer que el riesgo existe y prepararse para afrontarlo si ocurre, por ejemplo, estableciendo un plan de contingencia). Para cada riesgo significativo se debe documentar un plan de acción: qué se hará, quién es responsable y en qué momento o condición se ejecutará la respuesta (Tasnim, 2024).

Finalmente, se procede a la monitorización y control de los riesgos a lo largo del proyecto. Esto implica un seguimiento periódico del estado de cada riesgo identificado y la eficacia de las acciones emprendidas. En reuniones de proyecto se revisan los riesgos abiertos, se verifican indicadores (por ejemplo, señales tempranas de que un riesgo está aumentando) y se actualiza el plan de riesgos según sea necesario (Tasnim, 2024).

La gestión de riesgos es, por tanto, un proceso iterativo y dinámico: nuevos riesgos pueden surgir en cualquier fase y los riesgos existentes pueden cambiar en probabilidad o impacto, requiriendo re-evaluación continua, al integrar este ciclo en todas las fases del proyecto (desde análisis hasta despliegue), los equipos pueden abordar oportunamente los riesgos emergentes y mantener el control, en lugar de reaccionar tarde ante sorpresas (Davydov, 2024).

2.2.1.5. Herramientas y técnicas de gestión de riesgos

Existen numerosas herramientas y técnicas que apoyan cada etapa del proceso de gestión de riesgos en proyectos de software, para la identificación de

riesgos, una de las técnicas más utilizadas es la lluvia de ideas (brainstorming) con el equipo de proyecto y stakeholders relevantes (Luthfiansyah, Prasetyo, & Raharjo, 2024).

En sesiones estructuradas de brainstorming, los participantes listan colectivamente todo tipo de riesgos imaginables, fomentando una visión amplia de las posibles amenazas. Otra herramienta común son las listas de verificación basadas en experiencias previas (checklists de riesgos típicos), que ayudan a no pasar por alto riesgos conocidos en proyectos similares. Adicionalmente, técnicas analíticas como el análisis SWOT (Fortalezas, Debilidades, Oportunidades, Amenazas) permiten descubrir riesgos internos (debilidades) y externos (amenazas) del proyecto al examinar su contexto estratégico (Brown, 2025).

Cabe mencionar también el análisis de causa raíz (Root Cause Analysis), que aunque se emplea tradicionalmente para investigar problemas ocurridos, también puede adaptarse a la identificación proactiva de riesgos latentes al examinar las posibles causas fundamentales de futuros fallos, estas herramientas de identificación, combinadas, garantizan un catálogo más completo de riesgos iniciales. En cuanto al análisis y evaluación de riesgos, existen herramientas cualitativas y cuantitativas para profundizar en los riesgos identificados. Una técnica básica es la construcción de la matriz de probabilidad e impacto, donde cada riesgo se coloca en una cuadrícula según qué tan probable es y qué tan grave sería su impacto, priorizando visualmente los riesgos más críticos (Brown, 2025).

Para riesgos complejos, se pueden usar técnicas como el Análisis de Árbol de Decisión y la valoración de escenarios, que ayudan a estimar resultados esperados considerando diferentes desenlaces de riesgo, asimismo, métodos cuantitativos como la simulación Monte Carlo son populares para modelar incertidumbres en estimaciones de proyecto (cronograma o costo): al simular miles de escenarios con distribuciones de probabilidad, Monte Carlo ofrece una distribución probabilística del resultado del proyecto, identificando la probabilidad de cumplir plazos o presupuestos bajo ciertas condiciones de riesgo.

Otra técnica es el Análisis FMEA (Failure Mode and Effects Analysis), procedente de la ingeniería de calidad, que puntúa riesgos en base a su severidad, ocurrencia y detectabilidad para priorizar acciones. Recientemente, han emergido enfoques de inteligencia artificial para el análisis de riesgos: por ejemplo, el uso de redes neuronales y algoritmos de aprendizaje automático para predecir riesgos de proyecto a partir de datos históricos (Luthfiansyah, Prasetyo, & Raharjo, 2024).

Para la planificación de respuestas y el seguimiento de los riesgos, también se dispone de herramientas especializadas, el registro de riesgos (risk register) es fundamental: se trata de un documento o sistema donde se documentan todos los riesgos identificados, sus evaluaciones y las acciones de respuesta planificadas. Muchas organizaciones emplean software de gestión de proyectos o módulos específicos de software de gestión de riesgos para mantener este registro actualizado y colaborativo, dichas herramientas permiten asignar responsables a cada riesgo, establecer fechas de revisión y enviar alertas automáticas si un riesgo se materializa o si alguna respuesta debe ejecutarse (Jackowski, 2023).

Otra técnica es la “reunión de revisión de riesgos” periódica, en la cual el equipo discute el estado de los riesgos principales y la eficacia de las mitigaciones en curso – esta práctica de reportes periódicos de riesgo asegura que la alta dirección y todos los involucrados estén informados sobre la exposición actual del proyecto. Además, se preparan planes de contingencia para ciertos riesgos: por ejemplo, definir acciones alternativas en caso de que un riesgo alto ocurra (como tener proveedores de respaldo si falla el principal). La literatura reciente enfatiza la importancia de la planificación proactiva y la vigilancia continua: herramientas como los indicadores de riesgo clave (KRIs) monitorean métricas del proyecto que pueden servir de alerta temprana (por ejemplo, tasa de crecimiento de errores detectados por semana podría ser indicador de riesgo de calidad) (Luthfiansyah, Prasetyo, & Raharjo, 2024).

En metodologías ágiles, se han propuesto incluso gráficas de “burndown” de riesgo, similares a las de tareas, para visualizar cómo disminuye (o aumenta) la

exposición al riesgo a lo largo de las iteraciones, por tanto, una combinación de técnicas tradicionales (brainstorming, matrices, FMEA) con herramientas modernas (software colaborativo, dashboards de riesgo, AI para predicción) proporciona un arsenal completo para gestionar los riesgos de proyectos de software de manera efectiva (Luthfiansyah, Prasetyo, & Raharjo, 2024). La selección de herramientas adecuadas depende del contexto del proyecto, pero todas comparten el objetivo de identificar, analizar, monitorear y comunicar los riesgos oportunamente para que el equipo pueda tomar decisiones informadas y mantener el proyecto bajo control pese a la incertidumbre.

2.2.1.6. Modelos y estándares aplicables

En el ámbito de la gestión de riesgos en ingeniería de software, existen diversos modelos y estándares internacionales que ofrecen marcos de referencia validados para implementar buenas prácticas. Uno de los más influyentes es la norma ISO 31000:2018, que proporciona directrices generales para la gestión del riesgo en cualquier tipo de organización. Este estándar enfatiza principios como la integración de la gestión de riesgos en todos los procesos organizativos, la naturaleza iterativa y colaborativa del proceso, y la necesidad de adaptar el enfoque al contexto y cultura de cada organización (International Organization for Standardization, 2018).

ISO 31000 establece un marco holístico compuesto de principios, una estructura (framework) y un proceso de gestión de riesgos, promoviendo un enfoque proactivo y sistemático. Aunque no está orientado exclusivamente a proyectos de software, sus recomendaciones son plenamente aplicables a ellos, fomentando por ejemplo la responsabilidad compartida en la gestión de riesgos (todos en la organización tienen algún rol en identificar o gestionar riesgos) y la comunicación transparente de los riesgos a las partes interesadas (International Organization for Standardization, 2018).

Específicamente para ingeniería de software y proyectos de TI, destaca el estándar ISO/IEC/IEEE 16085:2020. Esta norma internacional, publicada

conjuntamente por ISO e IEEE, provee un marco de gestión de riesgos adaptado al ciclo de vida de sistemas y software. El estándar 16085:2020 complementa las normas de procesos de ciclo de vida de software (como ISO/IEC/IEEE 12207) y sigue la línea de ISO 31000 pero aterrizada en proyectos de desarrollo tecnológico (IEEE, 2021).

Entre sus aportes, define una terminología común y orientaciones específicas para llevar a cabo la gestión de riesgos en proyectos de software, asegurando que los riesgos se manejen de manera consistente a lo largo de todas las fases del proyecto. También especifica los "deliverables" o artefactos mínimos de información que deben generarse al implementar el proceso de gestión de riesgos (por ejemplo, registros de riesgos, planes de riesgo, informes), sirviendo como base para evaluar la conformidad de la gestión de riesgos en un proyecto (IEEE, 2021).

Un aspecto importante es que ISO 16085:2020 está pensado para ser universalmente aplicable a cualquier tipo de proyecto de software, sin importar industria o tamaño, proporcionando un enfoque unificado para integrar la multitud de técnicas y herramientas de riesgo dentro de un proceso coherente (IEEE, 2021). En síntesis, este estándar se convierte en una guía práctica para ingenieros de software y jefes de proyecto sobre cómo implementar la gestión de riesgos de acuerdo con las mejores prácticas internacionales específicamente adaptadas al desarrollo de software.

Además de los estándares formales, existen modelos de gestión y madurez que incorporan la gestión de riesgos como componente esencial. Un ejemplo es el Project Management Body of Knowledge (PMBOK) del PMI, que desde ediciones anteriores ya incluye la Gestión de Riesgos del Proyecto como una de sus diez áreas de conocimiento fundamentales. PMBOK provee procesos detallados (identificación, análisis cualitativo, análisis cuantitativo, planificación de respuestas, implementación de respuestas y monitoreo de riesgos) que han orientado a profesionales en todo el mundo, aunque su última edición (2021) ha

migrado a principios más flexibles manteniendo el énfasis en la anticipación de riesgos.

Por otro lado, el Capability Maturity Model Integration (CMMI) en su versión más reciente V2.0 posiciona explícitamente la Gestión de Riesgos como una de las prácticas cruciales para mejorar el desempeño de las organizaciones, de acuerdo con la documentación de CMMI V2.0, la gestión de riesgos ayuda a las organizaciones a identificar, evaluar y mitigar riesgos tanto en proyectos como en operaciones, incrementando las probabilidades de éxito de los proyectos y la resiliencia frente a imprevistos (Jedrzejowska, 2022). Esto muestra cómo los modelos de madurez integran el manejo sistemático de riesgos como un indicador de la capacidad organizativa: las organizaciones en niveles altos de madurez demuestran procesos de riesgo robustos y repetibles.

También en metodologías específicas como PRINCE2 (de origen británico) se incluye un proceso definido de gestión de riesgos (denominado “Riesgos” como uno de sus temas principales). Por último, en el campo de la seguridad de la información –estrechamente relacionado con software– existen marcos como el NIST Risk Management Framework (EEUU) que, aunque enfocado en riesgos de seguridad, establece prácticas transferibles a la gestión de riesgos en desarrollo software crítico (McGrath & Whitty, 2020).

Finalmente, tanto los estándares internacionales (ISO/IEEE) como los marcos de mejores prácticas (PMI, CMMI, PRINCE2, entre otros) ofrecen modelos aplicables que guían a los ingenieros de software en la implementación de una gestión de riesgos estructurada, seguir estas referencias comprobadas asegura que la organización no dependa solo de la intuición, sino que aplique principios y procesos probados para navegar las incertidumbres de los proyectos de software con éxito.

2.2.1.7. Dimensiones

En la arquitectura conceptual de Pressman, la gestión de riesgos no es una actividad pesimista, sino un mecanismo de defensa proactivo que asegura la

viabilidad del proyecto. El autor establece una dicotomía clara entre la gestión reactiva (el "modo bombero", donde el equipo reacciona a los problemas a medida que ocurren, consumiendo recursos de manera ineficiente) y la gestión proactiva (donde los riesgos se identifican y mitigan antes de convertirse en problemas). Desglosaremos la gestión de riesgos en tres dimensiones operativas: los procesos de gestión de riesgos, la gestión de recursos financieros (como fuente y mitigación de riesgo) y la gestión del talento humano.

A) Procesos de gestión de riesgos

Cuando se considera el riesgo en el contexto de la ingeniería del software, los tres fundamentos conceptuales de Charette siempre están presentes. El futuro es su preocupación: ¿qué riesgos pueden hacer que el proyecto de software salga defectuoso? El cambio es lo que preocupa: ¿cómo afectan en los cronogramas y en el éxito global los cambios que puede haber en los requisitos del cliente, en las tecnologías de desarrollo, en los entornos meta y en todas las otras entidades conectadas con el proyecto? Por último, se debe lidiar con las opciones: ¿qué métodos y herramientas deben usarse, ¿cuántas personas deben involucrarse, cuánto énfasis es “suficiente” poner en la calidad? (Pressman, 2010, pág. 640)

El proceso de la administración/gestión de riesgos debiera ser una parte integral de la administración/gestión y de la toma de decisiones y se debiera integrar en la estructura, las operaciones y los procesos de la organización. Puede aplicarse a nivel estratégico, operacional, de programas o de proyectos. Puede haber muchas aplicaciones del proceso de la administración/gestión de riesgos dentro de las organizaciones, adaptadas para lograr objetivos, y apropiadas a los contextos interno y externo en los cuales se aplican. A lo largo del proceso de la administración/gestión de riesgos se debiera considerar la naturaleza dinámica y variable del comportamiento humano y la cultura. Aunque el proceso de administración/gestión de riesgos se presenta frecuentemente como secuencial, en la práctica es iterativo. (International Organization for Standardization, 2018, pág. 10).

B) Gestión de recursos financieros

La administración de los proyectos de software comienza con un conjunto de actividades que de manera colectiva se llaman planificación de proyecto. Antes de que el proyecto pueda comenzar, el equipo de software debe estimar el trabajo que se va a realizar, los recursos que se requerirán y el tiempo que transcurrirá de principio a fin. Muchos trabajadores técnicos preferirían realizar trabajo técnico en lugar de pasar tiempo planificando. Muchos gerentes técnicos no tienen suficiente capacitación en administración técnica como para sentirse seguros de que su planificación mejorará el resultado de un proyecto. Dado que ninguna parte quiere hacer planes, con frecuencia nunca se hacen. Pero fallar en la planificación es uno de los errores más cruciales que un proyecto puede tener... la planificación efectiva es necesaria para resolver problemas corriente arriba [tempranamente en el proyecto] a bajo costo, en lugar de corriente abajo [tardíamente en el proyecto] a alto costo. El proyecto promedio emplea 80 por ciento de su tiempo en "poner al día": corregir errores que se cometieron anteriormente en el proyecto. (Pressman, 2010, pág. 593)

La alta dirección y los órganos de supervisión, cuando sea aplicable, debieran asegurar la asignación de los recursos apropiados para la administración/gestión de riesgos, que puede incluir, pero no limitarse a: las personas, las habilidades, la experiencia y las competencias; los procesos, los métodos y las herramientas de la organización a utilizar para administrar/gestionar los riesgos; los procesos y procedimientos documentados; los sistemas de administración/gestión de la información y del conocimiento; el desarrollo profesional y las necesidades de formación. Las organizaciones debieran considerar las competencias y limitaciones de sus recursos existentes. (International Organization for Standardization, 2018, pág. 8)

C) Gestión de talento humano

El "factor humano" es tan importante que el Software Engineering Institute desarrolló un Modelo de madurez de capacidades del personal (People-CMM, por

sus siglas en inglés), en reconocimiento al hecho de que “toda organización requiere mejorar continuamente su habilidad para atraer, desarrollar, motivar, organizar y conservar la fuerza de trabajo necesaria a fin de lograr sus objetivos empresariales estratégicos”. El People-CMM define las siguientes áreas prácticas clave para el personal de software: plantilla, comunicación y coordinación, ambiente de trabajo, desempeño administrativo, capacitación, compensación, análisis y desarrollo de competencias, desarrollo profesional, desarrollo de grupo de trabajo y desarrollo de equipo/cultura, entre otros. Las organizaciones que conforme a este modelo logran altos niveles de madurez de capacidades de personal tienen una probabilidad muy elevada de alcanzar la implementación de prácticas administrativas efectivas en los proyectos de software. (Pressman, 2010, pág. 555)

El comportamiento humano y la cultura influyen considerablemente en todos los aspectos de la administración/gestión de riesgos en todos los niveles y etapas. (International Organization for Standardization, 2018, pág. 4)

2.2.1.8. Indicadores

A) Efectividad de los procesos de gestión de riesgos:

El Software Engineering Institute (SEI) identifica siete principios que “ofrecen un marco conceptual para lograr una administración de riesgo efectiva”. Éstos son: Mantener una perspectiva global, tomar una visión de previsión, alentar la comunicación abierta, integrar, enfatizar un proceso continuo, desarrollar una visión de producto compartida y alentar el trabajo en equipo. (Pressman, 2010, pág. 642)

B) Adecuación de la gestión financiera en la mitigación de riesgos:

La planificación requiere adoptar un compromiso inicial, aun cuando es probable que este “compromiso” resulte erróneo. Siempre que se hacen estimaciones se mira hacia el futuro y se acepta cierto grado de incertidumbre habitual. (Pressman, 2010, pág. 594)

C) Percepción de la capacidad del equipo en la gestión de riesgos:

El personal debe organizarse en equipos eficaces, motivados para hacer trabajo de software de alta calidad, y coordinarse para lograr comunicación efectiva. El elemento esencial en todos los proyectos de software es el personal. Los ingenieros del software pueden organizarse en diferentes estructuras de equipo que van desde las jerarquías tradicionales de control hasta los equipos de "paradigma abierto". (Pressman, 2010, pág. 568)

2.2.2. Calidad de software

2.2.2.1. Definición según autores

Incluso los desarrolladores de software más experimentados estarán de acuerdo en que obtener software de alta calidad es una meta importante. Pero, ¿cómo se define la calidad del software? En el sentido más general se define como: *Proceso eficaz de software que se aplica de manera que crea un producto útil que proporciona valor medible a quienes lo producen y a quienes lo utilizan*. (Pressman, 2010, pág. 340)

Por otro lado, Sommerville define la calidad del software como un proceso continuo de evaluación que no solo incluye la ausencia de errores, sino también la capacidad del software para evolucionar con el tiempo y ajustarse a nuevos requerimientos. Además, sostiene que la calidad del software es crítica para la satisfacción del cliente, ya que impacta directamente en la experiencia del usuario final (Sommerville, 2020, pág. 656).

McConnell señala que la calidad del software debe medirse tanto en términos de calidad interna como de calidad externa. La calidad interna se refiere a la estructura del código y su mantenibilidad, mientras que la calidad externa está relacionada con el rendimiento, la fiabilidad y la satisfacción del usuario. Estas dos dimensiones deben ser gestionadas en paralelo para asegurar que el software no solo funcione correctamente, sino que también sea fácil de mantener y escalar a largo plazo (McConnell, 2021, pág. 474).

Finalmente, según Boehm, la calidad del software implica la capacidad de una aplicación para operar eficientemente bajo condiciones normales y excepcionales, minimizando el riesgo de fallos. (Boehm, Software Quality Framework, 2019)

2.2.2.2. Enfoques y modelos de calidad: ISO/IEC 25000 (SQuaRE), McCall, Boehm, Dromey

Los modelos de calidad de software proporcionan marcos de atributos para evaluar la calidad de productos software, uno de los primeros enfoques fue el modelo de McCall, propuesto en 1977, que clasificó la calidad en tres perspectivas: operación del producto, revisión del producto y transición del producto, se definió 11 atributos de calidad, incluyendo corrección, fiabilidad, eficiencia, integridad, usabilidad, mantenibilidad, flexibilidad, testabilidad, portabilidad, reutilizabilidad e interoperabilidad (McCall, Richards, & Walters, 1977).

Por su parte, Boehm (1978) amplió el modelo de McCall añadiendo un enfoque cuantitativo: introdujo un conjunto jerarquizado de 19 parámetros o características orientadas a la facilidad de uso, mantenimiento y adaptación del software. Boehm destacó atributos como portabilidad, fiabilidad, eficiencia, usabilidad, testabilidad, comprensibilidad y flexibilidad. Estos primeros modelos sentaron las bases al identificar factores de calidad del producto desde diversas perspectivas (tanto del usuario como del desarrollador) (Ali, y otros, 2022).

Con el tiempo, los estándares internacionales refinaron y extendieron estos marcos, el estándar ISO/IEC 9126 estableció en los años 90 seis características de calidad (funcionalidad, fiabilidad, usabilidad, eficiencia, mantenibilidad y portabilidad) y añadió el concepto de calidad en uso (efectividad, productividad, seguridad y satisfacción del usuario) (Perdomo & Zapata, 2021).

Posteriormente, la familia ISO/IEC 25000: SQuaRE (Software Product Quality Requirements and Evaluation) evolucionó a partir de ISO 9126 para proporcionar un marco más completo, en particular, ISO/IEC 25010 (publicada en 2011, parte de SQuaRE) define ocho características de calidad del producto

software: adecuación funcional, eficiencia de desempeño, compatibilidad, usabilidad, fiabilidad, seguridad, mantenibilidad y portabilidad, esta norma es reconocida como un modelo líder para evaluación de calidad, y sirve como base para métricas internas, externas y en uso (Sifuentes & Peralta, 2022). SQuaRE organiza sus estándares en divisiones para requisitos de calidad, medición y evaluaciones, integrando la evaluación de la calidad interna, externa y en uso del software.

Entre los enfoques académicos, Dromey (1996) propuso un modelo orientado a conectar las propiedades internas del software con los atributos de calidad. Formuló organizar la calidad en tres modelos interrelacionados: modelo de implementación, de requisitos y de diseño, su modelo introduce la noción de propiedades portadoras de calidad que vinculan estructuras del software con criterios de calidad.

En el modelo de Dromey (1996), las características de calidad incluyen muchas de las definidas por ISO 9126 (funcionalidad, fiabilidad, eficiencia, usabilidad, mantenibilidad, portabilidad, etc.), agregando énfasis en la madurez de procesos y reusabilidad, este enfoque buscó solventar algunas limitaciones de los modelos previos estableciendo trazabilidad entre defectos en componentes del software y atributos de calidad, lo que ayuda a guiar acciones de mejora (Ali, y otros, 2022). Por tanto, los distintos modelos – desde McCall y Boehm hasta ISO 25000 y Dromey – han ofrecido estructuras jerárquicas de factores y métricas para entender y evaluar la calidad del software, sirviendo como referencia para estándares y prácticas de ingeniería de software modernas.

2.2.2.3. Dimensiones

A) Funcionalidad

La funcionalidad se califica de acuerdo con el conjunto de características y capacidades del programa, la generalidad de las funciones que se entregan y la seguridad general del sistema. (Pressman, 2010, pág. 188)

La funcionalidad del software constituye una dimensión esencial de la calidad, dado que está orientada a garantizar que un sistema cumpla con los requisitos funcionales establecidos en su diseño, Sommerville señala que la funcionalidad debe garantizar la correcta interacción entre los usuarios y el sistema, procesando adecuadamente entradas y salidas, así como generando resultados consistentes. Una adecuada cobertura funcional permite que el software cumpla con las expectativas del cliente y asegure su utilidad práctica en diferentes escenarios operativos. De este modo, la funcionalidad no se limita a la ejecución de tareas básicas, sino que involucra la capacidad del software para responder con precisión a contextos diversos y requisitos específicos (Sommerville, 2020, pág. 656).

Finalmente, Sommerville subraya que la funcionalidad debe evaluarse mediante pruebas sistemáticas y métricas alineadas a estándares internacionales como ISO/IEC 25010, dichas métricas permiten medir la adecuación funcional, la exactitud de resultados y la interoperabilidad con otros sistemas. Esto facilita un proceso de verificación más riguroso, que no solo comprueba el cumplimiento de requisitos, sino también la robustez y consistencia del producto en distintos contextos de uso. (Sommerville, 2020)

B) Fiabilidad

La fiabilidad se evalúa con la medición de la frecuencia y gravedad de las fallas, la exactitud de los resultados que salen, el tiempo medio para que ocurra una falla (TMPF), la capacidad de recuperación ante ésta y lo predecible del programa. (Pressman, 2010, pág. 188)

Sommerville define que la fiabilidad de un software es la capacidad del sistema para funcionar correctamente durante un período de tiempo determinado bajo condiciones específicas (Sommerville, pág. 295).

Kerzner complementa esta definición al enfatizar que la fiabilidad implica también resiliencia frente a condiciones adversas, como sobrecargas de trabajo, fallos parciales de hardware o interrupciones imprevistas. De acuerdo con su

planteamiento, un software confiable debe mantener niveles de servicio aceptables incluso en escenarios de estrés, evitando colapsos o pérdida de información. Este enfoque es especialmente relevante en sectores como la salud o las finanzas, donde la disponibilidad continua del sistema es imprescindible (Kerzner, 2021, pág. 703).

Asimismo, Pressman destaca la importancia de implementar mecanismos de tolerancia a fallos y de realizar pruebas exhaustivas para verificar el comportamiento del software en distintas condiciones. Las métricas asociadas, como la disponibilidad, el tiempo medio entre fallos (MTBF) y el tiempo medio de recuperación (MTTR), constituyen parámetros de evaluación de gran valor. Así, la fiabilidad no solo garantiza la reducción de fallos, sino que contribuye a la construcción de un producto robusto y confiable a largo plazo. (Pressman, 2010)

C) Rendimiento

El rendimiento se mide con base en la velocidad de procesamiento, el tiempo de respuesta, el uso de recursos, el conjunto y la eficiencia. (Pressman, 2010, pág. 188)

El rendimiento de un software se refiere a la capacidad del sistema para responder de manera eficiente a las solicitudes de los usuarios y utilizar los recursos disponibles de manera óptima, según McConnell, el rendimiento se define como un atributo que incluye el tiempo de respuesta, el procesamiento de datos y la optimización del uso de memoria y almacenamiento. El rendimiento influye directamente en la percepción del usuario, pues un software que responde de forma ágil ofrece una experiencia más satisfactoria y confiable, de esta manera, el rendimiento se convierte en un factor estratégico para la adopción y continuidad del uso de un sistema (McConnell, 2021, pág. 588).

Por otro lado, Sommerville resalta que el rendimiento no debe analizarse únicamente en escenarios controlados, sino también considerando la escalabilidad, es decir, la capacidad del software de mantener niveles adecuados de eficiencia a medida que aumenta la carga de trabajo. Un software escalable es capaz de

adaptarse al crecimiento del número de usuarios o a mayores volúmenes de datos sin degradar su desempeño, este aspecto resulta fundamental en aplicaciones modernas que operan en la nube o en plataformas distribuidas (Sommerville, 2020, pág. 152).

Además, McConnell enfatiza la necesidad de aplicar técnicas de análisis y pruebas de rendimiento desde las primeras fases del desarrollo. Estas pruebas permiten identificar cuellos de botella y establecer mecanismos de optimización temprana, evitando problemas de saturación en la fase de producción. Así, el rendimiento no se limita a la velocidad de respuesta, sino que integra la eficiencia en la gestión de recursos y la adaptabilidad del software frente a escenarios cambiantes. (McConnell, 2021)

D) Mantenibilidad

La mantenibilidad combina la capacidad del programa para ser ampliable (extensibilidad), adaptable y servicial (estos tres atributos se denotan con un término más común: mantenibilidad), y además que pueda probarse, ser compatible y configurable (capacidad de organizar y controlar los elementos de la configuración del software) y que cuente con la facilidad para instalarse en el sistema y para que se detecten los problemas. (Pressman, 2010, pág. 188)

Además, Sommerville amplía esta perspectiva al incluir la capacidad de realizar mejoras de manera ágil y eficiente; una buena mantenibilidad implica la reducción de los costos y tiempos asociados al mantenimiento, asegurando la continuidad operativa del sistema. El autor resalta que la mantenibilidad es un factor clave para garantizar la sostenibilidad de un software en el tiempo, ya que los sistemas suelen evolucionar en respuesta a cambios tecnológicos o de negocio (Sommerville, 2020, pág. 152).

Finalmente, Pressman señala que la mantenibilidad puede evaluarse con métricas como la complejidad ciclomática o la densidad de comentarios, que permiten medir la facilidad de comprensión y modificación del código. Un software

mantenible se caracteriza por su adaptabilidad, al permitir la incorporación de nuevas funciones o la actualización de componentes sin comprometer la estabilidad general del sistema, así, la mantenibilidad no solo optimiza recursos, sino que también fortalece la capacidad de innovación y evolución de los proyectos tecnológicos. (Pressman, 2010)

2.2.2.4. Indicadores

A) Cumplimiento de requisitos:

El producto final es el Software que satisface las necesidades del consumidor, con un desempeño apropiado y confiable, y que agrega valor para todos los que lo utilizan. (Pressman, 2010, pág. 338)

B) Tasa de fallos:

Si se considera un sistema basado en computadora, una medida sencilla de su confiabilidad es el tiempo medio entre fallas (TMEF). (Pressman, 2010, pág. 377)

C) Eficiencia del rendimiento:

El rendimiento se prueba bajo condiciones operativas, configuraciones y cargas diferentes a fin de asegurar que el sistema responde a la interacción con el usuario y que maneja la carga extrema sin degradación operativa inaceptable. (Pressman, 2010, pág. 454)

D) Facilidad de mantenimiento:

La capacidad de mantenimiento es la facilidad con la que un programa puede corregirse si se encuentra un error, la facilidad con que se adapta si su entorno cambia o de mejorar si el cliente quiere un cambio en requerimientos. No hay forma de medir directamente la capacidad de mantenimiento; por tanto, deben usarse medidas indirectas. (Pressman, 2010, pág. 583)

2.2.2.5. Pruebas de software y aseguramiento de calidad (QA/QC)

El aseguramiento de la calidad del software (Quality Assurance, QA) y el control de calidad (Quality Control, QC) son componentes complementarios de la gestión de calidad en proyectos de software. QA se enfoca en los procesos y la prevención de defectos: comprende las actividades planificadas y sistemáticas para garantizar que el desarrollo siga estándares y procedimientos adecuados (Bhanushali, 2023).

En otras palabras, QA abarca la definición de políticas, metodologías, capacitaciones y revisiones orientadas a que el producto final cumpla con las expectativas de calidad, como indican Bhanushali (2023) y la Google Testing Blog, QA implica "la mejora continua y consistente de los procesos que nos dan confianza en que el producto cumplirá con las necesidades del cliente". Algunas prácticas de QA incluyen el establecimiento de estándares de codificación, gestión de la configuración, auditorías de calidad, evaluaciones de riesgos y la promoción de una cultura de calidad dentro del equipo de desarrollo.

Por su parte, el control de calidad (QC) se centra en el producto y en la detección de defectos. QC típicamente se implementa mediante pruebas de software, inspecciones y revisiones que verifican si el producto desarrollado cumple con los requisitos y estándares definidos (Altexsoft, 2023).

En la práctica, las pruebas de software son la actividad principal del control de calidad: abarcan desde pruebas unitarias e integración hasta pruebas de sistema, rendimiento, seguridad y usabilidad, realizadas para descubrir fallos antes de la entrega al usuario. Es importante distinguir que, mientras QA es proactivo (previene errores mejorando procesos desde el inicio), QC es reactivo (identifica errores en el producto terminado o en etapas finales) (Bhanushali, 2023).

Por ejemplo, QA establecerá un proceso de code review y estándares de diseño para prevenir defectos, mientras QC ejecutará tests y revisiones específicas para encontrar defectos en la implementación. En un entorno eficaz, ambos van de la mano: QA proporciona el marco para "construir calidad en el producto", y QC

comprueba la calidad alcanzada mediante inspecciones y pruebas (Lewis & Veerapillai, 2004).

La relación entre pruebas, QA y QC se puede resumir así: las pruebas de software son una herramienta fundamental de QC, y el QC a su vez es un componente de las iniciativas de QA globales, por ejemplo, un proceso integral de aseguramiento de calidad incluirá actividades preventivas (como formación del equipo en buenas prácticas, uso de metodologías ágiles o modelos formales) y actividades de control (como ejecutar suites de pruebas automatizadas continuamente).

Estudios recientes destacan el rol crítico de las pruebas en la calidad: aplicar técnicas de prueba apropiadas en ciclos ágiles mejora significativamente la calidad del producto (Thangiah & Basri, 2016), y las organizaciones que adoptan enfoques modernos de QA (integrando pruebas automatizadas, DevOps y CI/CD) logran reducir defectos y entregar software más confiable (Zlatko, 2021). Por ello, QA establece la confianza en que "se está construyendo el producto correcto de la forma correcta", mientras que las pruebas y QC verifican que "el producto funciona correctamente" antes de llegar al usuario final (Bhanushali, 2023).

2.2.2.6. Factores que afectan la calidad del software

La calidad de un producto software no depende solo de buenos requisitos o de pruebas extensivas, sino de una combinación de factores humanos, técnicos, de proceso y organizacionales a lo largo del ciclo de vida del desarrollo. Un factor fundamental es el factor humano: las habilidades, experiencia y formación del equipo de desarrollo tienen un impacto directo en la calidad. Una revisión sistemática de literatura halló que los factores personales —especialmente la experiencia previa del desarrollador y su educación— son la categoría más importante de factores humanos que influyen en la calidad y éxito de un proyecto software (Güveyi, Aktaş, & Kalipsiz, 2020).

Por ejemplo, un equipo con desarrolladores experimentados tiende a producir código más confiable y mantenible, y los factores interpersonales como la

comunicación y coordinación dentro del equipo también juegan un rol clave en prevenir defectos y malentendidos (Machuca, Gasca, Morillo, & Restrepo, 2021). Estudios señalan que fomentar buena comunicación de equipo (reuniones efectivas, revisión de código entre pares, cultura de colaboración) mitiga errores y mejora la consistencia del producto (Canedo & Santos, 2019).

Entre los factores técnicos y de proceso, destaca la complejidad del producto y la gestión de requisitos; un software con requisitos mal definidos o cambiantes incrementa la probabilidad de defectos; igualmente, un diseño complejo o mal estructurado dificulta la implementación correcta y las pruebas (Fitriani, Rahayu, & Sensuse, 2016). Por ello, la calidad de los requisitos y del diseño inicial afecta fuertemente la calidad final: prácticas como análisis exhaustivo de requisitos, diseño modular y arquitecturas claras contribuyen a productos más robustos. Adicionalmente, las prácticas de desarrollo adoptadas son un factor decisivo: el uso de metodologías adecuadas (por ejemplo, metodologías ágiles con iteraciones cortas y retroalimentación continua, o modelos formales en sistemas críticos) puede mejorar la calidad al detectar problemas más temprano.

Un estudio reporta que la adopción de ciertas prácticas ágiles (integración continua, refactorización frecuente, pair programming, etc.) se correlaciona con mejoras en atributos de calidad como mantenibilidad y fiabilidad. Asimismo, las técnicas de prueba y aseguramiento empleadas (p. ej., pruebas automatizadas, test-driven development, análisis estático de código) influyen directamente en la calidad: proyectos que invierten en pruebas rigurosas durante el desarrollo tienden a tener menos defectos en producción (Zlatko, 2021).

Otro conjunto de factores son los organizacionales y de gestión; el tiempo y presupuesto asignados al proyecto afectan la calidad: cronogramas apretados o recortes de presupuesto suelen llevar a compromisos con la calidad (por ejemplo, omitir pruebas o revisar menos el código). Asimismo, la cultura de calidad de la organización y el apoyo de la gerencia son determinantes: organizaciones que priorizan la calidad (p. ej., mediante programas de mejora de procesos tipo CMMI

o ISO 9001, o incluyendo QA desde el inicio del proyecto) logran productos más confiables que aquellas centradas solo en cumplir plazos (Guerrero & Hernández, 2024).

Además, factores externos como la complejidad del dominio de aplicación o la necesidad de integrar componentes de terceros pueden influir: por ejemplo, integrar múltiples librerías o servicios externos aumenta la superficie de posibles fallos (y requiere asegurar la calidad también en esos componentes). Por tanto, la calidad del software es multifactorial: depende de las personas que lo desarrollan, de las prácticas técnicas empleadas (herramientas, metodologías, pruebas), de la gestión del proyecto (planificación realista, control de cambios) y del contexto tecnológico. Identificar y gestionar proactivamente estos factores –por ejemplo, mediante capacitación continua, buenas prácticas de ingeniería y apoyo organizacional– es esencial para lograr software de alta calidad (Zlatko, 2021).

2.2.2.7. Costos de la calidad en desarrollo de software

El concepto de costos de la calidad se refiere a la inversión total necesaria para asegurar la calidad de un producto, así como los costos resultantes de la falta de calidad, según la metodología clásica de Cost of Quality (CoQ) adaptada a software, estos costos se dividen en costos de conformidad (lo que gastamos para prevenir y evaluar defectos) y costos de no conformidad (lo que gastamos por los defectos cuando ocurren) (Radecka, 2024).

Radecka (2024) añade que, dentro de los costos de conformidad, se incluyen: costos de prevención, como la formación del equipo, mejora de procesos, herramientas de calidad, documentación y actividades de diseño orientadas a prevenir errores; y costos de evaluación o evaluación (appraisal), que abarcan las actividades de control de calidad, por ejemplo, inspecciones, pruebas, auditorías y revisiones realizadas durante el desarrollo para detectar defectos antes de la entrega.

En cambio, los costos de no conformidad se dividen en fallos internos (defectos detectados antes de la entrega al usuario) y fallos externos (defectos que llegan al producto liberado o al cliente). Los costos por fallos internos incluyen el

retrabajo: corrección de bugs, reimplementación de funciones, tiempo de inactividad del equipo para diagnosticar y arreglar problemas, etc. Los costos por fallos externos son generalmente los más altos e incluyen: soporte al cliente, gestión de quejas, parches de emergencia, posibles penalizaciones contractuales, pérdida de reputación e incluso daños legales si el fallo provoca incumplimiento normativo o pérdidas a terceros (Radecka, 2024).

En el contexto de desarrollo de software, invertir en calidad temprano resulta económicamente ventajoso porque el costo de corregir defectos crece exponencialmente cuanto más tarde se descubren. Un estudio clásico de IBM demostró que corregir un error en la fase de diseño puede costar X unidades, mientras que corregirlo durante las pruebas cuesta ~15X, y en producción hasta ~100X (según el famoso modelo de Boehm). Es decir, los fallos externos (en producción) son decenas de veces más costosos que los internos. Por ejemplo, si un bug grave se detecta ya en manos del usuario, no solo implica dedicar horas de desarrollo para arreglarlo (retrabajo), sino también desplegar parches de urgencia, gestionar la insatisfacción de usuarios, posibles compensaciones, etc. (Zrymiak, 2020).

Por ello, muchas organizaciones calculan el índice de costo de calidad como la proporción entre lo gastado en prevención/appraisal vs. lo perdido en fallos; el objetivo es equilibrar esa inversión para minimizar el costo total. Estudios recientes como el reporte CISQ 2022 cuantifican el costo de la mala calidad del software: solo en EE. UU. se estimó en $2.41 trillones de dólares anuales, considerando pérdidas por defectos, vulnerabilidades de seguridad y deuda técnica acumulada (Zrymiak, 2020).

En la práctica, gestionar los costos de calidad implica destinar suficientes recursos a actividades de QA/QC (diseño de calidad, revisiones, pruebas exhaustivas, automatización, etc.) –costos de conformidad– para reducir al mínimo los fallos internos y externos. Un balance óptimo es aquel donde una mayor inversión preventiva y de evaluación reduce significativamente los costes de fallos

totales, resultando en un coste total menor. Por ejemplo, implementar pruebas automatizadas y revisión de código (costo de conformidad) requiere esfuerzo, pero previene lanzamientos defectuosos que costarían mucho más en correcciones urgentes y daño reputacional.

Asimismo, no todos los costos de calidad son tangibles: la pérdida de confianza de los clientes o la pérdida de oportunidades de negocio por fallos son difíciles de cuantificar pero muy reales. Por tanto, el concepto de costos de la calidad en software nos insta a invertir estratégicamente en calidad durante el desarrollo –mediante prevención y detección temprana– para evitar pagar el alto precio de la mala calidad posteriormente (Radecka, 2024).

2.2.3. Normativa vigente

2.2.3.1. Normas internacionales aplicables

La ISO 31000:2018 – Risk Management: Guidelines define un marco de referencia internacional para la gestión del riesgo, estableciendo principios, estructura y procesos destinados a identificar, analizar, evaluar, tratar, monitorear y comunicar los riesgos que pueden afectar el cumplimiento de los objetivos de una organización, su aplicación permite integrar la gestión del riesgo en todos los niveles jerárquicos y en la toma de decisiones estratégicas, garantizando que las acciones preventivas y correctivas se basen en criterios objetivos, cuantificables y verificables, la norma destaca la importancia de la mejora continua y la retroalimentación constante como pilares de un sistema de gestión eficaz.

La ISO/IEC 27001:2022 – Information Security Management Systems establece los requisitos para implementar y mantener un sistema de gestión de seguridad de la información, proporcionando un marco para proteger la confidencialidad, integridad y disponibilidad de los activos de información, esta norma se apoya en la evaluación de riesgos para identificar vulnerabilidades y aplicar controles que mitiguen amenazas internas y externas, promoviendo la creación de políticas, procedimientos y auditorías periódicas que fortalezcan la seguridad organizacional y la confianza de las partes interesadas.

La ISO/IEC 25010:2011 – Systems and Software Quality Requirements and Evaluation (SQuaRE) establece el modelo de calidad del software, definiendo ocho características fundamentales: funcionalidad, fiabilidad, usabilidad, eficiencia, mantenibilidad, portabilidad, compatibilidad y seguridad, cada una de ellas se desglosa en subcaracterísticas que permiten medir de manera sistemática el grado en que un producto de software cumple con los requisitos establecidos, la norma proporciona un lenguaje común para la evaluación de la calidad y un marco estructurado para el desarrollo y validación de sistemas informáticos.

El Project Management Body of Knowledge (PMBOK) – Séptima Edición, publicado por el Project Management Institute (PMI) en 2021, recopila las buenas prácticas y fundamentos para la gestión profesional de proyectos, su estructura conceptual abarca principios, dominios de desempeño y herramientas que orientan la planificación, ejecución, control y cierre de los proyectos, el PMBOK considera la gestión de riesgos como un componente transversal que influye en la calidad, el cronograma, el costo y el alcance, promoviendo un enfoque integral que articula las áreas técnicas con las dimensiones humanas y estratégicas de la gestión.

2.2.3.2. Normas nacionales peruanas

La Ley N.º 29733 – Ley de Protección de Datos Personales, promulgada en el año 2011, regula el tratamiento de los datos personales contenidos o destinados a ser incluidos en bancos de datos, sean automatizados o manuales, su objetivo es garantizar el derecho fundamental a la protección de los datos personales reconocido en la Constitución Política del Perú, estableciendo principios rectores como la legalidad, consentimiento, proporcionalidad, calidad, seguridad y disponibilidad, así como la obligación de las entidades públicas y privadas de implementar medidas de seguridad técnicas y organizativas para salvaguardar la información.

La Ley N.º 30096 – Ley de Delitos Informáticos, promulgada en el año 2013, tiene por finalidad tipificar y sancionar las conductas ilícitas que vulneran los sistemas informáticos, las redes de comunicación o los datos electrónicos,

reconociendo la información como un bien jurídico protegido, la norma incorpora categorías de delitos como el acceso no autorizado, la interceptación, la alteración o la destrucción de información, estableciendo penas y mecanismos de cooperación judicial internacional, además promueve la implementación de medidas preventivas mediante políticas de seguridad digital y monitoreo continuo de los sistemas.

El Decreto Supremo N.º 066-2019-PCM, que aprueba la Política Nacional de Transformación Digital, establece los lineamientos generales para la incorporación de tecnologías digitales en la administración pública y privada, promoviendo la innovación, la interoperabilidad, la ciberseguridad y la gestión eficiente de los servicios digitales, esta política busca orientar las acciones del Estado hacia la consolidación de una sociedad digital inclusiva y segura, garantizando que la transformación tecnológica se realice de manera sostenible y con respeto a los derechos ciudadanos.

La Norma Técnica Peruana NTP ISO 31000:2018 – Gestión del riesgo: Directrices, aprobada por el Instituto Nacional de Calidad (INACAL), representa la versión oficial peruana de la norma internacional ISO 31000, su objetivo es establecer los principios y lineamientos para la implementación de sistemas de gestión del riesgo en todo tipo de organizaciones, promoviendo la identificación, análisis, evaluación, tratamiento y monitoreo de los riesgos en coherencia con el contexto nacional, esta norma fomenta la toma de decisiones basadas en evidencia y la adopción de una cultura organizacional de mejora continua y prevención.

2.2.4. Modelos y herramientas técnicas complementarias

En el ámbito de la ingeniería de software, los modelos y herramientas técnicas constituyen un soporte esencial para fortalecer la gestión de riesgos y garantizar la calidad del producto desarrollado, entre ellos destaca el CMMI (Capability Maturity Model Integration), propuesto por el Software Engineering Institute (CMMI Product, 2018), el cual define un marco de madurez compuesto por cinco niveles que orientan la optimización de los procesos organizacionales, desde la gestión inicial hasta la mejora continua, este modelo integra prácticas de

planificación, monitoreo, control y aseguramiento de la calidad, permitiendo evaluar la capacidad de una organización para cumplir sus objetivos de manera predecible y eficiente, su aplicación favorece la estandarización de procedimientos, la reducción de errores y la alineación de los proyectos con estrategias de desarrollo sostenibles.

Dentro de las herramientas de evaluación del rendimiento se encuentra el Application Performance Monitoring (APM), conjunto de tecnologías que permiten medir y supervisar en tiempo real el desempeño de las aplicaciones, estas herramientas recopilan métricas sobre tiempos de respuesta, consumo de recursos, disponibilidad y estabilidad de los sistemas, facilitando la detección temprana de fallos y la optimización del entorno operativo, su empleo garantiza una experiencia de usuario satisfactoria y una gestión eficiente de la infraestructura tecnológica, ya que posibilita el análisis de causas raíz y la implementación de mejoras continuas en los servicios digitales (Lange, 2024).

Por otra parte, las métricas de fiabilidad como el MTTF (Mean Time To Failure) son empleadas para estimar el tiempo promedio que transcurre antes de que ocurra un fallo en un sistema, este indicador constituye una medida clave en la evaluación de la estabilidad y confiabilidad del software, puesto que permite proyectar la durabilidad de los componentes y planificar mantenimientos preventivos, según Pressman (2010), el análisis de fiabilidad contribuye a la toma de decisiones en las fases de validación y despliegue, optimizando la continuidad de las operaciones y reduciendo el impacto de las interrupciones en los procesos críticos.

Asimismo, las herramientas de trazabilidad y control de requerimientos resultan fundamentales para garantizar la coherencia entre las especificaciones del cliente y las funcionalidades desarrolladas, estas permiten registrar, seguir y validar los cambios realizados durante el ciclo de vida del software, reduciendo los riesgos asociados a omisiones o inconsistencias, la trazabilidad, de acuerdo con Sommerville (2020), asegura la integridad del proyecto al proporcionar una visión

integral del flujo de información desde la etapa de diseño hasta la implementación final.

Entre las técnicas de validación se incluyen las pruebas de aceptación, las pruebas de estrés y el análisis de impacto de cambios, las cuales permiten evaluar la robustez del sistema ante cargas extremas, variaciones de entorno o modificaciones de código, estas pruebas son esenciales para medir la capacidad del software de mantener su desempeño bajo condiciones exigentes, garantizando su confiabilidad y estabilidad en escenarios reales de operación (McConnell, 2021).

Finalmente, los patrones de diseño de software desempeñan un papel determinante en la arquitectura de sistemas al promover el desacoplamiento de componentes y la reutilización de estructuras de programación, dichos patrones establecen soluciones probadas a problemas recurrentes de diseño y mejoran la mantenibilidad y escalabilidad del software, Gamma et al. (2019) destacan que su uso sistemático facilita la implementación de políticas de codificación limpias y coherentes, asegurando la consistencia del producto y la eficiencia del proceso de desarrollo, de esta manera, los modelos y herramientas descritos conforman un conjunto integral de prácticas que fortalecen la fiabilidad, el control y la mejora continua dentro de los procesos de ingeniería de software.

2.3. Definición de conceptos

A. Calidad de software: Proceso eficaz de software que se aplica de manera que crea un producto útil que proporciona valor medible a quienes lo producen y a quienes lo utilizan. (Pressman, 2010, pág. 340)

B. Fiabilidad del software: la probabilidad que tiene un programa de cómputo de operar sin fallas en un ambiente específico por un tiempo específico. (Pressman, 2010, pág. 376)

C. Funcionalidad del software: Grado en el que el software satisface las necesidades planteadas según las establecen los atributos siguientes:

adaptabilidad, exactitud, interoperabilidad, cumplimiento y seguridad. (Pressman, 2010, pág. 343)

D. Gestión de riesgos: Proceso sistemático para identificar, evaluar y mitigar los riesgos que puedan afectar negativamente un proyecto o sistema (ISO 31000, 2018), la gestión de riesgos busca minimizar los impactos adversos mientras se maximizan las oportunidades.

E. ISO 27001: Norma internacional que especifica los requisitos para establecer, implementar, mantener y mejorar un sistema de gestión de seguridad de la información (SGSI).

F. ISO 31000: Es una norma internacional que proporciona principios y directrices para la gestión de riesgos. Describe un enfoque integral para identificar, analizar, evaluar, tratar, supervisar y comunicar los riesgos en toda la organización. (International Standardization Organization)

G. Magerit: Es una metodología de carácter público que puede ser utilizada libremente y no requiere autorización previa. Interesa principalmente a las entidades en el ámbito de aplicación del Esquema Nacional de Seguridad (ENS) para satisfacer el principio de la gestión de la seguridad basada en riesgos, así como el requisito de análisis y gestión de riesgos, considerando la dependencia de las tecnologías de la información para cumplir misiones, prestar servicios y alcanzar los objetivos de la organización. (Consejo Superior de Administración Electrónica de España)

H. Mantenibilidad del software: Facilidad con la que pueden efectuarse reparaciones al software, según lo indican los atributos que siguen: analizable, cambiable, estable, susceptible de someterse a pruebas. plazo (Pressman, 2010, pág. 343).

I. Proceso de desarrollo de software: Conjunto de actividades planificadas y estructuradas que se llevan a cabo para crear, desarrollar, probar y mantener

aplicaciones o sistemas de software. Incluye etapas como análisis, diseño, implementación, pruebas y mantenimiento (Sommerville, 2020).

J. Rendimiento: Grado en el que el software emplea óptimamente los recursos del sistema, según lo indican los subatributos siguientes: comportamiento del tiempo y de los recursos. (Pressman, 2010, pág. 343)

CAPÍTULO III: MARCO METODOLÓGICO

3.1. Hipótesis

3.1.1. Hipótesis general

La gestión de riesgos tiene una relación significativa con la calidad del software realizado por profesionales del CIP, Tacna – 2024.

3.1.2. Hipótesis específicas

La gestión de riesgos se relaciona significativamente con la funcionalidad del software realizado por profesionales del CIP, Tacna – 2024.

La gestión de riesgos se relaciona significativamente con la fiabilidad del software realizado por profesionales del CIP, Tacna – 2024.

La gestión de riesgos se relaciona significativamente con el rendimiento del software realizado por profesionales del CIP, Tacna – 2024.

La gestión de riesgos se relaciona significativamente con la mantenibilidad del software realizado por profesionales del CIP, Tacna – 2024.

3.2. Operacionalización de variables

3.2.1. Identificación de la primera variable

La primera variable es la gestión de riesgos en los proyectos de software, la cual se refiere al proceso mediante el cual se identifican, evalúan y responden los riesgos asociados al desarrollo de software, con el fin de mitigar su impacto en el proyecto.

A. Dimensiones

Las dimensiones de la gestión de riesgos son:

- Procesos de gestión de riesgos: El análisis y la administración del riesgo son acciones que ayudan al equipo de software a entender y manejar la incertidumbre. Muchos problemas pueden plagar un proyecto de software. Un riesgo es un problema potencial: puede ocurrir, puede no ocurrir. Pero, sin importar el resultado, realmente es una buena idea identificarlo, valorar su probabilidad de ocurrencia, estimar su impacto y establecer un plan de contingencia para el caso de que el problema realmente ocurra. (Pressman, 2010, pág. 640)

- Gestión de recursos financieros: Antes de que el proyecto pueda comenzar, el equipo de software debe estimar el trabajo que se va a realizar, los recursos que se requerirán y el tiempo que transcurrirá de principio a fin. (Pressman, 2010, pág. 593)

- Gestión de talento humano: El "factor humano" es tan importante que el Software Engineering Institute desarrolló un Modelo de madurez de capacidades del personal (People-CMM, por sus siglas en inglés), en reconocimiento al hecho de que "toda organización requiere mejorar continuamente su habilidad para atraer, desarrollar, motivar, organizar y conservar la fuerza de trabajo necesaria a fin de lograr sus objetivos empresariales estratégicos". (Pressman, 2010, pág. 555)

B. Indicadores

Los indicadores para estas dimensiones son:

- Efectividad de los procesos de gestión de riesgos. (Pressman, 2010, pág. 642)

- Adecuación de la gestión financiera en la mitigación de riesgos. (Pressman, 2010, pág. 594)

- Percepción de la capacidad del equipo en la gestión de riesgos. (Pressman, 2010, pág. 568)

C. Escala para la medición de la variable

La escala para medir la gestión de riesgos será una escala Likert de 5 puntos, donde:

1: Muy en desacuerdo

2: En desacuerdo

3: Neutral

4: De acuerdo

5: Muy de acuerdo

3.2.2. Identificación de la segunda variable

La segunda variable es la calidad del software, que mide el grado en el que el software cumple con los requisitos funcionales y no funcionales, además de las expectativas del usuario final en términos de funcionalidad, fiabilidad, rendimiento y mantenibilidad.

A. Dimensiones

Las dimensiones de la calidad del software son:

- Funcionalidad: La funcionalidad se califica de acuerdo con el conjunto de características y capacidades del programa, la generalidad de las funciones que se entregan y la seguridad general del sistema. (Pressman, 2010, pág. 188)

- Fiabilidad: La fiabilidad se evalúa con la medición de la frecuencia y gravedad de las fallas, la exactitud de los resultados que salen, el tiempo medio para que ocurra una falla (TMPF), la capacidad de recuperación ante ésta y lo predecible del programa. (Pressman, 2010, pág. 188)

- Rendimiento: El rendimiento se mide con base en la velocidad de procesamiento, el tiempo de respuesta, el uso de recursos, el conjunto y la eficiencia. (Pressman, 2010, pág. 188)

- Mantenibilidad: La mantenibilidad combina la capacidad del programa para ser ampliable (extensibilidad), adaptable y servicial (estos tres atributos se denotan con un término más común: mantenibilidad), y además que pueda probarse, ser compatible y configurable (capacidad de organizar y controlar los elementos de la configuración del software) y que cuente con la facilidad para instalarse en el sistema y para que se detecten los problemas. (Pressman, 2010, pág. 188)

B. Indicadores

Los indicadores para estas dimensiones son:

- Cumplimiento de requisitos. (Pressman, 2010, pág. 338)

- Tasa de fallos. (Pressman, 2010, pág. 377)

- Eficiencia del rendimiento. (Pressman, 2010, pág. 454)

- Facilidad de mantenimiento. (Pressman, 2010, pág. 583)

C. Escala para la medición de la variable

La escala para medir la calidad del software será una escala Likert de 5 puntos, donde:

1: Muy bajo

2: Bajo

3: Regular

4: Alto

5: Muy alto

3.2.3. Cuadro de operacionalización de variables

Variable	Dimensiones	Indicadores
Gestión de riesgos. (Pressman, 2010, pág. 640)	Procesos de gestión de riesgos. (Pressman, 2010, pág. 640)	Efectividad de los procesos de gestión de riesgos. (Pressman, 2010, pág. 642)
	Gestión de recursos financieros. (Pressman, 2010, pág. 593)	Adecuación de la gestión financiera en la mitigación de riesgos. (Pressman, 2010, pág. 594)
	Gestión de talento humano. (Pressman, 2010, pág. 555)	Percepción de la capacidad del equipo en la gestión de riesgos. (Pressman, 2010, pág. 568)
Calidad de software. (Pressman, 2010, pág. 340)	Funcionalidad. (Pressman, 2010, pág. 188)	Cumplimiento de requisitos. (Pressman, 2010, pág. 338)
	Fiabilidad. (Pressman, 2010, pág. 188)	Tasa de fallos. (Pressman, 2010, pág. 377)
	Rendimiento. (Pressman, 2010, pág. 188)	Eficiencia del rendimiento. (Pressman, 2010, pág. 454)
	Mantenibilidad. (Pressman, 2010, pág. 188)	Facilidad de mantenimiento. (Pressman, 2010, pág. 583)

3.3. Tipo de investigación

La investigación fue de tipo básica, orientada a generar conocimientos y teorías que describan, expliquen y predigan los fenómenos observados, sin buscar una aplicación práctica inmediata (Tamayo, 2019). El enfoque se centró en entender profundamente cómo la gestión de riesgos influye en la calidad del software, con el objetivo de contribuir a la teoría existente en el campo de la ingeniería de software.

En este caso, cobra relevancia por el contexto actual de transformación digital y la adopción de metodologías ágiles en los proyectos de software, lo cual exige comprender cómo la gestión de riesgos se integra en ciclos de desarrollo más cortos y flexibles; este panorama marca una oportunidad para aportar evidencia científica sobre los desafíos y beneficios de dicha integración.

3.4. Nivel de investigación

El nivel de investigación fue correlacional, ya que busca determinar la relación existente entre las variables estudiadas (Reyes, 2022). Resulta apropiado al analizar cómo la gestión de riesgos, considerada como un conjunto de procesos preventivos y correctivos, se asocia con la calidad del software en dimensiones específicas como la funcionalidad, la fiabilidad, el rendimiento y la mantenibilidad, relevante en proyectos locales donde los ingenieros enfrentan entornos de desarrollo intensivos en cambios tecnológicos.

3.5. Diseño de investigación

El diseño fue no experimental, transversal, esto implica que se recogieron datos en un único punto temporal, analizando las variables sin manipulación directa por parte del investigador (Rodríguez, 2020). Este diseño permite observar las relaciones y condiciones tal como se presentan en su contexto natural en los proyectos de software (Requena & Alcaide, 2017).

3.6. Ámbito y tiempo social de la investigación

La investigación se realizó específicamente en la ciudad de Tacna, la unidad de análisis fueron los ingenieros colegiados del CIP – Tacna del capítulo de Sistemas.

La investigación se realizó desde el mes de agosto del año 2024 y concluyó en junio del año 2025, el tiempo de 11 meses comprende la recopilación y análisis de información.

3.7. Población y muestra

3.7.1. Unidad de estudio

La unidad de estudio estuvo conformada por los ingenieros colegiados del Capítulo de Ingenieros de Sistemas del Colegio de Ingenieros del Perú – Consejo Departamental Tacna. Específicamente, se consideraron como unidad de análisis aquellos profesionales que participan o han participado en actividades vinculadas al desarrollo de software en instituciones públicas o privadas, y que cuentan con experiencia en procesos relacionados con la gestión de riesgos y/o calidad del software.

3.7.2. Población

La población estuvo compuesta por los 754 ingenieros colegiados en el Capítulo de Ingenieros de Sistemas del Colegio de Ingenieros del Perú (CIP) de Tacna, registrados desde el 01/01/1962 hasta el 31/05/2024.

3.7.3. Muestra

Se utilizó un muestreo probabilístico aleatorio simple para seleccionar el total de la muestra representativa de la población total. Esto permitió que cada miembro de la población tenga una probabilidad igual y conocida de ser seleccionado. Sin embargo, se aplicaron criterios de inclusión a los ingenieros que estén trabajando en el área.

$$n = \frac{N \times Z^2 \times p \times q}{(N-1) \times e^2 + Z^2 \times p \times q}$$

$$n = \frac{754 \times 1.96^2 \times 0.5 \times 0.5}{(754-1) \times 0.05^2 + 1.96^2 \times 0.5 \times 0.5} = 255$$

Técnica de Selección:

A pesar de que el tamaño fue calculado por un método probabilístico, la selección final de los 255 participantes se llevó a cabo a través de un Muestreo No Probabilístico por Juicio (o Intencional). Esta decisión metodológica se fundamentó en la naturaleza especializada de las variables y la necesidad de garantizar que los encuestados tuvieran el conocimiento y experiencia directos sobre el fenómeno de estudio. Por lo tanto, solo se incluyeron aquellos individuos que cumplieron rigurosamente con los siguientes criterios de inclusión:

Criterios de inclusión:

- Ingenieros colegiados activos en el Capítulo de Ingenieros de Sistemas del CIP de Tacna.
- Ingenieros con experiencia en desarrollo de software, participando en al menos un proyecto en los últimos dos años.
- Ingenieros con participación en proyectos que integren prácticas de gestión de riesgos en el último año.
- Ingenieros con disposición para participar en la encuesta y proporcionar información precisa.

3.8. Procedimiento, técnicas e instrumentos

3.8.1. Procedimiento

Se obtuvo el listado oficial de los ingenieros habilitados en el CIP – Tacna, se diseñó una sección informativa dentro del cuestionario que indicaba los criterios de inclusión para validar si el potencial participante cumplía con los criterios de inclusión.

Se contactó a los ingenieros por medios electrónicos digitales solicitándoles el llenado del instrumento en caso cumplan con los criterios de inclusión. Así mismo, se solicitó el consentimiento informado a los participantes.

El procesamiento de los datos se inició con la recolección y depuración de la información obtenida mediante encuestas aplicadas a los ingenieros colegiados de la región Tacna. Posteriormente, los datos fueron codificados, organizados y almacenados en bases de datos, utilizando herramientas estadísticas como SPSS y Microsoft Excel para garantizar un manejo sistemático y confiable de la información.

Una vez consolidada la base de datos, los resultados fueron procesados y representados mediante tablas, gráficos y diagramas, con el fin de facilitar la interpretación y visualización de las relaciones entre las variables de estudio. Esta etapa permitió identificar patrones y tendencias relevantes que aportaron claridad al análisis.

Para el contraste de hipótesis se aplicaron estadísticos descriptivos e inferenciales, empleando en particular la prueba de correlación de Spearman, debido a que los datos no presentaron una distribución normal. Dicho procedimiento permitió evaluar la relación entre la gestión de riesgos y la calidad del software, considerando sus dimensiones de funcionalidad, fiabilidad, rendimiento y mantenibilidad. Finalmente, los resultados fueron contrastados con los objetivos e hipótesis planteados, generando conclusiones sólidas y recomendaciones orientadas a optimizar la gestión de proyectos de software en la región.

3.8.2. Técnicas

Las encuestas fueron utilizadas como la principal técnica de recolección de datos, complementadas con el análisis de documentación pertinente disponible en el CIP para validar y enriquecer la información obtenida.

3.8.3. Instrumentos

Se desarrolló un cuestionario estructurado basado en una escala Likert para medir las percepciones y opiniones de los ingenieros sobre la efectividad de la gestión de riesgos y su impacto en las distintas dimensiones de la calidad del software.

3.9. Evaluación técnica y factibilidad

3.9.1. Evaluación técnica

El cuestionario fue diseñado en base a las dimensiones operacionales de ambas variables, garantizando la cobertura de los aspectos conceptuales y prácticos del estudio, fue validado mediante juicio de expertos, quienes evaluaron su pertinencia, claridad y coherencia, la confiabilidad se determinó mediante el coeficiente Alfa de Cronbach, obteniendo valores satisfactorios para ambas variables.

El análisis estadístico se desarrolló en SPSS, permitiendo procesar la información con precisión y aplicar pruebas de correlación adecuadas a la naturaleza de los datos, de esta manera se asegura la validez técnica del procedimiento.

3.9.2. Factibilidad

El estudio fue viable técnica y logísticamente por las siguientes razones:

- Acceso a la población: disponibilidad de contacto con ingenieros colegiados del CIP Tacna.
- Recursos tecnológicos: uso de software estadístico y herramientas digitales de libre acceso.
- Recursos humanos: participación directa del investigador y disposición de los profesionales encuestados.
- Recursos económicos: proyecto autofinanciado, con costos mínimos asociados a la gestión y procesamiento digital de datos.

CAPÍTULO IV: RESULTADOS

4.1. Desarrollo del proyecto y validación del diseño

4.1.1. Diseño de la presentación de los resultados

En el análisis descriptivo se emplearon tablas de frecuencia absolutas y relativas para cada uno de los ítems, dimensiones y variables, permitiendo observar la distribución de las respuestas y categorizar los niveles de gestión y calidad. Para ello, se utilizó un baremo de categorización específico que clasifica los puntajes obtenidos en tres niveles: Bajo, Medio y Alto, este baremo se aplicó a ambas variables, así como a sus respectivas dimensiones, según los rangos detallados en la siguiente tabla.

Tabla 1
Baremos de categorización de variables y dimensiones

	Bajo	**Medio**	**Alto**
Gestión de riesgos	9 a 21	22 a 33	34 a 45
Procesos de gestión de riesgos	3 a 7	8 a 11	12 a 15
Gestión de recursos financieros	3 a 7	8 a 11	12 a 15
Gestión de talento humano	3 a 7	8 a 11	12 a 15
Calidad de software	12 a 28	29 a 44	45 a 60
Funcionalidad	3 a 7	8 a 11	12 a 15
Fiabilidad	3 a 7	8 a 11	12 a 15
Rendimiento	3 a 7	8 a 11	12 a 15
Mantenibilidad	3 a 7	8 a 11	12 a 15

Por otro lado, el análisis inferencial incluyó la prueba de normalidad mediante los estadísticos de Kolmogorov-Smirnov y Shapiro-Wilk, cuyos

resultados indicaron una distribución no normal de los datos. En consecuencia, se aplicó la correlación de Spearman para determinar la relación entre las variables y sus dimensiones. Esta prueba permitió comprobar las hipótesis planteadas, mostrando correlaciones significativas entre la gestión de riesgos y las dimensiones de funcionalidad, fiabilidad, rendimiento y mantenibilidad del software.

4.1.2. Resultados

4.1.2.1. Análisis descriptivo de la Gestión de riesgos

Análisis e interpretación de los ítems

Tabla 2
Los riesgos potenciales son identificados de manera oportuna y completa en los proyectos

	Frecuencia	**Porcentaje**
Muy en desacuerdo	6	2,35
En desacuerdo	48	18,82
Neutral	107	41,96
De acuerdo	72	28,24
Muy de acuerdo	22	8,63
Total	**255**	**100,00**

Análisis e interpretación de resultados:

De acuerdo con los resultados, el 41,96 % de los encuestados se mantuvo en posición neutral frente a la afirmación, mientras que solo un 28,24 % estuvo de acuerdo y un 8,63 % muy de acuerdo. En contraste, el 21,17 % expresó desacuerdo o fuerte desacuerdo. Esta distribución sugiere que, si bien existe una percepción positiva moderada sobre la identificación oportuna y completa de los riesgos en los proyectos, también hay una proporción significativa de profesionales que no lo evidencian con claridad. Este resultado podría indicar una falta de estandarización o documentación adecuada en los procesos de identificación de riesgos dentro de los equipos de desarrollo.

Figura 1
Los riesgos potenciales son identificados de manera oportuna y completa en los proyectos

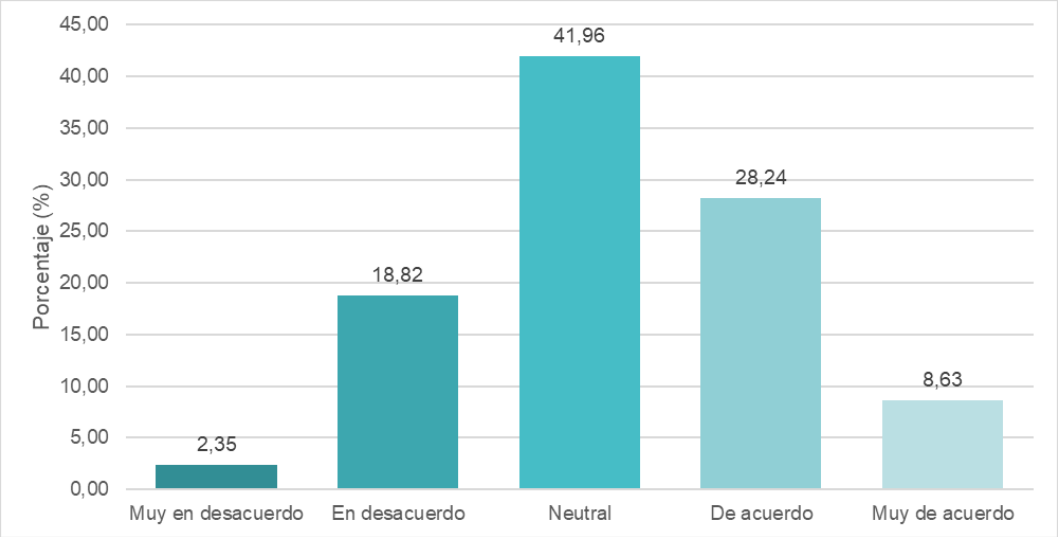


Tabla 3
Se dispone de procedimientos establecidos para documentar y gestionar los riesgos

	Frecuencia	Porcentaje
Muy en desacuerdo	16	6,27
En desacuerdo	53	20,78
Neutral	65	25,49
De acuerdo	104	40,78
Muy de acuerdo	17	6,67
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 47,45 % de los encuestados estuvo de acuerdo o muy de acuerdo con que existen procedimientos establecidos para documentar y gestionar los riesgos, mientras que el 25,49 % se mostró neutral. Por otro lado, un 27,05 % manifestó desacuerdo. Esto refleja una percepción mayoritariamente favorable hacia la existencia de protocolos, aunque todavía hay una porción importante de profesionales que duda o no los reconoce claramente. Aunque algunas organizaciones han formalizado procesos de gestión de riesgos, estas prácticas aún no están plenamente estandarizadas ni difundidas entre todos los equipos de trabajo.

Figura 2
Se dispone de procedimientos establecidos para documentar y gestionar los riesgos

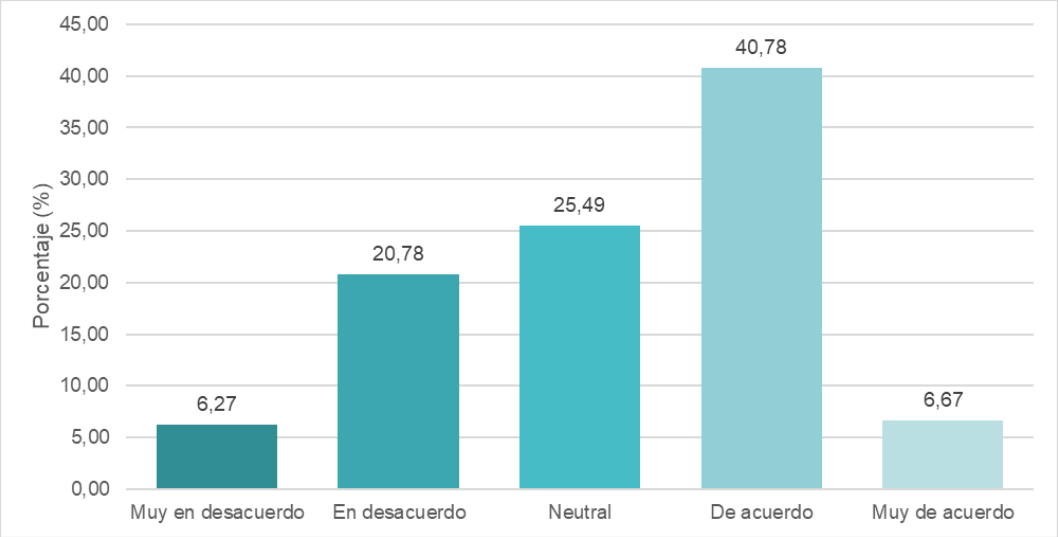


Tabla 4
Los procesos de gestión de riesgos son evaluados regularmente para garantizar su efectividad

	Frecuencia	Porcentaje
Muy en desacuerdo	1	0,39
En desacuerdo	22	8,63
Neutral	87	34,12
De acuerdo	102	40,00
Muy de acuerdo	43	16,86
Total	**255**	**100,00**

Análisis e interpretación de resultados: Los datos muestran que el 56,86 % de los profesionales considera que los procesos de gestión de riesgos son evaluados regularmente, mientras que el 34,12 % adoptó una posición neutral y solo el 9,02 % expresó desacuerdo. Este resultado evidencia una tendencia positiva hacia la revisión continua de los procesos, lo cual es clave para garantizar su efectividad. Sin embargo, la proporción de respuestas neutrales podría estar asociada a una falta de claridad en la periodicidad o profundidad de dichas evaluaciones. Por tanto, aunque la práctica existe, podría requerirse una mejor sistematización o comunicación interna.

Figura 3
Los procesos de gestión de riesgos son evaluados regularmente para garantizar su efectividad

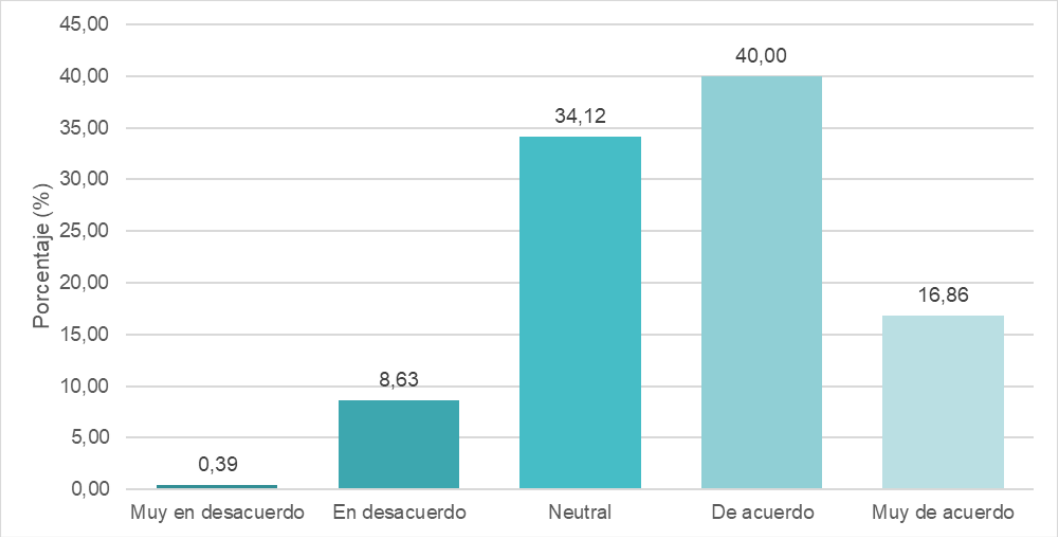


Tabla 5
Los recursos financieros asignados son suficientes para abordar los riesgos identificados

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	84	32,94
Neutral	87	34,12
De acuerdo	53	20,78
Muy de acuerdo	22	8,63
Total	**255**	**100,00**

Análisis e interpretación de resultados: Los resultados muestran que el 36,47 % de los encuestados considera que los recursos financieros asignados son suficientes para abordar los riesgos identificados, frente a un 34,12 % que se mantuvo neutral y un 36,47 % que expresó desacuerdo. Esta distribución revela una percepción dividida entre los profesionales, donde una parte importante no está convencida de la suficiencia de los recursos, esto podría reflejar limitaciones presupuestales o una falta de alineación entre la planificación financiera y las verdaderas necesidades de los proyectos.

Figura 4
Los recursos financieros asignados son suficientes para abordar los riesgos identificados

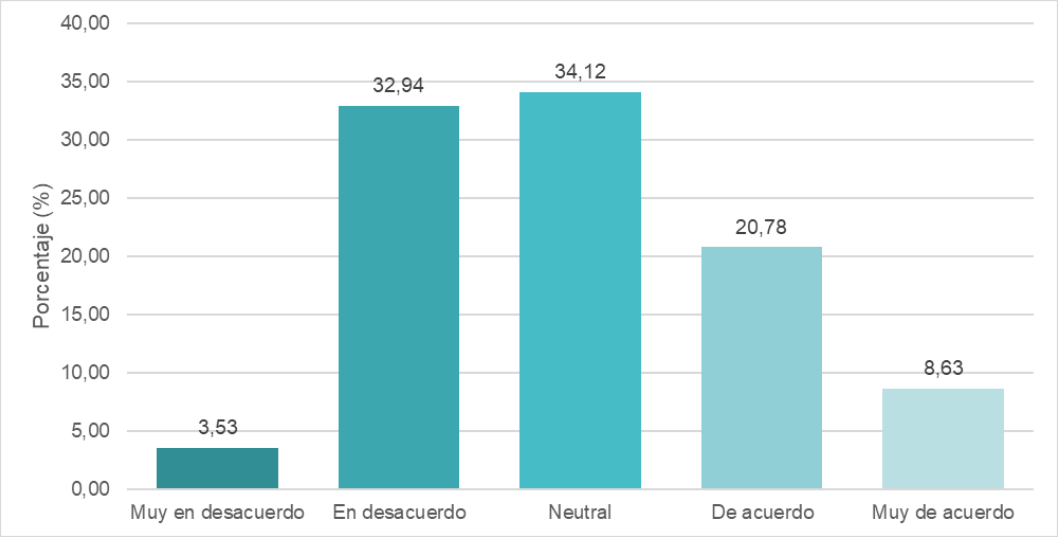


Tabla 6
El uso de los recursos financieros es supervisado regularmente para garantizar su efectividad en la mitigación de riesgos

	Frecuencia	Porcentaje
Muy en desacuerdo	1	0,39
En desacuerdo	33	12,94
Neutral	93	36,47
De acuerdo	120	47,06
Muy de acuerdo	8	3,14
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 50,2 % de los profesionales encuestados considera que el uso de los recursos financieros es supervisado regularmente para garantizar su efectividad, mientras que el 36,47 % adoptó una postura neutral y un 13,33 % expresó desacuerdo. Estos resultados indican que, si bien hay una percepción mayoritaria de control, existe también una parte significativa que no lo percibe claramente o lo desconoce. La elevada neutralidad sugiere que los mecanismos de supervisión podrían no estar siendo comunicados o documentados de manera efectiva, lo que debilita la confianza del equipo en los procesos de monitoreo financiero en la gestión de riesgos.

Figura 5
El uso de los recursos financieros es supervisado regularmente para garantizar su efectividad en la mitigación de riesgos

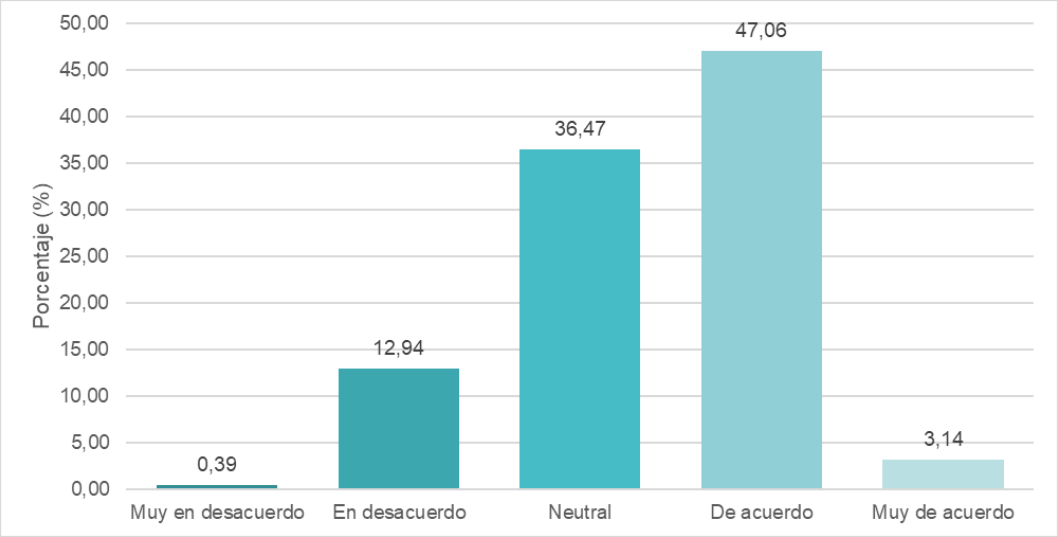


Tabla 7
Se realizan ajustes en los recursos financieros según la evolución de los riesgos del proyecto

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	9	3,53
Neutral	50	19,61
De acuerdo	173	67,84
Muy de acuerdo	14	5,49
Total	**255**	**100,00**

Análisis e interpretación de resultados: La mayoría de los encuestados (73,33 %) estuvo de acuerdo o muy de acuerdo en que se realizan ajustes en los recursos financieros según la evolución de los riesgos del proyecto. Solo un 19,61 % se mantuvo en posición neutral, y el 7,06 % expresó desacuerdo. Este resultado refleja una percepción positiva sobre la capacidad de los equipos para adaptarse a los cambios del entorno mediante la reasignación de recursos. Este tipo de práctica es fundamental para una gestión de riesgos dinámica y efectiva, lo que sugiere que en los proyectos evaluados existe una gestión financiera flexible y receptiva ante posibles amenazas.

Figura 6
Se realizan ajustes en los recursos financieros según la evolución de los riesgos del proyecto

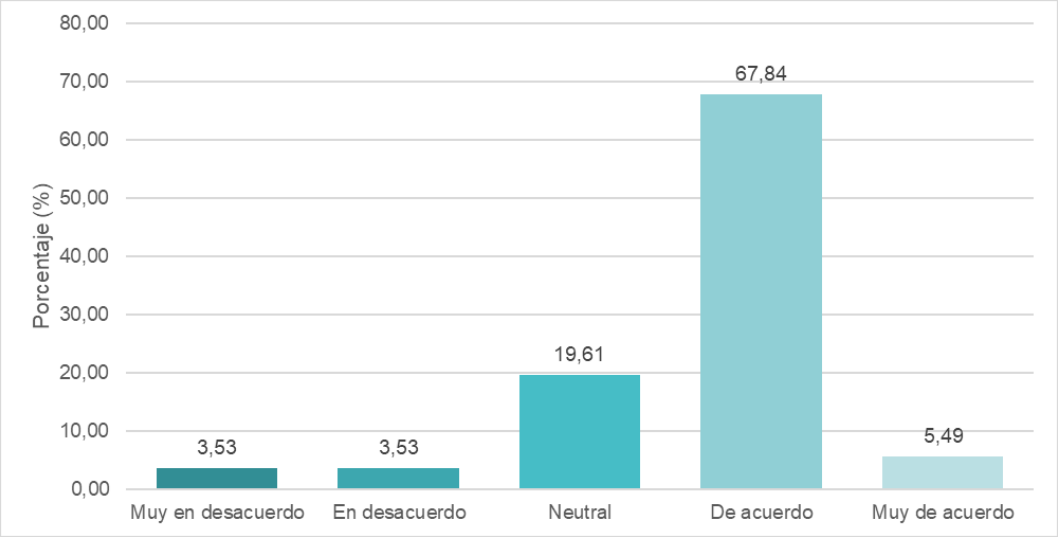


Tabla 8
El equipo de los proyectos cuenta con la experiencia necesaria para gestionar eficazmente los riesgos

	Frecuencia	Porcentaje
Muy en desacuerdo	1	0,39
En desacuerdo	34	13,33
Neutral	73	28,63
De acuerdo	115	45,10
Muy de acuerdo	32	12,55
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 57,65 % de los encuestados está de acuerdo o muy de acuerdo con que los equipos cuentan con la experiencia necesaria para gestionar riesgos. Sin embargo, un 28,63 % se mostró neutral, mientras que el 13,72 % expresó desacuerdo. Esto sugiere que, aunque más de la mitad de los profesionales percibe un nivel adecuado de experiencia en sus equipos, existe una fracción considerable que no tiene una opinión clara al respecto o lo cuestiona. Estos datos podrían estar relacionados con la falta de certificaciones formales, experiencia en gestión de riesgos o escasa rotación de roles en proyectos similares.

Figura 7
El equipo de los proyectos cuenta con la experiencia necesaria para gestionar eficazmente los riesgos

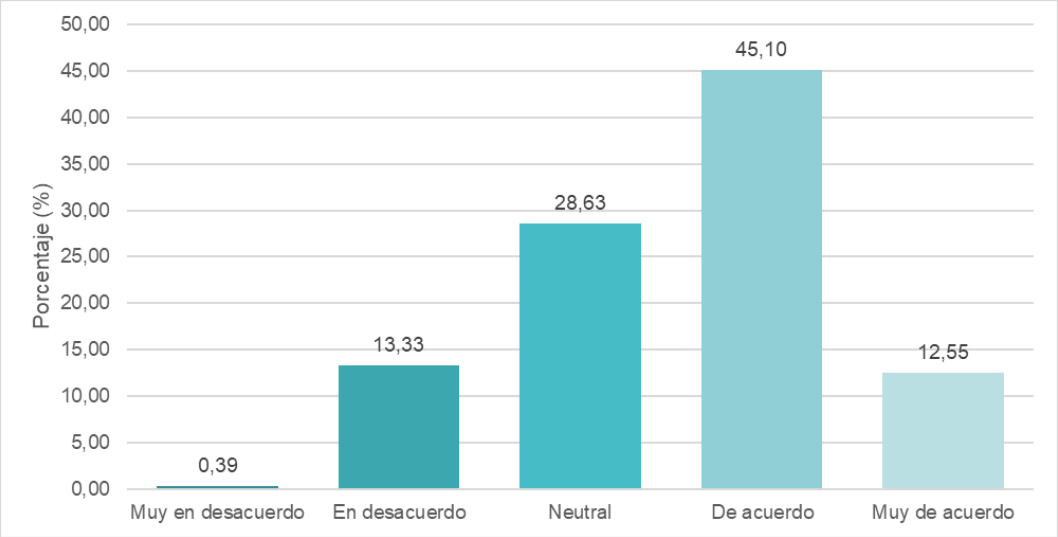


Tabla 9
Se proporciona capacitación regular al equipo sobre la identificación y mitigación de riesgos en proyectos de software

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	24	9,41
Neutral	82	32,16
De acuerdo	118	46,27
Muy de acuerdo	22	8,63
Total	**255**	**100,00**

Análisis e interpretación de resultados:

El 54,90 % de los profesionales considera que se brinda capacitación regular al equipo sobre la identificación y mitigación de riesgos, mientras que un 32,16 % optó por una postura neutral y el 12,94 % mostró desacuerdo. Estos resultados revelan que más de la mitad de los encuestados reconoce prácticas de formación continua, lo que es favorable para la gestión efectiva de riesgos. Sin embargo, el alto porcentaje de neutralidad puede evidenciar falta de uniformidad o de constancia en los programas de capacitación, lo que limita el impacto positivo de estas estrategias en los distintos contextos organizacionales.

Figura 8
Se proporciona capacitación regular al equipo sobre la identificación y mitigación de riesgos en proyectos de software

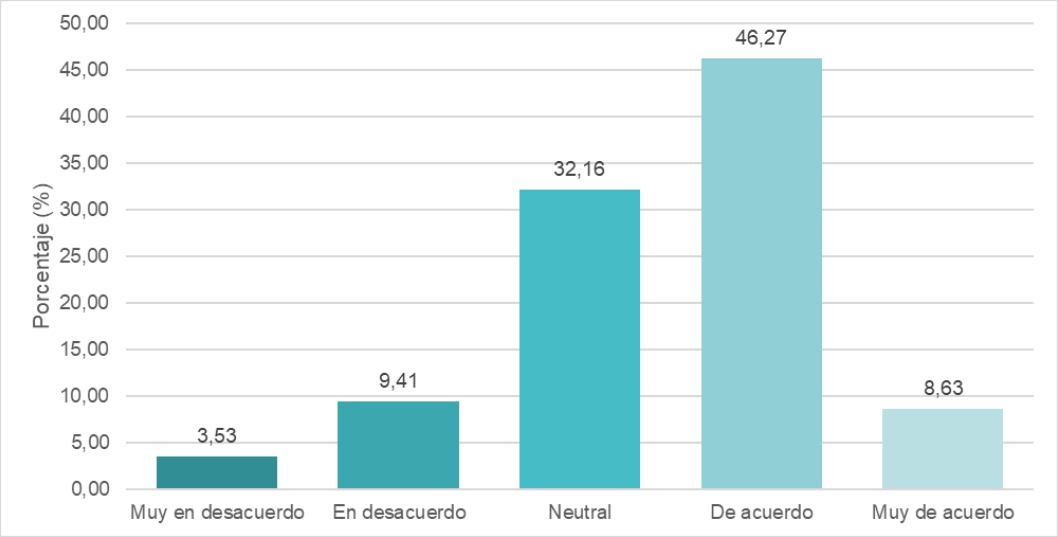


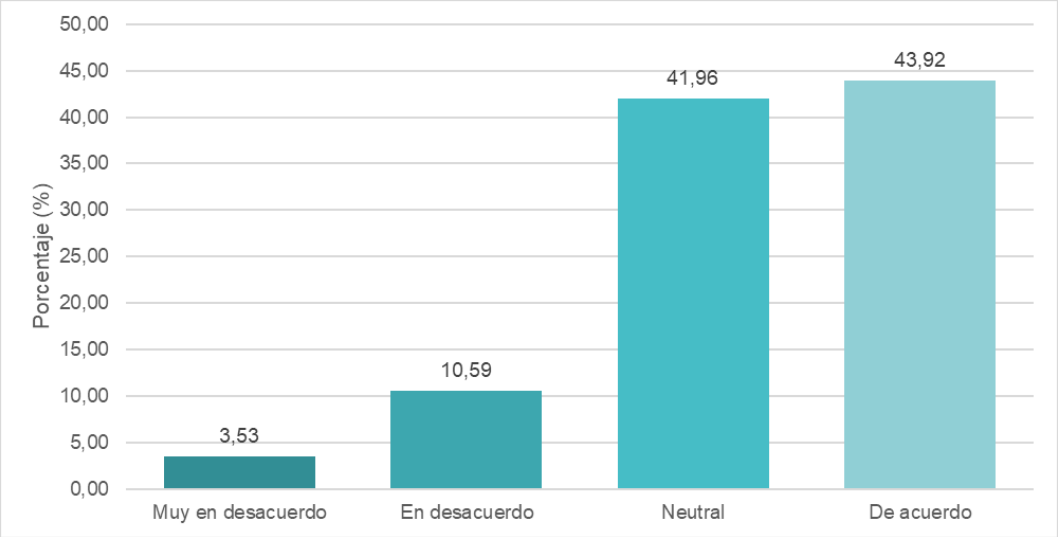
Tabla 10
Se dispone de herramientas específicas para apoyar la gestión de riesgos en los proyectos

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	27	10,59
Neutral	107	41,96
De acuerdo	112	43,92
Total	**255**	**100,00**

Análisis e interpretación de resultados:

El 43,92 % de los encuestados manifestó estar de acuerdo con que se cuenta con herramientas específicas para la gestión de riesgos, mientras que un 41,96 % adoptó una postura neutral y el 14,12 % expresó desacuerdo. Esta distribución evidencia una percepción dividida. Aunque una parte significativa de profesionales afirma contar con estas herramientas, otra proporción considerable no puede confirmarlo con claridad, lo que podría estar relacionado con el bajo nivel de difusión, capacitación o adopción tecnológica. Esto sugiere que, si bien existen recursos disponibles, no todos los equipos los están aprovechando de manera efectiva o consistente.

Figura 9
Se dispone de herramientas específicas para apoyar la gestión de riesgos en los proyectos



Análisis e interpretación de la variable y dimensiones categorizadas

Tabla 11
Nivel de la dimensión Procesos de gestión de riesgos

	Frecuencia	**Porcentaje**
Bajo	43	16,86
Medio	146	57,25
Alto	66	25,88
Total	**255**	**100,00**

Análisis e interpretación de resultados: En cuanto a la dimensión procesos de gestión de riesgos, el 57,25 % de los profesionales evaluados mostró un nivel medio, seguido por un 25,88 % con nivel alto y un 16,86 % en nivel bajo. Estos resultados reflejan que la mayoría de los encuestados reconoce que existen acciones parcialmente desarrolladas para gestionar los riesgos, aunque no con un nivel óptimo. La proporción moderada en el nivel alto sugiere avances positivos en algunos contextos, mientras que el porcentaje bajo indica espacios donde aún no se han implementado procesos sólidos.

Figura 10
Nivel de la dimensión Procesos de gestión de riesgos

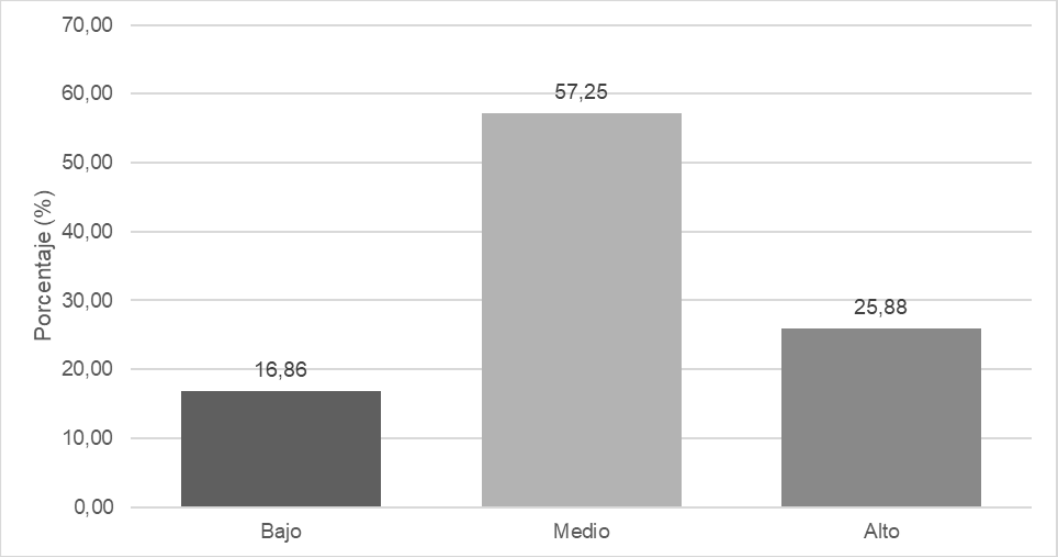


Tabla 12
Nivel de la dimensión Gestión de recursos financieros

	Frecuencia	Porcentaje
Bajo	21	8,24
Medio	170	66,67
Alto	64	25,10
Total	**255**	**100,00**

Análisis e interpretación de resultados: Los resultados muestran que el 66,67 % de los encuestados se ubica en un nivel medio respecto a la gestión de recursos financieros en la mitigación de riesgos. Un 25,10 % logró un nivel alto y solo un 8,24 % presentó un nivel bajo. Esta distribución revela una tendencia positiva, aunque predominante en la categoría media, lo cual indica que la planificación financiera está presente, pero con limitaciones para alcanzar la efectividad deseada. El reducido porcentaje en nivel bajo sugiere que, en general, existen prácticas de asignación de recursos, aunque probablemente no siempre se ajusten con precisión a las necesidades de riesgo.

Figura 11
Nivel de la dimensión Gestión de recursos financieros

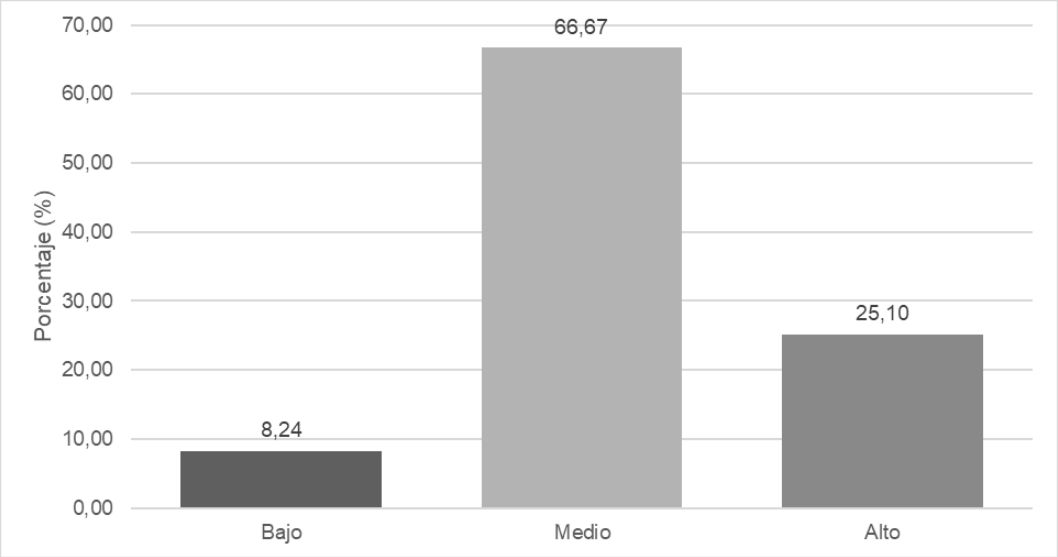


Tabla 13
Nivel de la dimensión Gestión de talento humano

	Frecuencia	Porcentaje
Bajo	24	9,41
Medio	142	55,69
Alto	89	34,90
Total	**255**	**100,00**

Análisis e interpretación de resultados: Sobre la gestión de talento humano en la gestión de riesgos, el 55,69 % de los encuestados se encuentra en un nivel medio, mientras que el 34,90 % logró un nivel alto y el 9,41 % un nivel bajo. El hecho de que más de la mitad de los profesionales se ubique en un nivel intermedio indica que las organizaciones cuentan con equipos con cierta preparación, pero no necesariamente con experiencia robusta o formación constante en gestión de riesgos. Los resultados sugieren que, si bien hay capital humano disponible, se requiere fortalecer sus capacidades con programas más estructurados de capacitación técnica y herramientas especializadas.

Figura 12
Nivel de la dimensión Gestión de talento humano

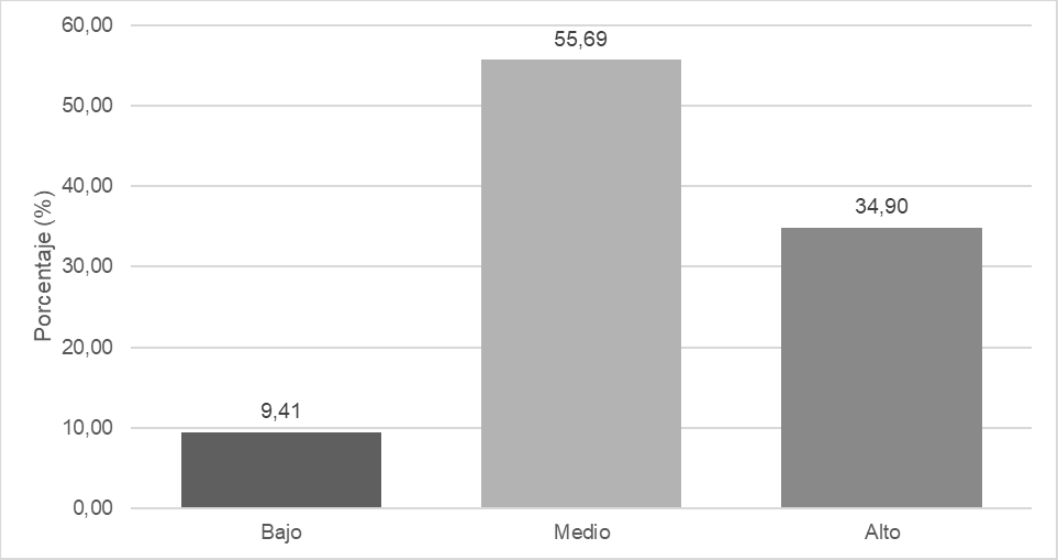
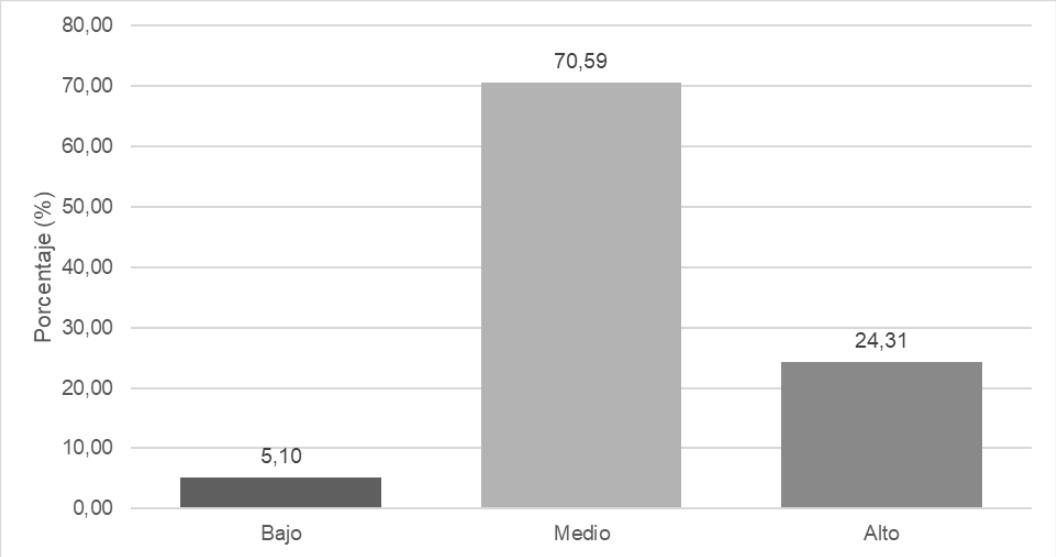


Tabla 14
Nivel de la variable Gestión de riesgos

	Frecuencia	Porcentaje
Bajo	13	5,10
Medio	180	70,59
Alto	62	24,31
Total	**255**	**100,00**

Análisis e interpretación de resultados: En términos globales de la variable gestión de riesgos, el 70,59 % de los encuestados se encuentra en un nivel medio, el 24,31 % en nivel alto y apenas el 5,10 % en nivel bajo. Esta distribución refleja que, si bien la gestión de riesgos se aplica en muchos casos, aún no se implementa con la consistencia o profundidad que se espera en entornos profesionales avanzados. La mayoría de los profesionales reconoce la existencia de procesos relacionados, pero con margen de mejora. Se concluye que hay una base funcional instalada, pero es necesario reforzar su aplicación mediante políticas, procesos y formación especializada.

Figura 13
Nivel de la variable Gestión de riesgos



4.1.2.2. Análisis descriptivo de la Calidad de software

Análisis e interpretación de los ítems

Tabla 15
El software desarrollado cumple con todos los requisitos funcionales definidos en la etapa de planificación

	Frecuencia	**Porcentaje**
Muy en desacuerdo	4	1,57
En desacuerdo	29	11,37
Neutral	49	19,22
De acuerdo	142	55,69
Muy de acuerdo	31	12,16
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 67,85 % indicó estar de acuerdo o muy de acuerdo en que el software desarrollado cumple con los requisitos funcionales definidos en la etapa de planificación. Un 19,22 % se mostró neutral y el 12,94 % expresó desacuerdo. Estos resultados reflejan una percepción positiva respecto al cumplimiento de lo planificado en los proyectos de software, sin embargo, aún existe un porcentaje relevante indica deficiencias en la documentación de requerimientos o en el control de calidad.

Figura 14
El software desarrollado cumple con todos los requisitos funcionales definidos en la etapa de planificación

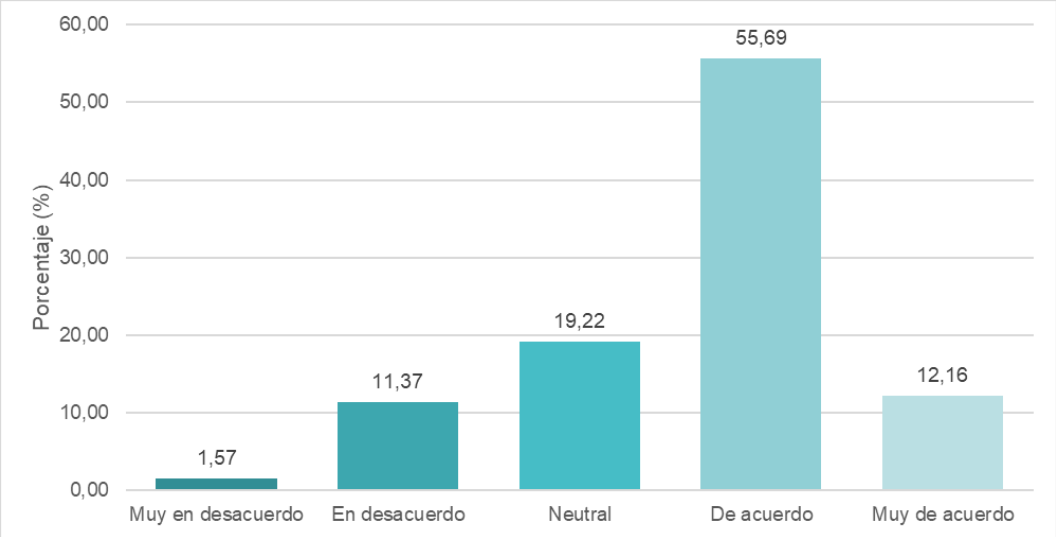


Tabla 16
Se realizan pruebas exhaustivas para validar que todas las funcionalidades cumplen con los objetivos establecidos

	Frecuencia	Porcentaje
Muy en desacuerdo	12	4,71
En desacuerdo	34	13,33
Neutral	73	28,63
De acuerdo	98	38,43
Muy de acuerdo	38	14,90
Total	**255**	**100,00**

Análisis e interpretación de resultados: Un 53,33 % de los participantes estuvo de acuerdo o muy de acuerdo en que se realizan pruebas exhaustivas para validar las funcionalidades. No obstante, el 28,63 % respondió neutral y un 18,04 % expresó desacuerdo. Esta variabilidad sugiere que, aunque más de la mitad percibe rigurosidad en las pruebas, aún hay espacios donde los procesos de validación no están estandarizados o no se comunican de forma clara. La alta proporción de respuestas neutrales podría reflejar una falta de conocimiento por parte del personal sobre los mecanismos internos de prueba o documentación insuficiente de los resultados.

Figura 15
Se realizan pruebas exhaustivas para validar que todas las funcionalidades cumplen con los objetivos establecidos

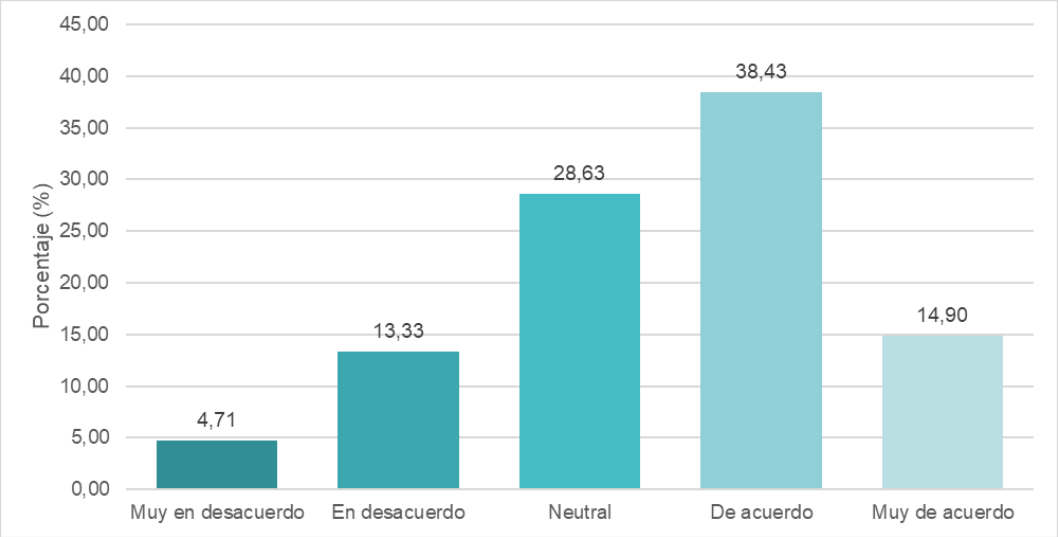


Tabla 17
Las funcionalidades del software son comprensibles y utilizables por los usuarios finales

	Frecuencia	Porcentaje
Muy en desacuerdo	4	1,57
En desacuerdo	24	9,41
Neutral	28	10,98
De acuerdo	159	62,35
Muy de acuerdo	40	15,69
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 77,64 % de los encuestados señaló estar de acuerdo o muy de acuerdo con que las funcionalidades del software son comprensibles y utilizables por los usuarios finales. Solo un 10,98 % se mantuvo en posición neutral y un 10,98 % expresó desacuerdo. Esta tendencia revela un aspecto positivo en la experiencia del usuario y en la calidad del diseño funcional. La alta percepción de claridad y facilidad de uso indica que se han incorporado criterios de usabilidad efectivos durante el diseño, desarrollo e implementación de los productos de software evaluados, lo cual favorece la aceptación por parte del usuario final.

Figura 16
Las funcionalidades del software son comprensibles y utilizables por los usuarios finales

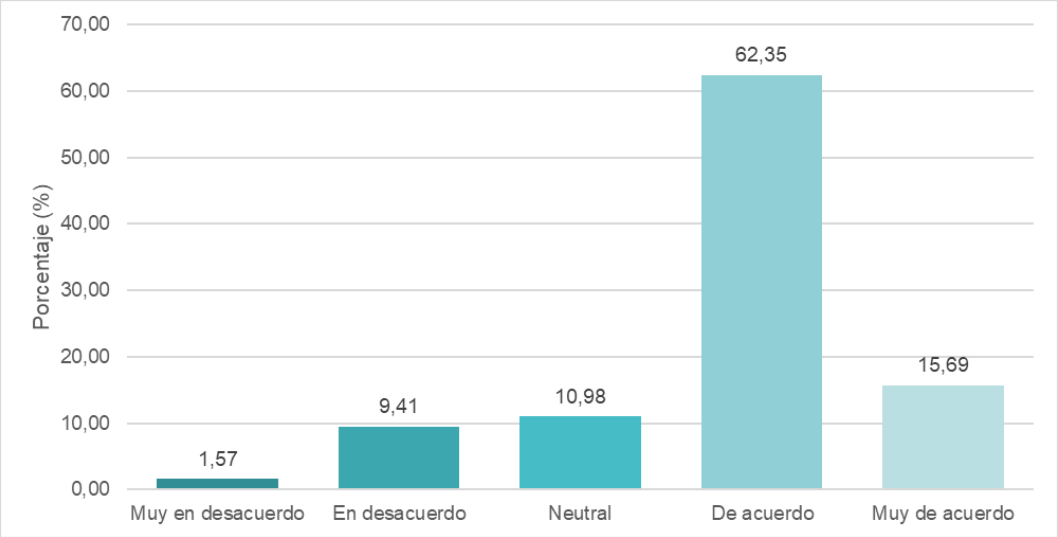


Tabla 18
El software opera de manera estable y confiable bajo las condiciones de uso previstas

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	12	4,71
Neutral	42	16,47
De acuerdo	150	58,82
Muy de acuerdo	42	16,47
Total	**255**	**100,00**

Análisis e interpretación de resultados: Según los datos, el 75,29 % de los encuestados estuvo de acuerdo o muy de acuerdo con que el software opera de manera estable y confiable bajo condiciones previstas. Un 16,47 % se mostró neutral y solo un 8,24 % expresó desacuerdo. Estos resultados indican que la mayoría de los productos desarrollados cumple con los criterios de fiabilidad esperados. No obstante, el grupo neutral podría estar asociado a usuarios con poca interacción técnica directa o sin acceso a métricas de comportamiento del software, lo cual plantea la necesidad de promover una cultura de seguimiento a indicadores de desempeño técnico.

Figura 17
El software opera de manera estable y confiable bajo las condiciones de uso previstas

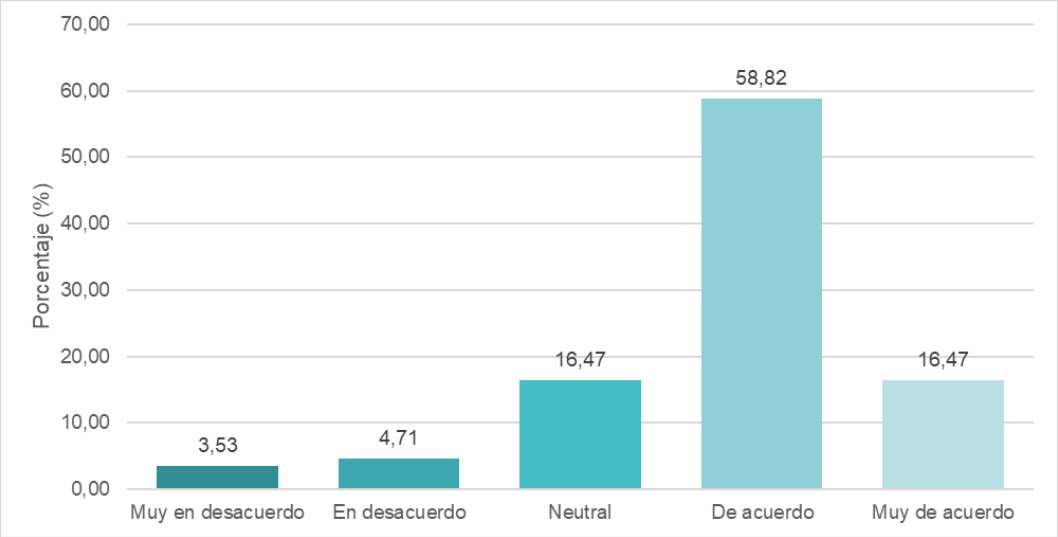


Tabla 19
Se realizan pruebas para garantizar que el software pueda recuperarse rápidamente en caso de fallos

	Frecuencia	Porcentaje
Muy en desacuerdo	15	5,88
En desacuerdo	14	5,49
Neutral	35	13,73
De acuerdo	145	56,86
Muy de acuerdo	46	18,04
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 74,90 % de los participantes considera que se realizan pruebas para garantizar la recuperación rápida ante fallos, mientras que un 13,73 % se mostró neutral y un 11,37 % expresó desacuerdo. Estos resultados revelan una alta percepción de cumplimiento en los planes de contingencia o en la implementación de soluciones resilientes ante errores. Aun así, el porcentaje de neutralidad y desacuerdo sugiere que en algunos entornos no se aplican o comunican estas estrategias de manera eficaz.

Figura 18
Se realizan pruebas para garantizar que el software pueda recuperarse rápidamente en caso de fallos

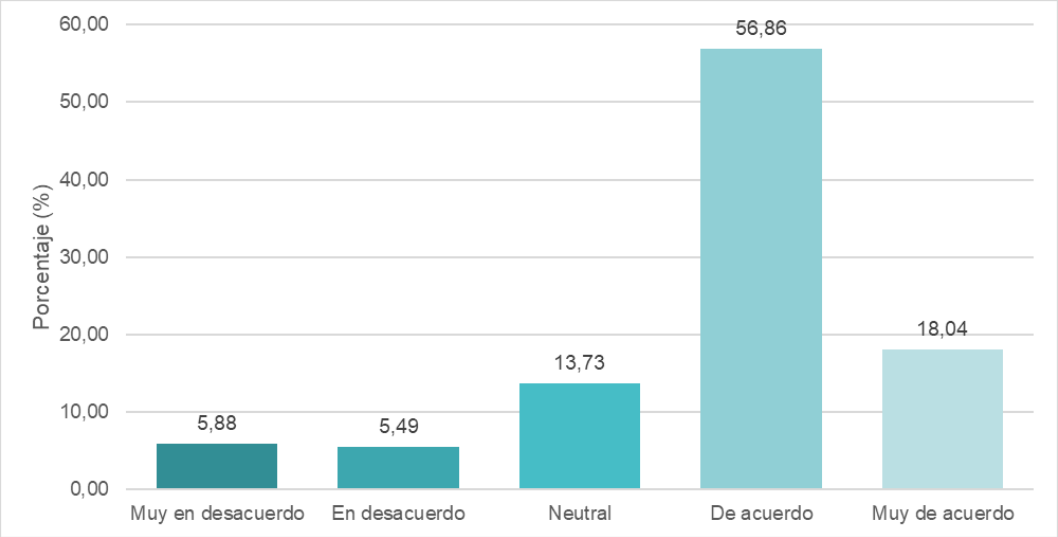


Tabla 20
La frecuencia de fallos en el software es baja durante las fases de prueba y uso operativo

	Frecuencia	Porcentaje
En desacuerdo	52	20,39
Neutral	69	27,06
De acuerdo	115	45,10
Muy de acuerdo	19	7,45
Total	**255**	**100,00**

Análisis e interpretación de resultados: En cuanto a la percepción sobre la frecuencia de fallos, el 52,55 % de los encuestados estuvo de acuerdo o muy de acuerdo en que esta es baja durante las fases de prueba y operación. Sin embargo, el 27,06 % optó por una respuesta neutral y el 20,39 % expresó desacuerdo. Esta distribución evidencia una percepción menos sólida en cuanto a la fiabilidad operativa del software. Aunque la mayoría muestra conformidad, los niveles de neutralidad y desacuerdo indican que existen áreas donde se experimentan errores frecuentes, lo que sugiere la necesidad de reforzar los procedimientos de prueba y corrección temprana.

Figura 19
La frecuencia de fallos en el software es baja durante las fases de prueba y uso operativo

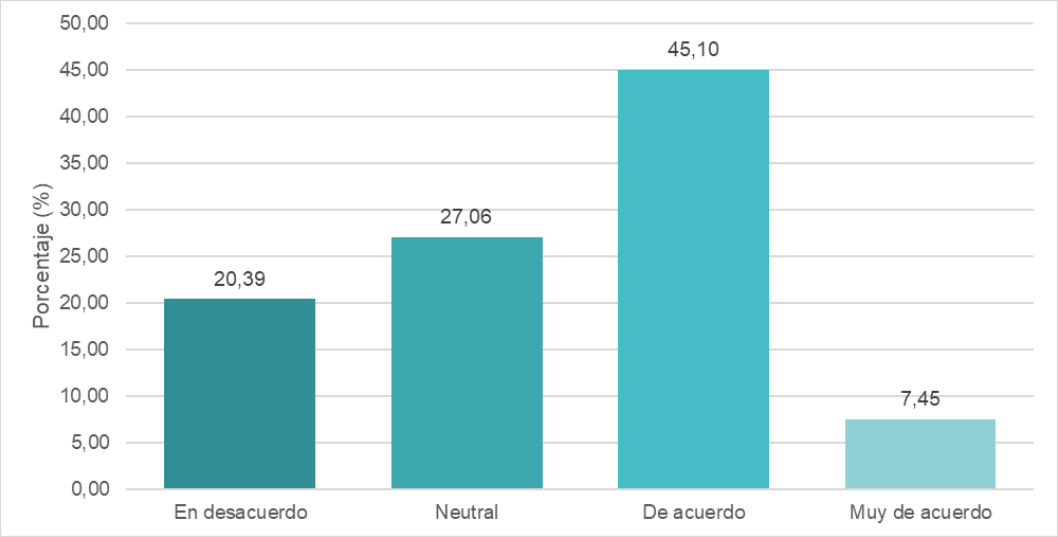


Tabla 21
El software responde de manera eficiente bajo las condiciones normales de uso

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	9	3,53
Neutral	44	17,25
De acuerdo	161	63,14
Muy de acuerdo	32	12,55
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 75,69 % de los encuestados afirmó que el software responde de manera eficiente bajo condiciones normales de uso. Solo un 17,25 % se mostró neutral y un 7,06 % expresó desacuerdo. Esta alta proporción de respuestas positivas sugiere que el desempeño del software es adecuado en términos de velocidad y consumo de recursos. No obstante, el porcentaje neutral podría estar asociado a la falta de métricas objetivas disponibles para los usuarios o a variaciones de rendimiento entre distintos entornos de operación. En conjunto, se observa un panorama favorable respecto a la eficiencia operativa de las soluciones implementadas.

Figura 20
El software responde de manera eficiente bajo las condiciones normales de uso

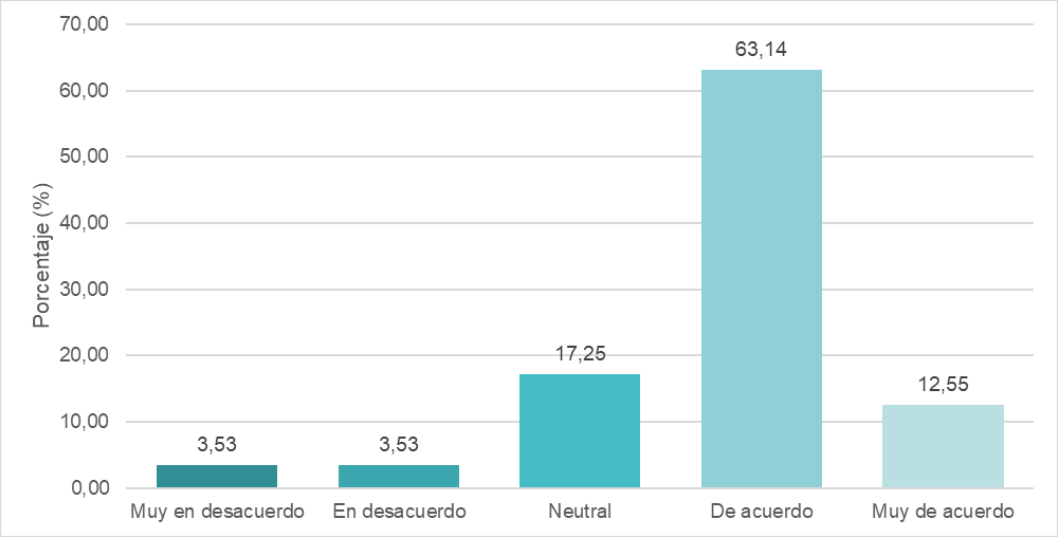


Tabla 22
Los recursos disponibles de hardware y software son utilizados de manera óptima por el sistema

	Frecuencia	Porcentaje
Muy en desacuerdo	4	1,57
En desacuerdo	27	10,59
Neutral	20	7,84
De acuerdo	189	74,12
Muy de acuerdo	15	5,88
Total	**255**	**100,00**

Análisis e interpretación de resultados: Los datos muestran que el 74,12 % de los encuestados está de acuerdo con que los recursos de hardware y software son utilizados de forma óptima por el sistema, mientras que un 5,88 % está muy de acuerdo. Solo un 10,59 % está en desacuerdo y un 1,57 % muy en desacuerdo. Esta tendencia refleja una percepción predominantemente positiva sobre la eficiencia con la que el sistema utiliza los recursos tecnológicos disponibles, lo que sugiere una adecuada configuración y gestión del entorno operativo. Sin embargo, se identifica un pequeño porcentaje crítico que podría estar relacionado con contextos específicos de infraestructura o mantenimiento.

Figura 21
Los recursos disponibles de hardware y software son utilizados de manera óptima por el sistema

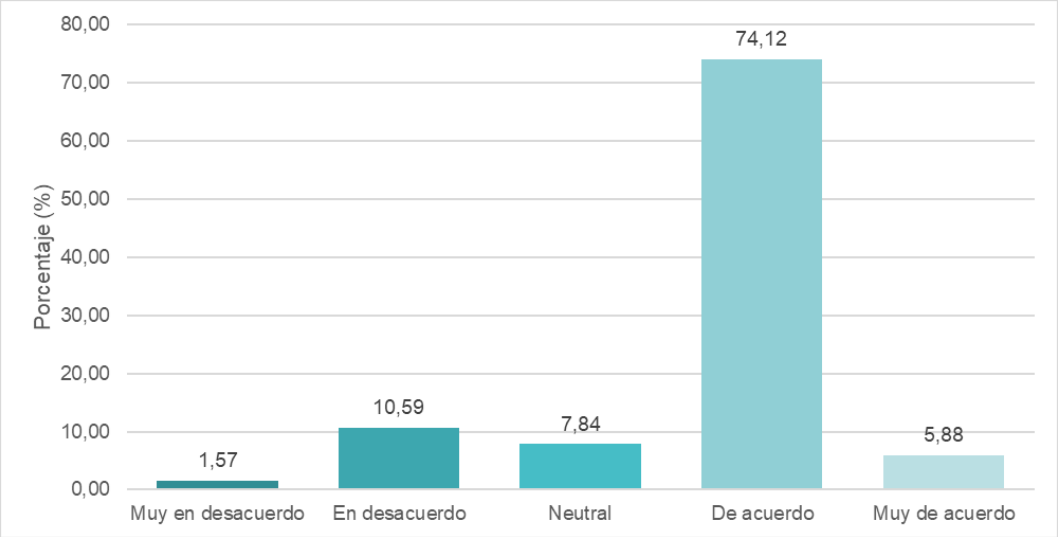


Tabla 23
El rendimiento del software se mantiene constante incluso bajo altas cargas de trabajo

	Frecuencia	Porcentaje
Muy en desacuerdo	9	3,53
En desacuerdo	39	15,29
Neutral	77	30,20
De acuerdo	130	50,98
Total	**255**	**100,00**

Análisis e interpretación de resultados: La mayoría de los encuestados, representando el 50,98 %, está de acuerdo con que el rendimiento del software se mantiene constante incluso bajo condiciones de carga elevadas, y un 3,53 % está muy de acuerdo. Por otro lado, el 15,29 % está en desacuerdo y el 3,53 % muy en desacuerdo. Estos resultados sugieren que, en general, el software responde de forma estable bajo presión operativa, aunque existe un grupo minoritario que manifiesta deficiencias de rendimiento bajo ciertas circunstancias, lo cual debe ser considerado para mejoras en eficiencia y escalabilidad.

Figura 22
El rendimiento del software se mantiene constante incluso bajo altas cargas de trabajo

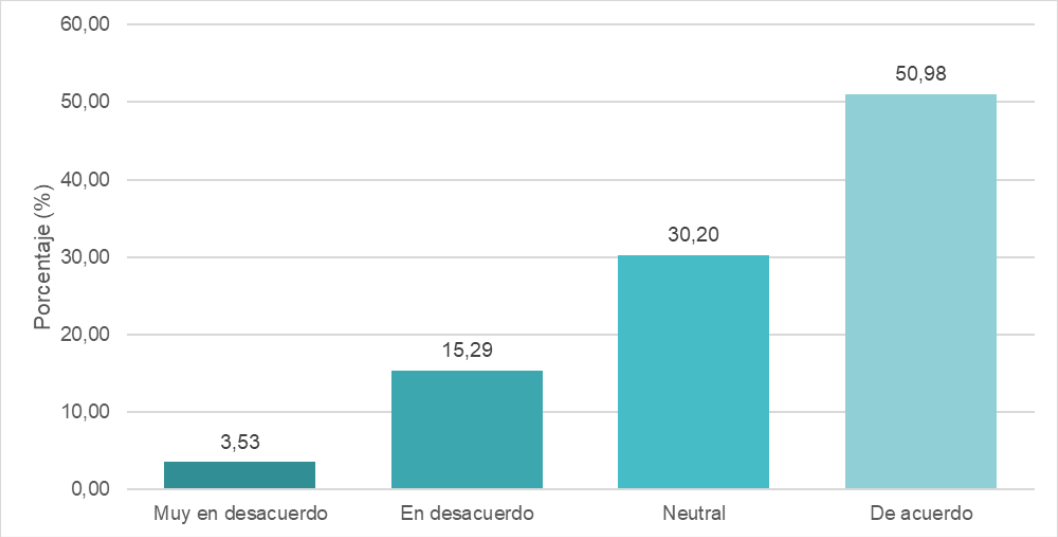


Tabla 24
El software permite realizar modificaciones sin afectar su funcionamiento general

	Frecuencia	**Porcentaje**
Muy en desacuerdo	10	3,92
En desacuerdo	37	14,51
Neutral	69	27,06
De acuerdo	129	50,59
Muy de acuerdo	10	3,92
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 50,59 % de los profesionales encuestados indica estar de acuerdo con que el software permite realizar modificaciones sin afectar su funcionamiento general, mientras que solo un 3,92 % está muy de acuerdo. Un 27,06 % se mantiene neutral, lo que indica cierta reserva respecto a la facilidad de mantenimiento. Estos datos reflejan que la mayoría percibe el software como adaptable, aunque se evidencian áreas de mejora en cuanto a la flexibilidad del diseño y la implementación de cambios sin impactos significativos.

Figura 23
El software permite realizar modificaciones sin afectar su funcionamiento general

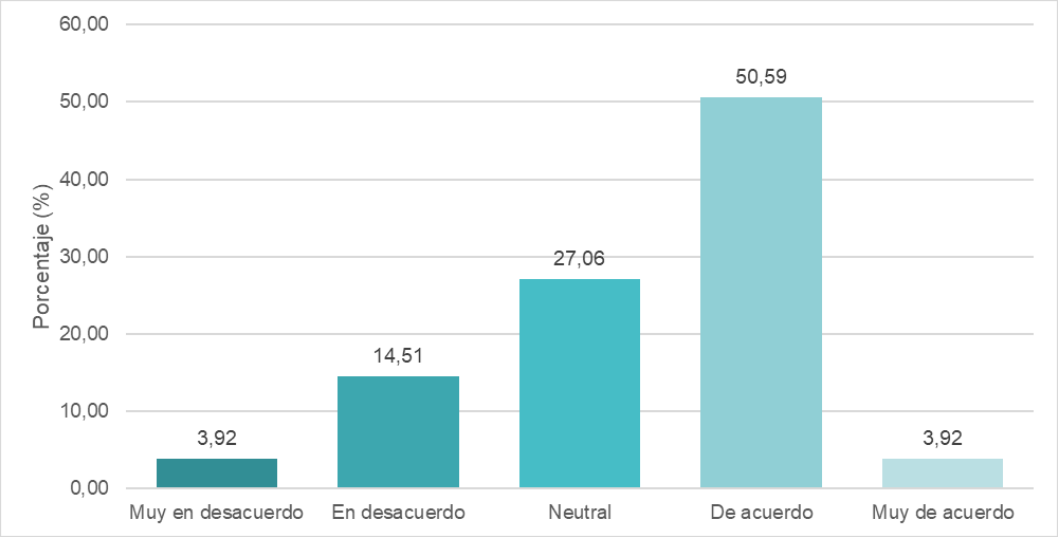


Tabla 25
Las tareas de mantenimiento son realizadas en un tiempo razonable

	Frecuencia	**Porcentaje**
Muy en desacuerdo	4	1,57
En desacuerdo	34	13,33
Neutral	53	20,78
De acuerdo	140	54,90
Muy de acuerdo	24	9,41
Total	**255**	**100,00**

Análisis e interpretación de resultados: Según los resultados, el 54,90 % de los encuestados está de acuerdo con que las tareas de mantenimiento se realizan en un tiempo razonable, con un 9,41 % muy de acuerdo. El 20,78 % se mantiene neutral, y un 13,33 % está en desacuerdo. Esto revela una percepción mayoritariamente favorable respecto a los tiempos de mantenimiento, lo que implica un nivel aceptable de eficiencia en las intervenciones técnicas, aunque el porcentaje en desacuerdo sugiere oportunidades para optimizar los procesos de soporte técnico.

Figura 24
Las tareas de mantenimiento son realizadas en un tiempo razonable

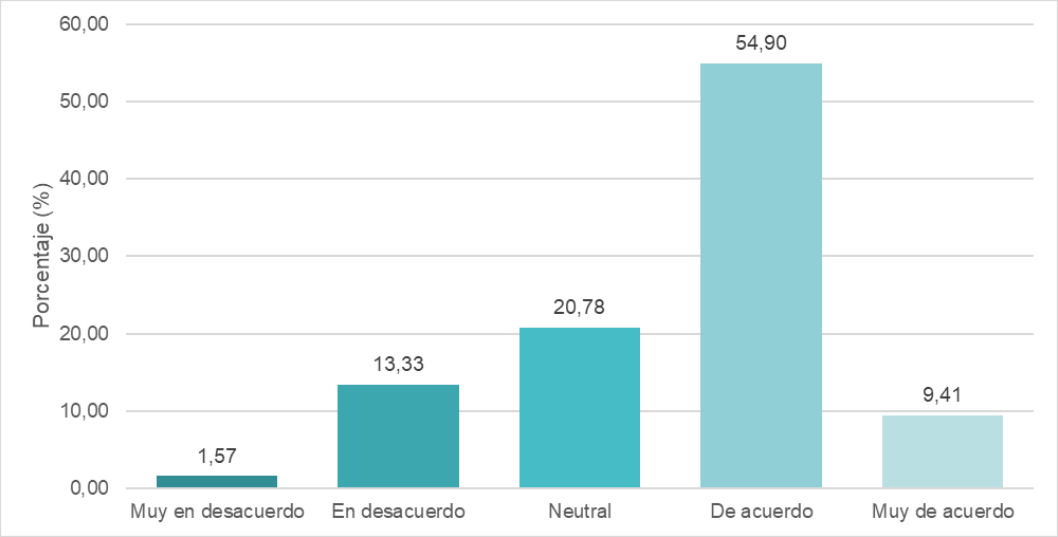
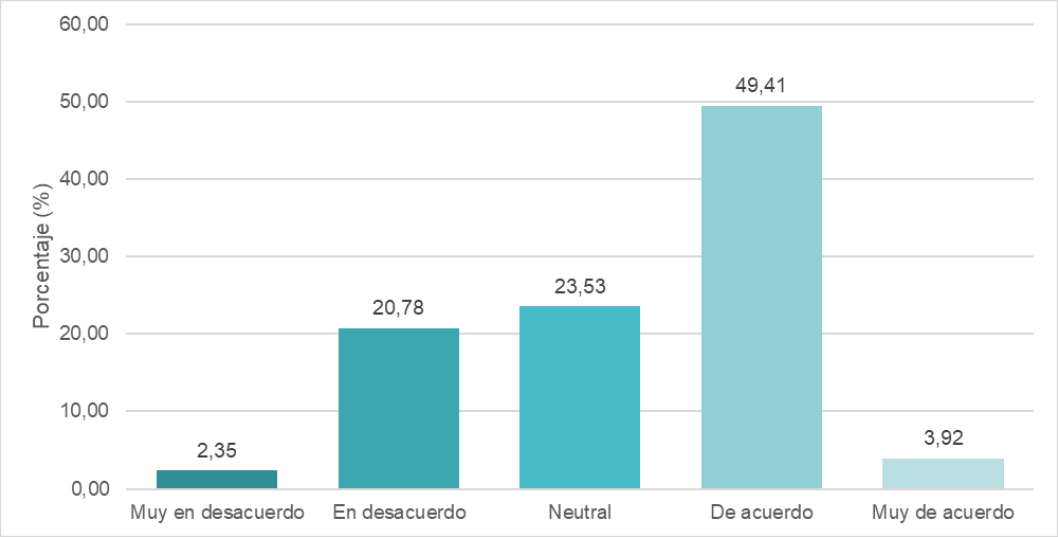


Tabla 26
El software se adapta de manera flexible a los cambios en los requisitos del proyecto

	Frecuencia	Porcentaje
Muy en desacuerdo	6	2,35
En desacuerdo	53	20,78
Neutral	60	23,53
De acuerdo	126	49,41
Muy de acuerdo	10	3,92
Total	**255**	**100,00**

Análisis e interpretación de resultados: Un 49,41 % de los encuestados considera que el software se adapta de manera flexible a los cambios en los requisitos del proyecto, mientras que solo un 3,92 % está muy de acuerdo. En contraste, el 20,78 % está en desacuerdo y un 2,35 % muy en desacuerdo. Esto evidencia que, aunque la mayoría aprecia cierta adaptabilidad, existe una proporción significativa de profesionales que perciben limitaciones en la flexibilidad del software, lo que podría impactar la capacidad de respuesta ante necesidades cambiantes del entorno o del cliente.

Figura 25
El software se adapta de manera flexible a los cambios en los requisitos del proyecto



Análisis e interpretación de la variable y dimensiones categorizadas

Tabla 27
Nivel de la dimensión Funcionalidad

	Frecuencia	**Porcentaje**
Bajo	26	10,20
Medio	112	43,92
Alto	117	45,88
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 45,88 % de los encuestados considera que el software desarrollado presenta un nivel alto de funcionalidad, mientras que un 43,92 % lo evalúa como medio y solo un 10,20 % lo califica como bajo. Esta distribución sugiere que la mayoría de los ingenieros colegiados perciben que el software cumple con los requisitos funcionales establecidos, garantizando que las funcionalidades son útiles y coherentes con las necesidades del usuario. Sin embargo, aún existe un porcentaje que indica posibles áreas de mejora para lograr una funcionalidad óptima en todos los proyectos.

Figura 26
Nivel de la dimensión Funcionalidad

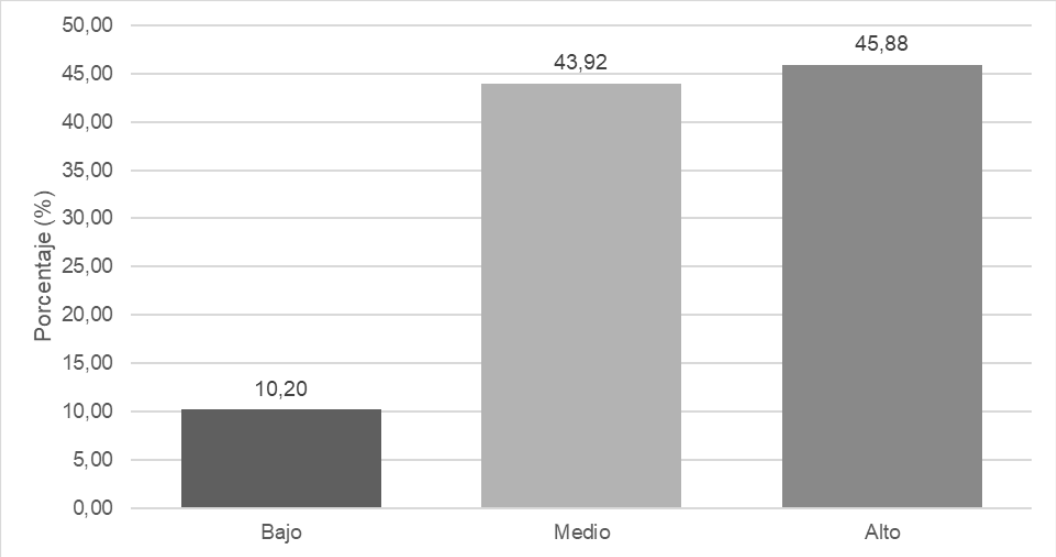


Tabla 28
Nivel de la dimensión Fiabilidad

	Frecuencia	Porcentaje
Bajo	15	5,88
Medio	138	54,12
Alto	102	40,00
Total	**255**	**100,00**

Análisis e interpretación de resultados: Según los resultados, el 40,00 % de los encuestados considera que la fiabilidad del software es alta, el 54,12 % opina que es media y el 5,88 % la califica como baja. Estos datos indican que, aunque una parte importante de los sistemas es estable y confiable, más de la mitad de los desarrolladores aún perciben una fiabilidad moderada. Este nivel medio puede implicar que el software cumple de manera aceptable, pero no constante, con los estándares esperados de operación sin errores.

Figura 27
Nivel de la dimensión Fiabilidad

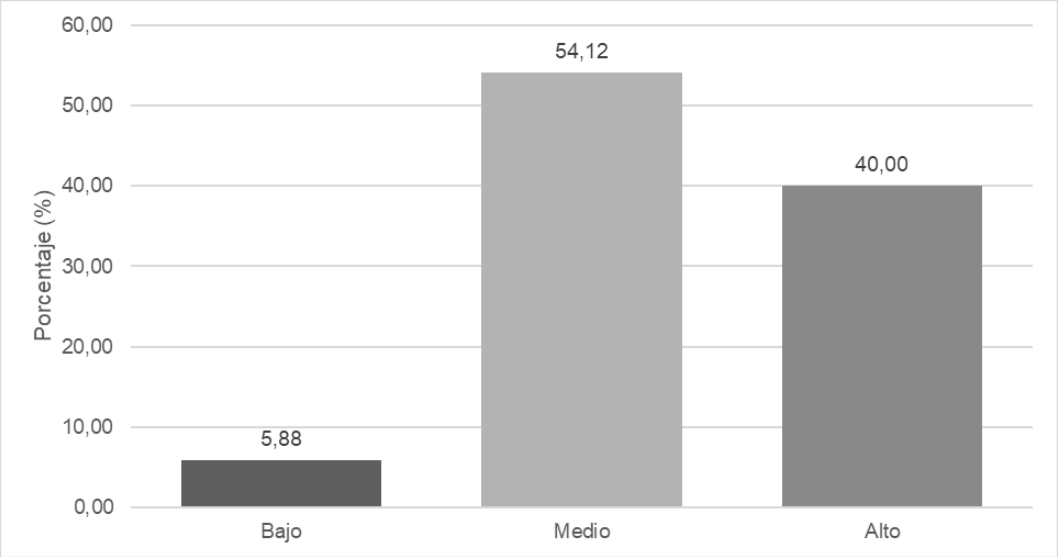


Tabla 29
Nivel de la dimensión Rendimiento

	Frecuencia	Porcentaje
Bajo	18	7,06
Medio	133	52,16
Alto	104	40,78
Total	**255**	**100,00**

Análisis e interpretación de resultados: En esta dimensión, el 40,78 % de los profesionales encuestados considera que el software tiene un rendimiento alto, mientras que el 52,16 % lo percibe como medio y solo el 7,06 % lo califica como bajo. Esto refleja que, si bien el sistema responde de forma eficiente en la mayoría de los casos, aún existen oportunidades de optimización en entornos con cargas elevadas o procesos complejos que podrían afectar la experiencia del usuario.

Figura 28
Nivel de la dimensión Rendimiento

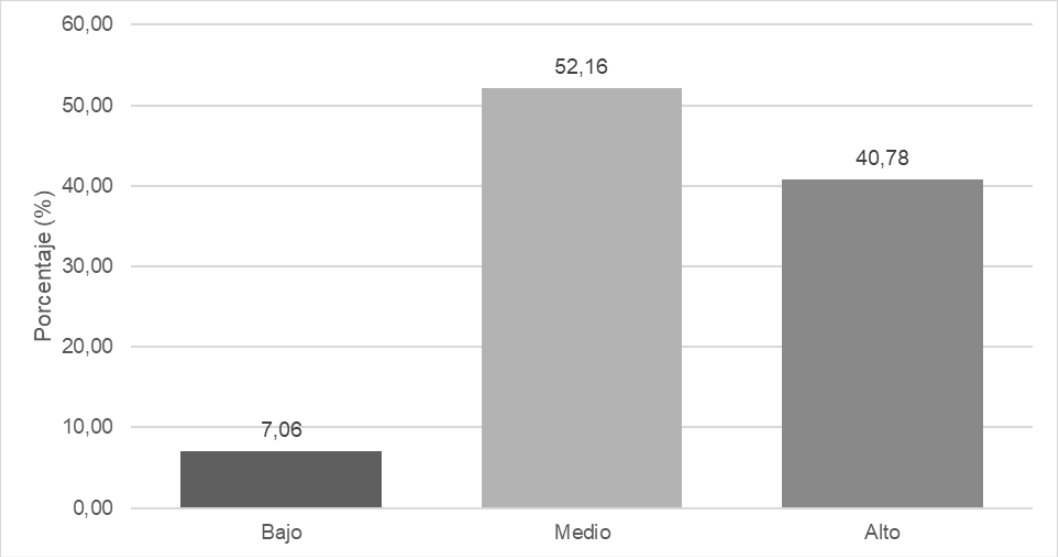


Tabla 30
Nivel de la dimensión Mantenibilidad

	Frecuencia	Porcentaje
Bajo	38	14,90
Medio	142	55,69
Alto	75	29,41
Total	**255**	**100,00**

Análisis e interpretación de resultados: El 29,41 % de los encuestados considera que el software tiene alta mantenibilidad, el 55,69 % opina que esta es media y el 14,90 % señala un nivel bajo. Estos resultados sugieren que el mantenimiento del software es viable, pero no siempre eficiente o ágil, lo que puede afectar los tiempos de respuesta ante incidentes o cambios en los requerimientos. Mejorar esta dimensión facilitaría la evolución y adaptación del software a nuevas necesidades del entorno.

Figura 29
Nivel de la dimensión Mantenibilidad

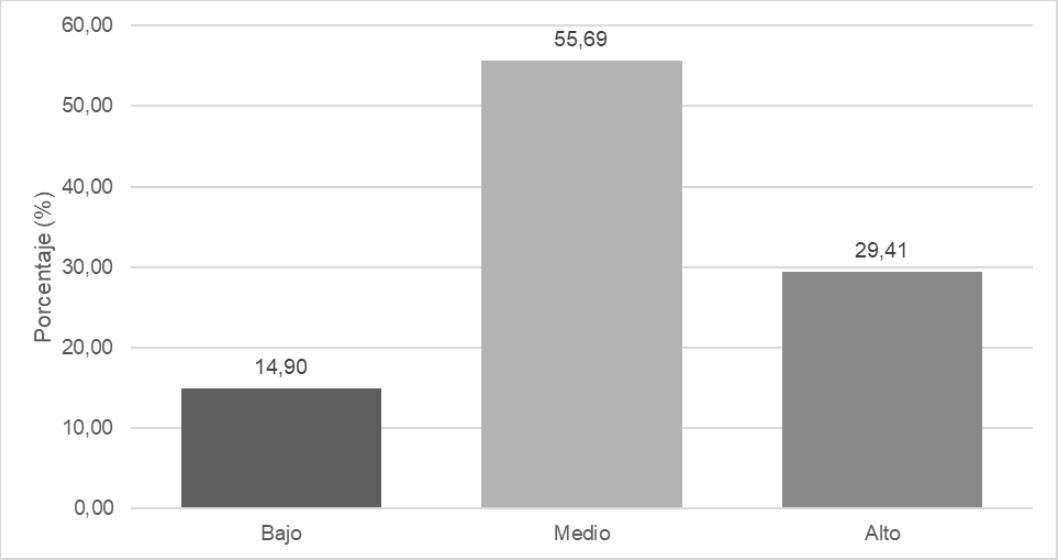
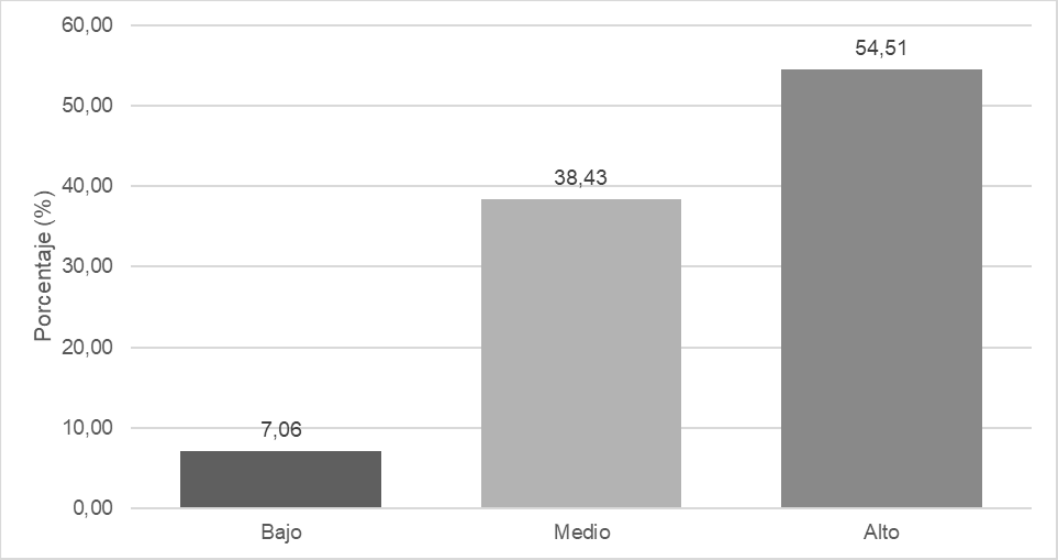


Tabla 31
Nivel de la variable Calidad de software

	Frecuencia	Porcentaje
Bajo	18	7,06
Medio	98	38,43
Alto	139	54,51
Total	**255**	**100,00**

Análisis e interpretación de resultados: En términos generales, el 54,51 % de los ingenieros califica la calidad del software como alta, el 38,43 % la considera media y solo el 7,06 % la percibe como baja. Esta distribución indica una percepción predominantemente positiva de la calidad del software desarrollado por los profesionales del CIP de Tacna. No obstante, la proporción media también alerta sobre la necesidad de fortalecer algunas áreas clave para asegurar estándares más altos de calidad a nivel integral.

Figura 30
Nivel de la variable Calidad de software



4.2. Análisis del impacto

4.2.1. Prueba estadística

Tabla 32
Análisis de la distribución de datos de las variables

	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Estadístico	gl	Sig.	Estadístico	gl	Sig.
Gestión de riesgos	0,116	255	0,000	0,965	255	0,000
Calidad de software	0,240	255	0,000	0,862	255	0,000

Análisis e interpretación de resultados: Para ambas variables analizadas, los valores de significancia (Sig.) son menores a 0,05 (p = 0,000), en el caso de la variable gestión de riesgos, se observa un estadístico de 0,116 en Kolmogorov-Smirnov y 0,965 en Shapiro-Wilk, mientras que para la variable calidad de software los valores fueron de 0,240 y 0,862 respectivamente. Dado que en ambos casos los niveles de significancia son inferiores al umbral establecido (p < 0,05), se rechaza la hipótesis nula de normalidad.

4.2.2. Comprobación de hipótesis

4.2.2.1. Comprobación de hipótesis específica 1

Formulación de hipótesis:

H1: La gestión de riesgos se relaciona significativamente con la funcionalidad del software realizado por profesionales del CIP, Tacna – 2024.

H0: La gestión de riesgos no se relaciona significativamente con la funcionalidad del software realizado por profesionales del CIP, Tacna – 2024.

Nivel de significancia: 0,05 = 5%

Condición de aceptación o rechazo:

Si, el valor P <= 0,05 → Rechazar la hipótesis nula (H0)

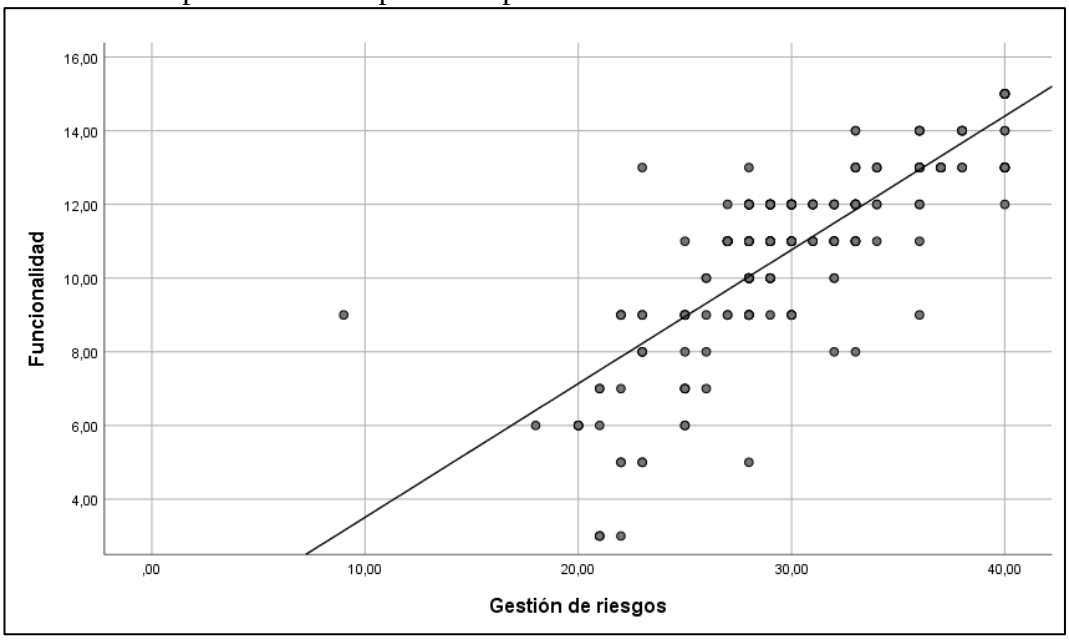
Si, el valor P > 0,05 → Aceptar la hipótesis nula (H0)

Resultados:

Tabla 33
Comprobación de la hipótesis específica 1

			Gestión de riesgos	Funcionalidad
Rho de Spearman	Gestión de riesgos	Coeficiente de correlación	1,000	,820**
		Sig. (bilateral)		0,000
		N	255	255
	Funcionalidad	Coeficiente de correlación	,820**	1,000
		Sig. (bilateral)	0,000	
		N	255	255

Figura 31
Gráfico de dispersión de la hipótesis específica 1



Interpretación de resultados:

Con base en la prueba de correlación de Spearman, se obtiene un coeficiente de correlación de 0,820, lo cual indica una correlación positiva alta entre la gestión de riesgos y la funcionalidad del software. Asimismo, el valor de significancia bilateral es de 0,000, que es menor al nivel crítico de 0,05, lo que lleva a rechazar la hipótesis nula (H_0) y aceptar la hipótesis de investigación (H_1). Esto evidencia que una mejor gestión de riesgos se asocia significativamente con una mayor funcionalidad del software desarrollado por los profesionales del CIP en Tacna, 2024. El gráfico de dispersión respalda esta conclusión, mostrando una tendencia ascendente clara entre ambas variables. Por tanto, se concluye que, al mejorar los procesos de identificación, documentación y mitigación de riesgos, se incrementa la capacidad del software para cumplir con los requisitos funcionales establecidos.

4.2.2.2. Comprobación de hipótesis específica 2

Formulación de hipótesis:

H1: La gestión de riesgos se relaciona significativamente con la fiabilidad del software realizado por profesionales del CIP, Tacna – 2024.

H0: La gestión de riesgos no se relaciona significativamente con la fiabilidad del software realizado por profesionales del CIP, Tacna – 2024.

Nivel de significancia: 0,05 = 5%

Condición de aceptación o rechazo:

Si, el valor P <= 0,05 → Rechazar la hipótesis nula (H0)

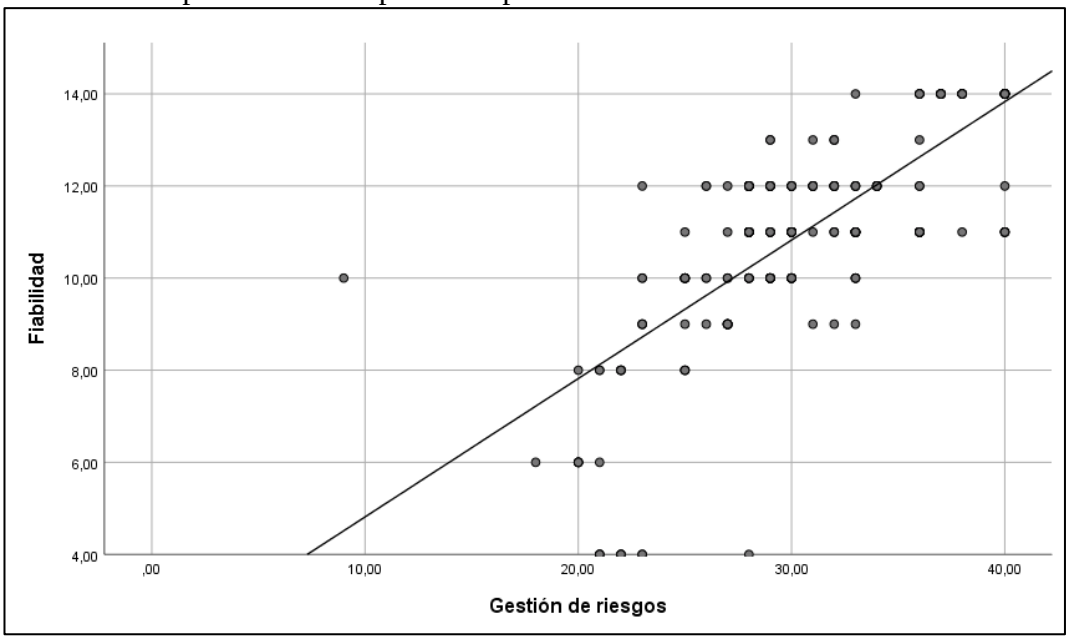
Si, el valor P > 0,05 → Aceptar la hipótesis nula (H0)

Resultados:

Tabla 34
Comprobación de la hipótesis específica 2

			Gestión de riesgos	Fiabilidad
Rho de Spearman	Gestión de riesgos	Coeficiente de correlación	1,000	,686**
		Sig. (bilateral)		0,000
		N	255	255
	Fiabilidad	Coeficiente de correlación	,686**	1,000
		Sig. (bilateral)	0,000	
		N	255	255

Figura 32
Gráfico de dispersión de la hipótesis específica 2



Interpretación de resultados:

Según la tabla, el coeficiente de correlación de Spearman entre la gestión de riesgos y la fiabilidad del software es de 0,686, lo que indica una correlación positiva alta entre ambas variables. El valor de significancia bilateral es 0,000, inferior al umbral de 0,05, lo que permite rechazar la hipótesis nula (H_0) y aceptar la hipótesis alterna (H_1). Esto confirma que una adecuada gestión de riesgos se asocia significativamente con un mayor nivel de fiabilidad en el software desarrollado por los profesionales del CIP en Tacna. Además, el gráfico de dispersión refuerza esta relación, revelando una clara tendencia ascendente. En consecuencia, se deduce que, al implementar correctamente los procesos de identificación, análisis y seguimiento de riesgos, se logra reducir la tasa de fallos y garantizar un funcionamiento más confiable del sistema de software.

4.2.2.3. Comprobación de hipótesis específica 3

Formulación de hipótesis:

H1: La gestión de riesgos se relaciona significativamente con el rendimiento del software realizado por profesionales del CIP, Tacna – 2024.

H0: La gestión de riesgos no se relaciona significativamente con el rendimiento del software realizado por profesionales del CIP, Tacna – 2024.

Nivel de significancia: 0,05 = 5%

Condición de aceptación o rechazo:

Si, el valor P <= 0,05 → Rechazar la hipótesis nula (H0)

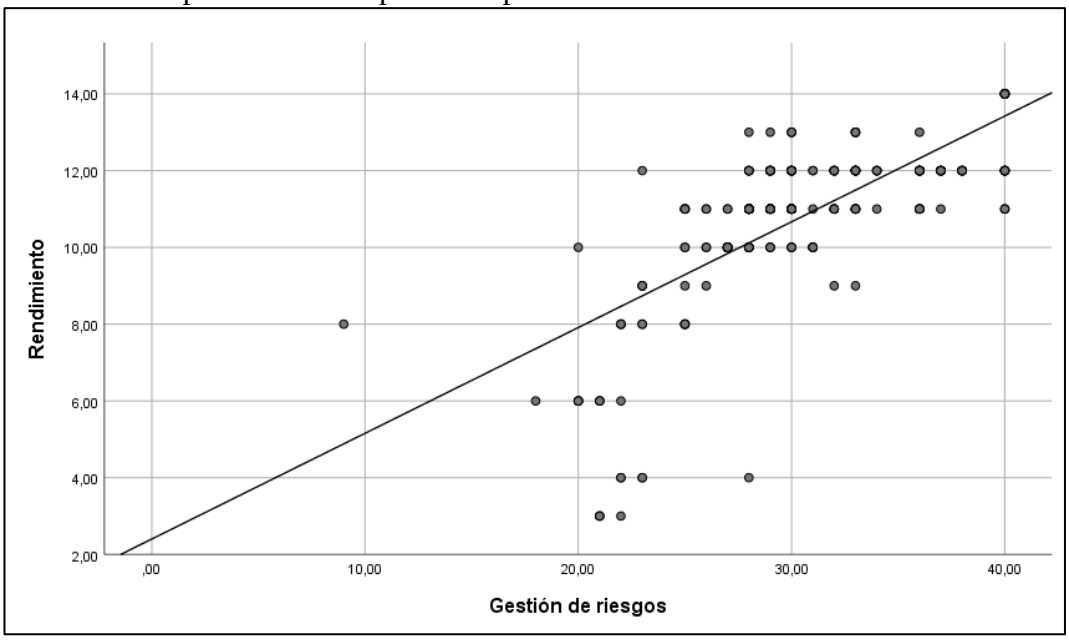
Si, el valor P > 0,05 → Aceptar la hipótesis nula (H0)

Resultados:

Tabla 35
Comprobación de la hipótesis específica 3

			Gestión de riesgos	Rendimiento
Rho de Spearman	Gestión de riesgos	Coeficiente de correlación	1,000	,728**
		Sig. (bilateral)		0,000
		N	255	255
	Rendimiento	Coeficiente de correlación	,728**	1,000
		Sig. (bilateral)	0,000	
		N	255	255

Figura 33
Gráfico de dispersión de la hipótesis específica 3



Interpretación de resultados:

En la tabla se observa que el coeficiente de correlación de Spearman entre la gestión de riesgos y el rendimiento del software es de 0,728, lo cual representa una correlación positiva alta. Asimismo, el valor de significancia bilateral es 0,000, menor que el nivel crítico de 0,05, lo que permite rechazar la hipótesis nula (H_0) y aceptar la hipótesis alterna (H_1). Esto indica que una adecuada gestión de riesgos se relaciona significativamente con un mejor rendimiento del software elaborado por los profesionales del CIP en Tacna.

El gráfico de dispersión respalda visualmente este resultado, evidenciando una clara tendencia lineal ascendente. En consecuencia, se concluye que la implementación de prácticas efectivas para identificar, mitigar y supervisar riesgos en los proyectos contribuye a mejorar la capacidad del software para operar eficientemente bajo diversas condiciones, incluso en escenarios de alta demanda operativa.

4.2.2.4. Comprobación de hipótesis específica 4

Formulación de hipótesis:

H1: La gestión de riesgos se relaciona significativamente con la mantenibilidad del software realizado por profesionales del CIP, Tacna – 2024.

H0: La gestión de riesgos no se relaciona significativamente con la mantenibilidad del software realizado por profesionales del CIP, Tacna – 2024.

Nivel de significancia: 0,05 = 5%

Condición de aceptación o rechazo:

Si, el valor P <= 0,05 → Rechazar la hipótesis nula (H0)

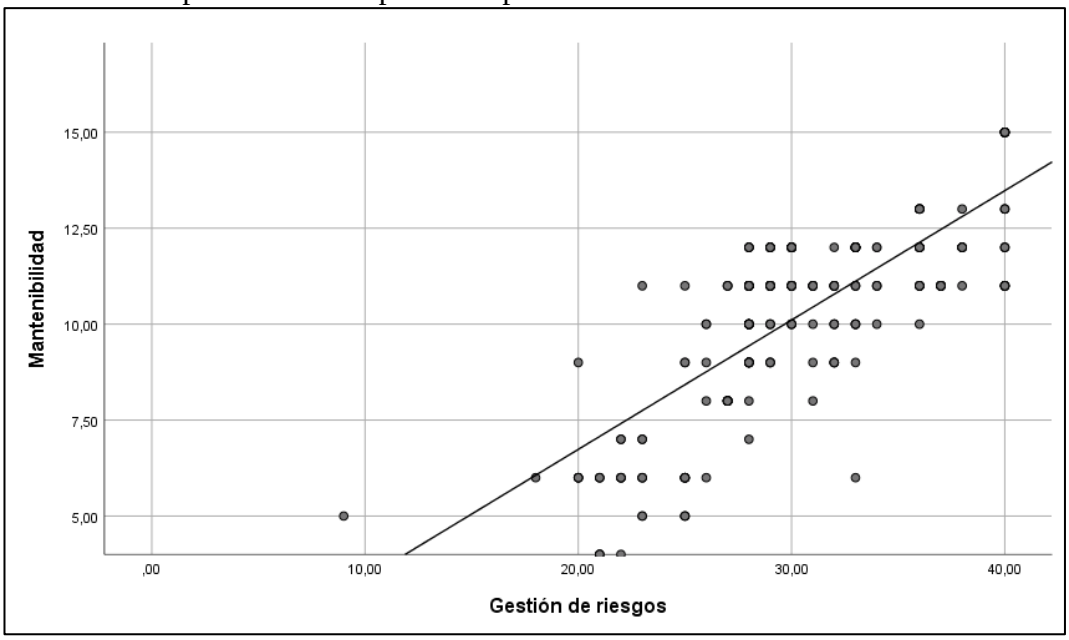
Si, el valor P > 0,05 → Aceptar la hipótesis nula (H0)

Resultados:

Tabla 36
Comprobación de la hipótesis específica 4

			Gestión de riesgos	Mantenibilidad
Rho de Spearman	Gestión de riesgos	Coeficiente de correlación	1,000	,715**
		Sig. (bilateral)		0,000
		N	255	255
	Mantenibilidad	Coeficiente de correlación	,715**	1,000
		Sig. (bilateral)	0,000	
		N	255	255

Figura 34
Gráfico de dispersión de la hipótesis específica 4



Interpretación de resultados:

En la tabla se muestra que el coeficiente de correlación de Spearman entre la gestión de riesgos y la mantenibilidad del software es de 0,715, lo que representa una correlación positiva alta. Además, el valor de significancia (Sig. = 0,000) es menor al nivel de significancia de 0,05, lo cual permite rechazar la hipótesis nula (H_0) y aceptar la hipótesis alterna (H_1). Esto indica que la gestión de riesgos se relaciona significativamente con la mantenibilidad del software realizado por profesionales del CIP en Tacna durante el año 2024.

El gráfico de dispersión respalda este hallazgo, al evidenciar una tendencia lineal positiva entre ambas variables. Por lo tanto, se concluye que una adecuada identificación, documentación y control de riesgos contribuye a que el software desarrollado sea más fácil de modificar, actualizar y mantener sin comprometer su funcionamiento, reduciendo los tiempos y costos de mantenimiento.

4.2.2.5. Comprobación de hipótesis general

Formulación de hipótesis:

H1: La gestión de riesgos tiene una relación significativa con la calidad del software realizado por profesionales del CIP, Tacna – 2024.

H0: La gestión de riesgos no tiene una relación significativa con la calidad del software realizado por profesionales del CIP, Tacna – 2024.

Nivel de significancia: 0,05 = 5%

Condición de aceptación o rechazo:

Si, el valor P <= 0,05 → Rechazar la hipótesis nula (H0)

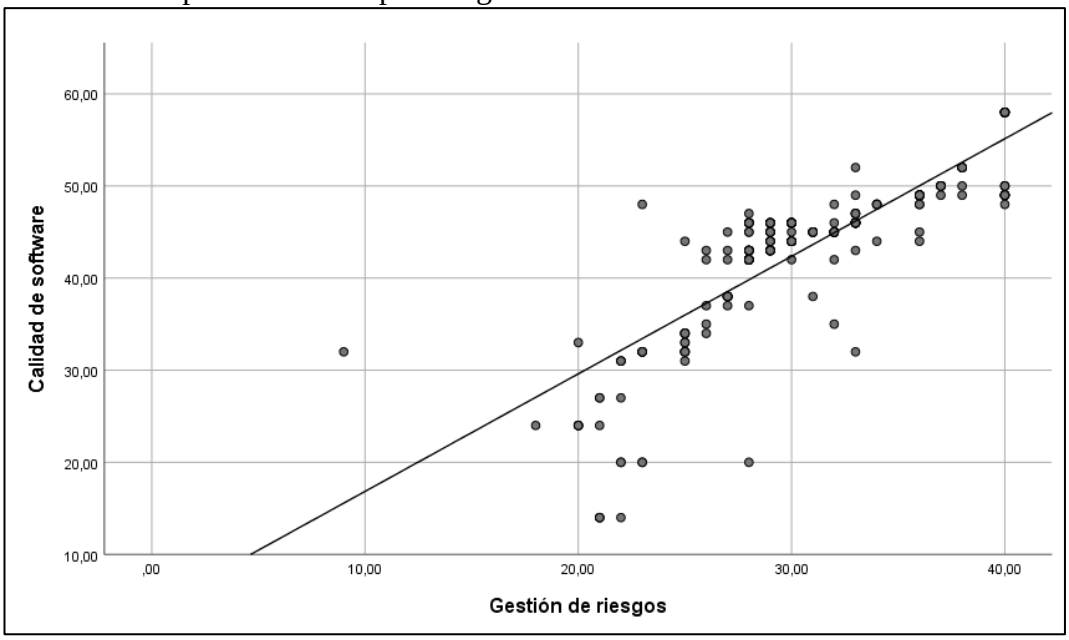
Si, el valor P > 0,05 → Aceptar la hipótesis nula (H0)

Resultados:

Tabla 37
Comprobación de la hipótesis general

			Gestión de riesgos	**Calidad de software**
Rho de Spearman	Gestión de riesgos	Coeficiente de correlación	1,000	,884**
		Sig. (bilateral)		0,000
		N	255	255
	Calidad de software	Coeficiente de correlación	,884**	1,000
		Sig. (bilateral)	0,000	
		N	255	255

Figura 35
Gráfico de dispersión de la hipótesis general



Interpretación de resultados:

Según la tabla, el coeficiente de correlación de Spearman entre la gestión de riesgos y la calidad del software es de 0,884, lo que indica una correlación positiva muy alta. El valor de significancia bilateral es 0,000, menor al umbral de 0,05, lo que permite rechazar la hipótesis nula (H_0) y aceptar la hipótesis alterna (H_1). Esto confirma que existe una relación significativa entre la gestión de riesgos y la calidad del software desarrollado por profesionales del CIP en Tacna durante el año 2024.

La figura refuerza esta conclusión mediante un gráfico de dispersión que muestra una fuerte tendencia lineal positiva. En términos prácticos, esto sugiere que una adecuada gestión de riesgos —incluyendo la identificación, documentación, seguimiento y control— incide directamente en mejores niveles de funcionalidad, fiabilidad, rendimiento y mantenibilidad del software entregado, lo cual impacta positivamente en su calidad final.

4.3. Interpretación de resultados

En primera instancia, con respecto a la relación general entre las variables, en la investigación se determinó un coeficiente de correlación de Rho = 0.884, lo

que representa una asociación positiva muy alta entre la gestión de riesgos y la calidad del software en los profesionales del CIP Tacna. Este hallazgo es significativamente superior al reporte de Paucar et al. (2021), quien en su estudio con ingenieros del CIP Lima encontró una correlación de Rho = 0.634. Mientras que en la capital se observó una relación moderada, en este estudio se puede constatar que la dependencia entre ambos factores es crítica en el contexto local. Al respecto, coincido con la visión de Pressman (2010), quien define la gestión de riesgos como una "actividad sombrilla" esencial; el autor advierte que "si no se atacan activamente los riesgos, ellos lo atacarán a usted", lo cual refuerza mi conclusión de que la calidad no es un accidente sino el resultado de una estrategia proactiva de mitigación.

En cuanto a la dimensión de gestión de recursos financieros, se encontró que esta dimensión revela una tendencia positiva, aunque predominante en la categoría media, lo cual indica que la planificación financiera está presente, pero con limitaciones para alcanzar la efectividad deseada. Este resultado guarda una estrecha coherencia con lo expuesto por Paucar et al. (2021), cuya investigación también destacó que la planificación financiera influye de manera determinante en el cumplimiento de los objetivos de calidad. Asimismo, este resultado guarda relación con la investigación de Valverde Flores (2022), quien sostiene que la aplicación de metodologías rigurosas permite tratar las amenazas de manera efectiva, pero recalca que esto requiere necesariamente la asignación de recursos humanos y financieros para implementar salvaguardas y planes de contingencia. Sobre este punto, me remito a Pressman (2010), quien sostiene que la gestión de proyectos debe equilibrar las "4 P" (personal, producto, proceso y proyecto), señalando que una asignación de recursos realista y un control de riesgos financieros son requisitos indispensables para evitar la degradación del proyecto.

Sobre el talento humano, se puede identificar que más de la mitad de los profesionales se ubican en un nivel intermedio indicando que las organizaciones cuentan con equipos con cierta preparación. Este valor se alinea con las preocupaciones de Afshari y Gandomani (2022), quienes en su estudio sobre

metodologías híbridas subrayan que la falta de formación específica en riesgos compromete la integridad del sistema. En el análisis del CIP Tacna, donde el 55.69% del personal se ubica en un nivel medio de competencia. Como refuerzo teórico a este hallazgo, Pressman (2010) afirma contundentemente que "el factor humano es el elemento más importante en la ingeniería de software", argumentando que un equipo capacitado no solo escribe código, sino que aplica el rigor necesario para resolver problemas potenciales antes de que impacten al usuario.

Respecto a la funcionalidad, se obtuvo una correlación de Rho = 0.820, siendo esta una de las dimensiones más robustas de la investigación. Este dato permite contrastar la realidad local con la tendencia de vanguardia presentada por Owoc y Stambulski (2025), quienes sugieren que para alcanzar niveles óptimos de funcionalidad se debe transitar hacia la automatización con Machine Learning. Si bien los antecedentes internacionales apuntan a la inteligencia artificial, los resultados demuestran que, en la práctica profesional de Tacna, la base de la funcionalidad sigue siendo una gestión de riesgos convencional bien ejecutada. Al respecto, Pressman (2010) define la calidad como la concordancia con los requisitos funcionales explícitos y advierte que "la calidad de un sistema es el resultado directo de las decisiones de diseño y de la solidez de las pruebas aplicadas", validando mi postura de que una gestión de riesgos orientada a los requisitos es la mejor salvaguarda de la funcionalidad.

Para la fiabilidad del software, se registró una correlación significativa de Rho = 0.686, lo que indica que, en la muestra, una gestión de riesgos estructurada favorece la estabilidad operativa. Este resultado complementa la revisión sistemática de Masso et al. (2020), quienes identificaron que las brechas en la gestión de riesgos a lo largo del ciclo de vida impactan negativamente en la capacidad del sistema para operar sin fallos. Mientras que Masso et al. ofrece una visión general, el estudio cuantifica esta relación, demostrando que la mitigación temprana reduce la probabilidad de errores en producción. También este resultado es coherente con los hallazgos de Piñero et al. (2021), quienes señalan que integrar buenas prácticas de gestión de riesgos desde el inicio del desarrollo reduce la

aparición de fallos y mejora la satisfacción del usuario, del mismo modo, Ramírez (2022) comprobó que aplicar modelos como CMMI mejora la confiabilidad al fortalecer los controles internos del desarrollo. Asimismo, lo expuesto por Betetta Soler y Cordova Matos (2025), quienes demostraron en su estudio en empresas de Lima que una gestión de pruebas rigurosa y documentada se traduce directamente en la identificación temprana de errores, mitigando así los riesgos técnicos asociados a la fiabilidad. Como sustento, Pressman (2010) define la fiabilidad como la probabilidad de operación libre de fallos, subrayando que "un proceso de software que no integre el aseguramiento de la calidad y la gestión de riesgos está destinado a producir sistemas frágiles" que fallan bajo condiciones de uso real.

En la eficiencia del rendimiento, se logró establecer una correlación de Rho = 0.728 entre las variables analizadas. Este hallazgo se alinea con el modelo de Sernaque (2022), quien sostiene que la gestión de riesgos permite optimizar el uso de recursos y garantizar que el software responda eficientemente. Piñero et al. (2021) destacan que la eficiencia del desempeño puede garantizarse cuando se aplican buenas prácticas desde las etapas iniciales del ciclo de vida del software. En la investigación, se pudo validar que la optimización del rendimiento es una consecuencia directa de haber identificado riesgos técnicos y de infraestructura de manera oportuna. Sobre este punto, Pressman (2010) afirma que el rendimiento no depende solo del código, sino de la arquitectura técnica decidida bajo un análisis de riesgos, señalando que "la eficiencia es un atributo de calidad que debe medirse y controlarse desde las etapas iniciales" para evitar cuellos de botella que afecten la experiencia del usuario.

Finalmente, se alcanzó una correlación de Rho = 0.715 en la dimensión de mantenibilidad, lo que demuestra que la gestión de riesgos facilita la evolución del software sin degradar su calidad técnica. Este resultado dialoga con el estudio de Afshari y Javdani Gandomani (2022), quienes critican que en entornos ágiles la falta de documentación de riesgos suele generar una "deuda técnica" que dificulta el mantenimiento. Los datos sugieren que los profesionales que gestionan sus riesgos logran software más flexible al cambio. Como síntesis final de mi discusión,

Pressman (2010) explica que la mantenibilidad es la facilidad con la que un programa puede corregirse o mejorarse, y sentencia que "el software de calidad es aquel que posee una estructura interna sólida", permitiendo que el mantenimiento sea una actividad de mejora y no una lucha constante contra errores heredados de una mala planificación.

4.4. Limitaciones del estudio y mejoras identificadas

El estudio presentó ciertas limitaciones relacionadas principalmente con el alcance geográfico y metodológico, se centró en los ingenieros de sistemas del CIP – Tacna, lo que limita la generalización de los resultados a otros contextos profesionales o regionales, asimismo el análisis se basó en percepciones declaradas mediante encuesta, sin integrar mediciones de desempeño reales de proyectos, lo cual restringe la validación empírica directa.

Sin embargo, estas limitaciones abren líneas de mejora futura, se recomienda incorporar modelos predictivos como PLS-SEM, aplicar simulaciones de riesgo y ampliar la muestra a otros capítulos profesionales, con el fin de validar de forma integral la influencia de la gestión de riesgos sobre la calidad del software.

4.5 Descripción de la solución propuesta y su implementación

Con base en los resultados obtenidos, que demostraron la existencia de una correlación significativa entre la gestión de riesgos y la calidad del software, se propone una solución técnica denominada “Modelo integrado de gestión de riesgos en proyectos de software (MIGRISOFT)”, este modelo busca fortalecer los procesos de desarrollo de software mediante la incorporación sistemática de la gestión de riesgos en todas las etapas del ciclo de vida del proyecto, con el propósito de mejorar la calidad del producto, reducir los fallos y optimizar los recursos humanos y financieros.

El modelo MIGRISOFT se sustenta en los principios de la ISO 31000:2018, el PMBOK 7ª Edición y los criterios de calidad de la ISO/IEC 25010:2011, adaptados a las condiciones operativas, su estructura propone un enfoque práctico, modular y progresivo que permite su implementación tanto en proyectos de

pequeña escala como en desarrollos complejos, garantizando control, trazabilidad y aprendizaje continuo.

4.5.1. Estructura del modelo MIGRISOFT

El modelo se compone de cuatro fases operativas interconectadas que se ejecutan de manera iterativa y cíclica, asegurando la identificación, mitigación y control permanente de los riesgos durante el desarrollo del software.

Fase 1: Identificación de riesgos

En esta primera etapa se elabora un registro maestro de riesgos con apoyo de listas de verificación, entrevistas a los equipos de desarrollo y revisión de experiencias pasadas. Los riesgos se clasifican en tres categorías:

- Técnicos: errores de codificación, incompatibilidad de versiones, defectos en la arquitectura, fallos en pruebas de integración.
- Humanos: rotación de personal, falta de capacitación, comunicación deficiente, errores de estimación de tareas.
- Financieros y de gestión: retrasos en cronogramas, sobrecostos, falta de recursos o incumplimiento contractual.

Para sistematizar esta información se recomienda el uso de formularios digitales en Google Sheets o Microsoft Excel, que permitan registrar el tipo de riesgo, descripción, responsable, causa raíz y probabilidad estimada.

Fase 2: Evaluación y priorización de riesgos

Una vez identificados, los riesgos son evaluados mediante una matriz de probabilidad–impacto, que asigna valores del 1 al 5 en función de su frecuencia y severidad, la combinación de ambos criterios permite definir un nivel de exposición total. Para la representación visual se recomienda la herramienta Heat Map (Mapa de calor), que permite priorizar los riesgos más relevantes y focalizar los esfuerzos de mitigación.

Asimismo, esta fase incluye la definición de indicadores de riesgo (KRIs), como el número de defectos por módulo, variación de cronograma y frecuencia de incidencias, que facilitarán la medición objetiva del avance del proyecto y la efectividad de las medidas correctivas.

Fase 3: Planificación de respuestas

Con base en la evaluación anterior, se diseñan estrategias de respuesta al riesgo, siguiendo los lineamientos del PMBOK:

- Evitar: modificar el plan del proyecto para eliminar la amenaza, por ejemplo sustituir un componente inestable.
- Mitigar: reducir la probabilidad o impacto mediante acciones preventivas, por ejemplo pruebas automatizadas o revisiones de código.
- Transferir: delegar la responsabilidad a terceros, a través de contratos de garantía, seguros y soporte externo.
- Aceptar: asumir el riesgo cuando su impacto es bajo o el costo de mitigación es elevado.

Cada acción se documenta en un plan de respuesta al riesgo, con responsable designado, recursos requeridos, fecha de ejecución y mecanismo de verificación; para el control operativo se recomienda implementar tableros de seguimiento en Trello, Jira o Asana, donde cada tarea se asocia al riesgo correspondiente y se actualiza su estado en tiempo real.

Fase 4: Monitoreo, control y retroalimentación

La última fase consiste en el seguimiento continuo del ciclo de riesgos durante todo el proyecto, cada semana o sprint el equipo revisa la matriz de riesgos, actualiza los valores de probabilidad e impacto, y documenta los incidentes ocurridos y las acciones correctivas aplicadas.

Los resultados se presentan en reportes de estado generados automáticamente en Excel o Power BI, permitiendo observar tendencias, riesgos recurrentes y variaciones en la calidad del producto. Al cierre del proyecto se ejecuta una revisión post-implementación en la que se registran las lecciones aprendidas y los riesgos más críticos, fortaleciendo la memoria organizacional y la madurez del equipo en gestión de riesgos.

4.5.2. Implementación operativa del modelo

La implementación del modelo MIGRISOFT se plantea en cuatro etapas de despliegue progresivo, con actividades específicas para cada fase:

Tabla 38
Descripción de las etapas de implementación

Etapa	Descripción de actividades principales	Responsables	Herramientas sugeridas
Preparación y capacitación	Capacitación en gestión de riesgos, estándares ISO 31000 y uso de herramientas colaborativas.	Jefe de proyecto / Ingeniero de calidad	Google Workspace, Microsoft Teams
Integración al proceso de desarrollo	Inserción del registro de riesgos dentro del flujo de trabajo ágil (Scrum/Kanban).	Scrum Master / Líder técnico	Jira, Trello, Asana
Evaluación y control continuo	Aplicación de matriz de riesgos y actualización de tableros de seguimiento.	Equipo de desarrollo / QA	Excel, Power BI
Retroalimentación y mejora	Documentación de resultados y actualización de políticas de riesgo.	Comité de calidad / PMO	Documentos internos, Confluence

4.5.3. Indicadores de desempeño y resultados esperados

La implementación del modelo se evaluará mediante indicadores clave de desempeño (KPIs) que permitan medir la efectividad de la gestión de riesgos en la mejora de la calidad del software:

Tabla 39
Descripción de indicadores de desempeño

Indicador	Descripción	Meta esperada
Tasa de defectos por módulo	Número de fallos detectados durante pruebas / total de módulos	≤ 3%
Cumplimiento de cronograma	Porcentaje de tareas completadas en plazo	≥ 90%
Satisfacción del cliente interno	Valor promedio en encuesta postentrega	≥ 4/5
Reducción de incidentes críticos	Disminución de errores de alta severidad respecto al proyecto anterior	≥ 25%

4.5.4. Beneficios técnicos y estratégicos

El modelo MIGRISOFT ofrece beneficios tangibles tanto para los equipos de desarrollo como para las organizaciones que lo adopten:

- Prevención sistemática de fallos, reduciendo retrabajos y pérdidas de tiempo.
- Optimización de recursos humanos y financieros, mediante planificación anticipada.
- Fortalecimiento de la cultura de calidad, basada en evidencia y mejora continua.
- Estandarización de procesos según normas internacionales, facilitando auditorías o certificaciones.
- Transferencia tecnológica sostenible, adaptable a otros sectores o disciplinas de la ingeniería.

4.5.5 Caso Práctico de Aplicación del Modelo MIGRISOFT

Para validar la operatividad del modelo propuesto, se presenta un escenario de aplicación en un proyecto de desarrollo de software denominado "InvenTrack", un sistema web para la gestión de inventarios y facturación electrónica.

Fase 1: Identificación de Riesgos

Siguiendo los lineamientos de MIGRISOFT, el equipo de desarrollo inició el proyecto elaborando un registro maestro de riesgos. Se utilizaron listas de verificación y entrevistas con el líder técnico para clasificar las amenazas en tres categorías:

- Riesgo Técnico: Inestabilidad en la conexión con el servidor de la entidad tributaria para la facturación electrónica.
- Riesgo Humano: Posible rotación del desarrollador encargado del módulo de seguridad.
- Riesgo de Gestión: Retraso en la entrega de los requerimientos finales por parte del cliente.

Toda la información fue sistematizada en un formulario digital de Google Sheets para registrar causas raíz y responsables.

Fase 2: Evaluación y Priorización

Los riesgos identificados se sometieron a una matriz de probabilidad–impacto con valores del 1 al 5.

- El riesgo técnico de "Inestabilidad de conexión" obtuvo una puntuación de 20 (P:4 x I:5), ubicándose en la zona de alta prioridad del Heat Map.
- Se establecieron indicadores de riesgo (KRIs) como el número de defectos detectados por módulo para monitorear la severidad en tiempo real.

Fase 3: Planificación de Respuestas

Basándose en el PMBOK 7ª Edición, se diseñaron las siguientes estrategias:

- Mitigar (Riesgo Técnico): Implementación de una cola de mensajes (queues) para reintentar el envío de facturas en caso de caída del servidor externo.

- Aceptar (Riesgo de Gestión): Se asumió el retraso menor en requerimientos secundarios debido a que el costo de aceleración superaba el beneficio.

- Control Operativo: Las tareas de mitigación se asignaron en un tablero de Jira, permitiendo el seguimiento del estado de cada riesgo en tiempo real.

Fase 4: Monitoreo, Control y Retroalimentación

Durante cada sprint, el equipo revisó la matriz de riesgos para actualizar los valores de probabilidad e impacto.

- Al finalizar el proyecto, se generó un reporte en Power BI que demostró una tasa de defectos del 2.8%, cumpliendo con la meta establecida de $\leq 3\%$.

- En la reunión post-implementación, se documentaron las lecciones aprendidas sobre la integración de APIs externas, fortaleciendo la memoria organizacional del equipo.

4.6 Evaluación de costos, sostenibilidad y viabilidad a largo plazo

El proyecto fue autofinanciado y se ejecutó con recursos digitales de libre acceso, como Google Forms, Excel y SPSS, lo que garantiza su bajo costo operativo, la propuesta es sostenible porque aprovecha plataformas ya disponibles y puede replicarse en cualquier entorno organizacional sin requerir licencias privativas. A largo plazo, el modelo MIGRISOFT es viable y escalable, pues puede adaptarse a diferentes tipos de proyectos tecnológicos, manteniendo su utilidad en entornos públicos y privados gracias a su enfoque práctico, modular y de fácil implementación.

CONCLUSIONES

1. En primer lugar, se concluye que la gestión de riesgos presenta una relación altamente significativa con la calidad del software desarrollado por los ingenieros colegiados del Colegio de Ingenieros del Perú – Consejo Departamental Tacna. Esto se evidencia en el coeficiente de correlación de Spearman de 0,884 con una significancia de 0,000, lo cual indica una asociación positiva alta. A partir de este resultado, se deduce que cuando se implementan procesos de identificación, evaluación y tratamiento de riesgos de forma adecuada, se fortalecen de manera integral las principales dimensiones de calidad, como la funcionalidad, la fiabilidad, el rendimiento y la mantenibilidad. Por consiguiente, integrar la gestión de riesgos como parte estructural del ciclo de vida del desarrollo de software no solo reduce las probabilidades de fallos, sino que también optimiza los resultados técnicos y operativos del producto final.

2. En segundo lugar, se establece que existe una relación significativa y positiva entre la gestión de riesgos y la funcionalidad del software. El análisis arrojó un coeficiente de Spearman de 0,820 y un valor p de 0,000, lo que demuestra una fuerte asociación estadística. Este resultado permite afirmar que una gestión efectiva de riesgos contribuye directamente al cumplimiento de los requisitos funcionales definidos en la etapa de planificación, permitiendo que el software desarrollado responda fielmente a las necesidades del usuario. En este sentido, la adecuada anticipación de posibles amenazas técnicas o de alcance facilita la correcta implementación de funcionalidades, optimizando la experiencia de uso y la utilidad práctica del sistema. Así, se refuerza la importancia de abordar los

riesgos desde el diseño del proyecto para asegurar un producto alineado a las expectativas iniciales.

3. Asimismo, se concluye que la fiabilidad del software guarda una relación significativa con la gestión de riesgos aplicada durante el desarrollo. El coeficiente de Spearman obtenido fue de 0,686, con una significancia estadística de 0,000, lo cual refleja una correlación moderada-alta y positiva. Este hallazgo indica que las medidas preventivas y correctivas basadas en un análisis riguroso de riesgos permiten disminuir la frecuencia de fallos del sistema, incrementando su estabilidad en entornos de uso real. En consecuencia, el diseño de planes de mitigación tempranos y la documentación sistemática de posibles contingencias fortalecen la capacidad del software para operar con continuidad y sin interrupciones, incluso ante imprevistos. Por ello, se ratifica que una gestión de riesgos bien ejecutada es un factor clave para mejorar la confiabilidad de los productos informáticos.

4. Del mismo modo, se evidencia una correlación significativa entre la gestión de riesgos y el rendimiento del software, sustentada por un coeficiente de 0,728 y un valor de significancia de 0,000. Esta relación positiva considerable permite inferir que una adecuada planificación y seguimiento de riesgos técnicos —especialmente aquellos vinculados a la carga de trabajo, uso de recursos y optimización del código— favorece el mantenimiento de un desempeño estable en el sistema. De hecho, el monitoreo constante de indicadores críticos y la previsión de cuellos de botella permiten que el software se desempeñe de manera eficiente, incluso en condiciones exigentes. En tal sentido, se concluye que la gestión de riesgos no solo garantiza la funcionalidad del sistema, sino también su capacidad de respuesta, escalabilidad y eficiencia operativa.

5. Finalmente, se determinó que la gestión de riesgos guarda una relación significativa con la mantenibilidad del software, con un coeficiente de correlación de 0,715 y significancia de 0,000. Este resultado evidencia que cuando se integran prácticas de gestión de riesgos desde las etapas iniciales del

desarrollo, se favorece un diseño estructurado, modular y documentado, que facilita futuras tareas de mantenimiento, actualización o adaptación. A través de la previsión de escenarios críticos y la definición de planes de contingencia, los desarrolladores pueden reducir la complejidad del código y minimizar el impacto de los cambios sobre el funcionamiento general del sistema. En consecuencia, se concluye que una gestión de riesgos eficiente no solo mejora el desempeño inicial del software, sino que también prolonga su vida útil y facilita su evolución en contextos dinámicos.

RECOMENDACIONES

1. En vista de que la investigación confirma una alta correlación entre la gestión de riesgos y la calidad del software, es fundamental que las organizaciones adopten un enfoque integral y estandarizado para abordar dicha gestión. En ese sentido, se recomienda implementar un marco de gestión de riesgos basado en modelos reconocidos como ISO/IEC 31000 o CMMI, que contemple las etapas de identificación, evaluación, planificación, monitoreo y control de riesgos. Esta estrategia debe aplicarse desde las fases iniciales del ciclo de vida del software, involucrando activamente a todos los actores del proyecto, desde desarrolladores hasta usuarios finales. La implementación coherente de este enfoque fortalecerá la cultura organizacional en torno al riesgo y favorecerá el desarrollo de productos con mayor calidad, funcionalidad y sostenibilidad.

2. Dado que se ha evidenciado una relación positiva entre una buena gestión de riesgos y la funcionalidad del software, es necesario adoptar prácticas que permitan vincular los requerimientos funcionales con posibles riesgos desde el inicio del proyecto. Para ello, se recomienda utilizar herramientas de trazabilidad y control de requerimientos, junto con técnicas como análisis de impacto de cambios, pruebas de aceptación y diagramas de casos de uso. Estas acciones deben realizarse en un entorno de desarrollo iterativo e incremental, donde cada funcionalidad sea validada progresivamente. De esta manera, se garantiza que el software final cumpla con los objetivos funcionales establecidos, asegurando la satisfacción del usuario y reduciendo los costos de retrabajo.

3. Considerando que una adecuada gestión de riesgos incide directamente en la fiabilidad del software, se sugiere promover un enfoque orientado a la resiliencia del sistema. Esto implica incorporar arquitecturas tolerantes a fallos, establecer criterios de robustez en las historias de usuario y desarrollar pruebas de regresión automatizadas. Además, es conveniente establecer políticas de revisión de código estructuradas y aplicar métricas específicas como el MTTF (tiempo medio entre fallos). Al aplicar estas acciones, no solo se disminuye la probabilidad de errores críticos, sino que también se aumenta la confianza en la estabilidad operativa del sistema, incluso en entornos complejos o de alta exigencia.

4. Dado que los resultados revelan una correlación significativa entre la gestión de riesgos y el rendimiento del software, se recomienda aplicar técnicas de análisis de rendimiento desde la etapa de diseño del sistema. Específicamente, se deben utilizar perfiles de carga, pruebas de estrés y herramientas APM (Application Performance Monitoring) para anticipar cuellos de botella y gestionar recursos como CPU, memoria y concurrencia de manera proactiva. Asimismo, se debe incorporar la optimización de código y la refactorización como actividades recurrentes. Estas prácticas permiten mantener la eficiencia del sistema bajo distintas condiciones operativas, garantizando una respuesta oportuna ante situaciones críticas y asegurando la calidad del servicio ofrecido.

5. Se recomienda a las organizaciones de desarrollo de software la adopción formal del modelo MIGRISOFT como eje transversal en el ciclo de vida de sus proyectos. Para asegurar su efectividad, la implementación debe enfocarse en los siguientes puntos clave: Ejecución Iterativa: Integrar las cuatro fases operativas (Identificación, Evaluación, Planificación y Monitoreo) de forma cíclica en cada sprint o etapa del proyecto. Priorización Estratégica: Utilizar la matriz de probabilidad-impacto y mapas de calor para focalizar esfuerzos en los riesgos de mayor exposición. Monitoreo de Calidad: Evaluar el éxito del modelo mediante indicadores específicos, como una tasa de defectos por módulo $\leq 3\%$ y un cumplimiento de cronograma $\geq 90\%$. Sostenibilidad

Operativa: Aprovechar herramientas de libre acceso (Excel, Google Sheets, Trello) para garantizar un bajo costo y facilitar la réplica del modelo en cualquier entorno. Cultura de Aprendizaje: Realizar revisiones post-implementación para documentar lecciones aprendidas y fortalecer la madurez del equipo.

REFERENCIAS

Afshari, M., & Javdani Gandomani, T. (2022). A novel risk management model in the Scrum and extreme. *International Journal of Electrical and Computer Engineering*, 2911 - 2921.

Ali, M., Yap, N., Ghani, A., Zulzalil, H., Admodisastro, N., & Najafabadi, A. (2022). A Systematic Mapping of Quality Models for AI Systems, Software and Components. *Appl. Sci., 12*(17), 1-19. Obtenido de https://doi.org/10.3390/app12178700

Altexsoft. (2023). *Quality Assurance, Quality Control and Testing — the Basics of Software Quality Management.* Obtenido de https://www.altexsoft.com/whitepapers/quality-assurance-quality-control-and-testing-the-basics-of-software-quality-management/

Arias, P., Ferro, R., & Abuchar, A. (2019). Analysis of methodologies applied to risk management in software development projects in Colombia. *Revista Ingeniería, Investigación y Desarrollo, 19*(2), 29-40. Obtenido de https://doi.org/10.19053/1900771X.v19.n2.2019.12879

Armstrong, M. (2019). *Handbook of Human Resource Management Practice.* Kogan Page.

Benites, C. (2022). *Aplicación de Aseguramiento de Calidad de Software y su influencia en proyectos de la empresa 3M, 2022.* Tesis de maestría, Universidad Científica del Sur, Lima.

Berk, J., & DeMarzo, P. (2020). *Corporate Finance.* Pearson.

Betetta Soler, M. P., & Cordova Matos, L. E. (2025). *Aplicación de ISTQB para la gestión de aseguramiento de calidad en una empresa de desarrollo de software, Lima 2024.* Lima: Universidad César Vallejo. Obtenido de https://repositorio.ucv.edu.pe/handle/20.500.12692/165465

Bhanushali, A. (2023). Ensuring Software Quality Through Effective Quality Assurance Testing: Best Practices and Case Studies. *International Journal of Advances in Scientific Research and Engineering, 23*(4), 1-18. Obtenido de https://www.researchgate.net/profile/Amit-Bhanushali/publication/375342628_Ensuring_Software_Quality_Thr

ough_Effective_Quality_Assurance_Testing_Best_Practices_and_Case_Studies/links/65474257ce88b87031c822df/Ensuring-Software-Quality-Through-Effective-Qua

Boehm, B. (2019). *Software Quality Framework.* Springer.

Boehm, B., & Brown, J. (1978). *Merritt.: Characteristics of Software Quality.* Amsterdam: Elsevier.

Bravo, A. (2017). *Herramientas de software de apoyo a la gestión de riesgos en proyectos basado en la guía PMBOOK.* Tesis de titulación, Pontificia Universidad Católica del Perú, Lima.

Brigham, E., & Ehrhardt, M. (2019). *Financial Management: Theory & Practice.* Cengage Learning.

Brown, L. (27 de Mayo de 2025). *Top Risk Management Tools for Project Managers*. Obtenido de https://www.invensislearning.com/blog/risk-management-tools-techniques-in-pm/

Canedo, A., & Santos, G. (2019). Factors Affecting Software Development Productivity: An empirical study. *the XXXIII Brazilian Symposium.* Obtenido de http://dx.doi.org/10.1145/3350768.3352491

Castro, Y., Solarte, G., & Muñoz, L. (2019). Planning, Management and Control of Software Quality. *Scientia et Technica, 24*(4). Obtenido de https://www.redalyc.org/journal/849/84961238009/html/

Chapman, C., & Ward, S. (2020). *Project Risk Management: Processes, Techniques and Insights.* Wiley.

Chernish, M., & Yemets, B. (04 de Febrero de 2024). *9 Risks in Software Development and How to Mitigate Them*. Obtenido de https://clockwise.software/blog/software-development-risks/

CMMI Product. (2018). *CMMI for Development, Version 1.3.* Carnegie Mellon University. doi:https://doi.org/10.1184/R1/6572342.v1

Crouhy, M., Galai, D., & Mark, R. (2021). *The Essentials of Risk Management.* McGraw-Hill.

Davydov, R. (14 de Junio de 2024). *Gestión de riesgos durante todo el ciclo de vida del desarrollo de software*. Obtenido de https://drjenespanol.com/articulos/gestion-de-riesgos-durante-todo-el-ciclo-de-vida-del-desarrollo-de-software/

Dessler, G. (2021). *Human Resource Management.* Pearson.

Dromey, R. (1996). Cornering the chimera [software quality]. *IEEE Softw., 13*, 33-43. Obtenido de https://doi.org/10.1109/52.476284

Fitriani, W., Rahayu, P., & Sensuse, D. (2016). Challenges in agile software development: A systematic literature review. *International Conference on Advanced Computer Science and Information Systems*, (págs. 155-164). Obtenido de https://ieeexplore.ieee.org/document/7872736

Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2019). *Design patterns: Elements of reusable object-oriented software.* Addison-Wesley.

Gontijo, B., Keil, M., Sanches, C., & Dinis, A. (2020). A Risk Management Tool for Agile Software Development. *Journal of Computer Information Systems, 61*(6), 561-570. Obtenido de https://doi.org/10.1080/08874417.2020.1839813

Guerrero, M., & Hernández, G. (2024). Metrics for the Evaluation of Team Productivity Factors in Agile Software Development: A Systematic Literature Mapping. *TecnoLógicas, 27*(59), 1-20. Obtenido de https://doi.org/10.22430/22565337.2918

Güveyi, E., Aktaş, M., & Kalipsiz, O. (2020). Human Factor on Software Quality: A Systematic Literature Review. *Computational Science and Its Applications – ICCSA 2020*(12252), 918-930. Obtenido de https://doi.org/10.1007/978-3-030-58811-3_65

Hillson, D. (2020). *The Risk Management Handbook: A Practical Guide to Managing the Multiple Dimensions of Risk.* Kogan Page.

IEEE. (2021). *ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes -- Risk management.* IEEE Computer Society.

International Organization for Standardization. (2011). *ISO/IEC 25010.* International Organization for Standardization.

International Organization for Standardization. (2018). *ISO 31000:2018. Risk management—Guidelines.* Colombia. Obtenido de https://www.ramajudicial.gov.co/documents/5454330/14491339/Norma.ISO.31000.2018.Espanol.pdf

Jackowski, M. (29 de Junio de 2023). *Software Risk Analysis: Tools and Strategies for Effective Risk Assessment*. Obtenido de https://asperbrothers.com/blog/software-risk-analysis/

Jedrzejowska, M. (11 de Marzo de 2022). *Process Areas in CMMI 2.0 model*. Obtenido de https://spyro-soft.com/blog/automotive/process-areas-in-cmmi-2-0-model

Kerzner, H. (2021). *Project Management: A Systems Approach to Planning, Scheduling, and Controlling .* Wiley.

Lange, K. (22 de Octubre de 2024). *What Is APM? Application Performance Monitoring, Explained*. Obtenido de https://www.splunk.com/en_us/blog/learn/apm-application-performance-monitoring.html

Lewis, W., & Veerapillai, G. (2004). *Software Testing and Continuous Quality Improvement.* USA: CRC Pres.

Luthfiansyah, F., Prasetyo, A., & Raharjo, T. (2024). A Systematic Review of Risk Management Tools and Techniques in Software Projects. *Indonesian Journal of Computer Science, 13*(1), 534-549. Obtenido de http://dx.doi.org/10.33022/ijcs.v13i1.3694

Machuca, L., Gasca, G., Morillo, S., & Restrepo, L. (2021). Factores sociales y humanos que influyen en la productividad del desarrollo de software: Medición de la percepción. *RISTI: Revista Ibérica de Sistemas e Tecnologias de Informação, 41*, 488-502. Obtenido de https://dialnet.unirioja.es/servlet/articulo?codigo=8610713&orden=0&info=link

Masso, J., Pino, F., Pardo, C., García, F., & Piattini, M. (2020). Risk management in the software life cycle: A systematic literature review. *Computer Standards & Interfaces*, 103431.

McCall, J., Richards, P., & Walters, G. (1977). *Factors in Software Quality, Volumes I, II, and III.* Washington: US Rome Air Development Center Reports.

McConnell, S. (2021). *Code Complete: A Practical Handbook of Software Construction.* Microsoft Press.

McGrath, S., & Whitty, J. (2020). The suitability of PRINCE2 for engineering infrastructure. *JMPM, 7*(4), 312-347. Obtenido de https://www.researchgate.net/profile/Jonathan-Whitty-2/publication/350006174_The_suitability_of_PRINCE2_for_engineering_infrastructure/links/604afd6d45851543166f3964/The-suitability-of-PRINCE2-for-engineering-infrastructure.pdf?_tp=eyJjb250ZXh0Ijp7ImZpcnN

Morales, E. (2018). *Validación metodología Pmbok en gestión de riesgos del proceso de desarrollo de software empresa sector educación.* Tesis de maestría, Universidad César Vallejo, Lima. Obtenido de https://hdl.handle.net/20.500.12692/14502

Nikolaenko, V., & Sidorov, A. (2023). Analysis of 105 IT Project Risks. *Journal of Risk and Financial Management, 16*(1), 33. Obtenido de https://doi.org/10.3390/jrfm16010033

Owoc, M. L., & Stambulski, A. (2025). Software quality management: Machine learning for recommendation of regression test suites.

Journal of Economics and Management, 117-137. doi:https://doi.org/10.22367/jem.2025.47.05

Paucar, D., Acho, P., & Peralta, C. (2021). Relación de la gestión de riesgos y calidad de software realizados por los profesionales del Colegio de Ingenieros del Perú del Consejo Departamental de Lima. *Interfases, 14*(14), 41-64. Obtenido de https://doi.org/10.26439/interfases2021.n014.5111

Perdomo, W., & Zapata, C. (2021). Medidas de la calidad del producto de software y su relación con los estados del alfa sistema de software. *Ingeniare. Revista chilena de ingeniería, 29*(2), 346-363. Obtenido de https://www.scielo.cl/pdf/ingeniare/v29n2/0718-3305-ingeniare-29-02-346.pdf

Pérez, Y., Gallegos, J., Zapata, S., & Ccama, D. C. (2020). Design Thinking en la planificación de pruebas de software. *Revista Innovación y Software, 1*(2), 40-51. Obtenido de https://www.redalyc.org/journal/6738/673870835004/html/

Piñero, M., Marin, A., Trujillo, Y., & Buedo, D. (2021). Buenas prácticas para prevenir los riesgos de la eficiencia del desempeño en productos de software. *Revista Cubana de Ciencias Informáticas, 15*(1), 89-113. Obtenido de http://scielo.sld.cu/pdf/rcci/v15n1/2227-1899-rcci-15-01-89.pdf

PMI. (2021). *Project Management Institute (PMI).* PMI Publications.

Pressman, R. (2010). *Software Engineering: A Practitioner's Approach.* McGraw-Hill.

Pressman, R., & Maxim, B. (2021). *Software Engineering: A Practitioner's Approach.* New York: McGraw-Hill Global Education Holdings.

Radecka, R. (9 de Julio de 2024). *Cost of quality or lack thereof in software development*. Obtenido de https://www.llinformatics.com/blog/cost-of-quality-in-software-development

Ramirez, D. (2022). *Proceso de Gestión de Riesgos para Proceso de Gestión de Riesgos para la UCI.* Tesis de maestría, Universidad de las Ciencias Informáticas, La Habana. Obtenido de https://repositorio.uci.cu/jspui/handle/ident/7966

Requena, P., & Alcaide, J. (2017). *Metodología de la investigación cuantitativa en las ciencias sociales.* Delta Publicaciones.

Reyes, E. (2022). *Metodología de la investigación científica.* Conneaut Lake: Page Publishing INC. Obtenido de https://books.google.com.pe/books?id=SmdxEAAAQBAJ&lpg

Rivera, J., Gonzabay, E., & Mendoza, B. (2023). Gestión y Mejora de Procesos de Desarrollo de Software. *Ciencia Latina Revista Científica Multidisciplinar, 7*(5), 3191-3210. Obtenido de https://doi.org/10.37811/cl_rcm.v7i5.7948

Rodríguez, Y. (2020). *Metodología de la investigación.* México: Soluciones educativas Klik.

Rojas, M., Castillo, J., & Mendoza, A. (2023). Seguridad de la información en la prevención de pérdida de datos: una revisión sistemática. *Revista Innovación y Software, 4*(2), 182-200. Obtenido de http://dx.doi.org/10.48168/innosoft.s12.a92

Sernaque, N. (2022). *Modelo para la gestión de riesgos de desarrollo de software bajo la perspectiva de la gestión de proyectos.* Tesis de maestría, Universidad Nacional Pedro Ruiz Gallo, Lambayeque. Obtenido de https://hdl.handle.net/20.500.12893/10434

Sifuentes, Y., & Peralta, J. (2022). Modelo de medición y evaluación de calidad del software basado en la norma ISO/IEC 25000 para medir la usabilidad en productos de software académicos universitarios. *TecnoHumanismo, 2*(4), 44-66. Obtenido de https://doi.org/10.53673/th.v2i4.125

Sommerville, I. (2020). *Software Engineering.* Pearson.

Tamayo, M. (2019). *Metodología de la investigación en las ciencias sociales.* Limusa.

Tasnim, A. (2024). Importance of Risk Management in Software Engineering. *Asian Journal of Technology & Management Research, 14*(1), 124-127. Obtenido de https://www.researchgate.net/profile/Ahmad-Siddiqui-5/publication/380854773_Importance_of_Risk_Management_in_Software_Engineering/links/665166ca0b0d2845745838b6/Importance-of-Risk-Management-in-Software-Engineering.pdf

Thangiah, M., & Basri, S. (2016). A preliminary analysis of various testing techniques in Agile development - a systematic literature review. *3rd International Conference on Computer and Information Sciences.* Obtenido de http://dx.doi.org/10.1109/ICCOINS.2016.7783283

Tummala, R., & Schoenherr, T. (2020). *Managing Risk in Supply Chains: Strategies and Practices.* Springer.

Valverde Flores, D. A. (2022). *Implementación de una gestión de riesgos de TI para mejorar la seguridad de la información de una empresa de agencia publicitaria - 2021.* Lima: Universidad Tecnológica del Perú. Obtenido de https://repositorio.utp.edu.pe/handle/20.500.12867/5529

Viteri, J. (2019). *Propuesta de gestión de riesgos de proyectos software desarrollados con la metodología Scrum.* Artículo científico, Escuela de las Fuerzas Armadas ESPE, Ecuador.

Zlatko, S. (2021). Impact of Agile-Hybrid Methods and Approaches on Software Quality: A Systematic Literature Review. *Proceedings of the Central European Conference on Information and Intelligent Systems, 32*, 13-15. Obtenido de https://archive.ceciis.foi.hr/public/conferences/2021/Proceedings/TSE/TSE6.pdf

Zrymiak, D. (1 de Setiembre de 2020). *Cost of Quality*. Obtenido de https://www.qualitymag.com/articles/96199-cost-of-quality

APÉNDICE

Anexo 01: Matriz de consistencia

PROBLEMA	OBJETIVOS	HIPÓTESIS	VARIABLES / DIMENSIONES	METODOLOGÍA
1. INTERROGANTE PRINCIPAL ¿Cómo se relaciona la gestión de riesgos y la calidad del software realizado por profesionales del CIP, Tacna – 2024?	**1. OBJETIVO GENERAL** Determinar cómo se relaciona de la gestión del riesgo con la calidad del software desarrollado por profesionales del CIP Tacna	**1. HIPÓTESIS GENERAL** La gestión de riesgos tiene una relación significativa con la calidad del software realizado por profesionales del CIP, Tacna – 2024	**Variable 1 (X)** X1: Gestión de riesgos Dimensiones: – Procesos de gestión de riesgos Indicador: – Efectividad de los procesos de gestión de riesgos – Gestión de recursos financieros Indicador: – Adecuación de la gestión financiera en la mitigación de riesgos. – Gestión de talento humano Indicador: – Percepción de la capacidad del equipo en la gestión de riesgos **Variable 2 (Y)** Y1: Calidad de software	– **Tipo de investigación:** Básica. – **Diseño de investigación**: No experimental – transversal. – **Ámbito de estudio:** CIP Tacna – **Población:** 754 Ingenieros colegiados en el Capítulo de Ingenieros de Sistemas CIP Tacna – **Muestra:** 255 ingenieros colegiados – **Técnicas de recolección de datos:** Encuesta – **Instrumentos:** Cuestionario
2. INTERROGANTES ESPECÍFICAS a) ¿Cuál es la relación de la gestión de riesgos y la funcionalidad del software realizado por profesionales del CIP, Tacna – 2024? b) ¿Cuál es la relación de la gestión de riesgos y la fiabilidad del software realizado por profesionales del CIP, Tacna – 2024? c) ¿Cuál es la relación de la gestión de riesgos y el rendimiento del software realizado por profesionales del CIP, Tacna – 2024? d) ¿Cuál es la relación de la gestión de riesgos y la	**2. OBJETIVOS ESPECÍFICOS** a) Analizar cómo se relaciona la gestión del riesgo con la funcionalidad del software desarrollado por profesionales del CIP Tacna. b) Establecer cómo se relaciona la gestión del riesgo con la fiabilidad del software desarrollado por profesionales del CIP Tacna. c) Señalar cómo se relaciona la gestión del riesgo con el rendimiento del software desarrollado	**2. HIPÓTESIS ESPECÍFICAS** a) La gestión de riesgos se relaciona significativamente con la funcionalidad del software realizado por profesionales del CIP, Tacna – 2024. b) La gestión de riesgos se relaciona significativamente con la fiabilidad del software realizado por profesionales del CIP, Tacna – 2024. c) La gestión de riesgos se relaciona significativamente con el rendimiento del software		

mantenibilidad del software realizado por profesionales del CIP, Tacna – 2024?	por profesionales del CIP Tacna. d) Especificar cómo se relaciona la gestión del riesgo con la mantenibilidad del software desarrollado por profesionales del CIP Tacna.	realizado por profesionales del CIP, Tacna – 2024. d) La gestión de riesgos se relaciona significativamente con la mantenibilidad del software realizado por profesionales del CIP, Tacna – 2024.	Dimensiones: – Funcionalidad Indicador: – Cumplimiento de requisitos – Fiabilidad Indicador: – Tasa de fallos – Rendimiento Indicador: – Eficiencia del rendimiento – Mantenibilidad Indicador: – Facilidad de mantenimiento	
Relevancia de la investigación CONTRIBUCIÓN DE LA INVESTIGACIÓN AL LOGRO DE LOS OBJETIVOS DE DESARROLLO SOSTENIBLE (ODS)				

Anexo 02: Instrumentos de recolección de datos

Cuestionario de recolección de datos

Apreciado ingeniero, a continuación, se presente un conjunto de afirmaciones. Por favor responda con sinceridad las siguientes preguntas. No hay respuestas correctas ni incorrectas. Esta encuesta es estrictamente confidencial y anónima, servirá como información para fines académicos.

Muy en desacuerdo	En desacuerdo	Neutral	De acuerdo	Muy de acuerdo
1	2	3	4	5

Ítems	**Puntuación**				
GESTIÓN DE RIESGOS					
Dimensión 01: Procesos de gestión de riesgos					
1. Los riesgos potenciales son identificados de manera oportuna y completa en los proyectos.	1	2	3	4	5
2. Se dispone de procedimientos establecidos para documentar y gestionar los riesgos.	1	2	3	4	5
3. Los procesos de gestión de riesgos son evaluados regularmente para garantizar su efectividad.	1	2	3	4	5
Dimensión 02: Gestión de recursos financieros					
4. Los recursos financieros asignados son suficientes para abordar los riesgos identificados.	1	2	3	4	5
5. El uso de los recursos financieros es supervisado regularmente para garantizar su efectividad en la mitigación de riesgos.	1	2	3	4	5
6. Se realizan ajustes en los recursos financieros según la evolución de los riesgos del proyecto.	1	2	3	4	5
Dimensión 03: Gestión de talento humano					
7. El equipo de los proyectos cuenta con la experiencia necesaria para gestionar eficazmente los riesgos.	1	2	3	4	5
8. Se proporciona capacitación regular al equipo sobre la identificación y mitigación de riesgos en proyectos de software.	1	2	3	4	5
9. Se dispone de herramientas específicas para apoyar la gestión de riesgos en los proyectos.	1	2	3	4	5
CALIDAD DE SOFTWARE					
Dimensión 01: Funcionalidad					
1. El software desarrollado cumple con todos los requisitos funcionales definidos en la etapa de planificación.	1	2	3	4	5
2. Se realizan pruebas exhaustivas para validar que todas las funcionalidades cumplen con los objetivos establecidos.	1	2	3	4	5
3. Las funcionalidades del software son comprensibles y utilizables por los usuarios finales.	1	2	3	4	5
Dimensión 02: Fiabilidad					
4. El software opera de manera estable y confiable bajo las condiciones de uso previstas.	1	2	3	4	5

5. Se realizan pruebas para garantizar que el software pueda recuperarse rápidamente en caso de fallos.	1	2	3	4	5
6. La frecuencia de fallos en el software es baja durante las fases de prueba y uso operativo.	1	2	3	4	5
Dimensión 03: Rendimiento					
7. El software responde de manera eficiente bajo las condiciones normales de uso.	1	2	3	4	5
8. Los recursos disponibles de hardware y software son utilizados de manera óptima por el sistema.	1	2	3	4	5
9. El rendimiento del software se mantiene constante incluso bajo altas cargas de trabajo.	1	2	3	4	5
Dimensión 04: Mantenibilidad					
10. El software permite realizar modificaciones sin afectar su funcionamiento general.	1	2	3	4	5
11. Las tareas de mantenimiento son realizadas en un tiempo razonable.	1	2	3	4	5
12. El software se adapta de manera flexible a los cambios en los requisitos del proyecto.	1	2	3	4	5

Gracias por su colaboración.

Anexo 03: Validación del instrumento

	UNIVERSIDAD PRIVADA DE TACNA Escuela de Postrado Centro de Investigación Formato de Validación por expertos		
Codificación CEIN fve - 001	Versión 00	Vigencia 2015	Páginas 02

INFORME DE OPINIÓN DE EXPERTOS DEL INSTRUMENTO DE INVESTIGACIÓN

I. DATOS GENERALES:

1.1. Apellidos y nombres del informante (Experto): Gianfranco Alexey Málaga Tejada
1.2. Grado Académico: Maestro
1.3 Profesión: Ingeniero de Sistemas
1.4. Institución donde labora: UNJBG
1.5. Cargo que desempeña: Docente
1.6 Denominación del Instrumento:
Cuestionario sobre gestión de riesgos y calidad de software realizado por profesionales del CIP Tacna
1.7. Autor del instrumento: Enrique Waldo Condori Siles
1.8 Programa de postgrado: Ingeniería de Software

II. VALIDACIÓN 1

INDICADORES DE EVALUACIÓN DEL INSTRUMENTO	CRITERIOS Sobre los ítems del instrumento	Muy Malo 1	Malo 2	Regular 3	Bueno 4	Muy Bueno 5
1. CLARIDAD	Están formulados con lenguaje apropiado que facilita su comprensión					X
2. OBJETIVIDAD	Están expresados en conductas observables, medibles				X	
3. CONSISTENCIA	Existe una organización lógica en los contenidos y relación con la teoría					X
4. COHERENCIA	Existe relación de los contenidos con los indicadores de la variable				X	
5. PERTINENCIA	Las categorías de respuestas y sus valores son apropiados					X
6. SUFICIENCIA	Son suficientes la cantidad y calidad de ítems presentados en el instrumento					X
	SUMATORIA PARCIAL					
	SUMATORIA TOTAL					28

	UNIVERSIDAD PRIVADA DE TACNA Escuela de Postrado Centro de Investigación Formato de Validación por expertos		
Codificación CEIN fve - 001	Versión 00	Vigencia 2015	Páginas 02

III. RESULTADOS DE LA VALIDACIÓN

3.1. Valoración total cuantitativa: 28

3.2. Opinión: FAVORABLE X DEBE MEJORAR _____

NO FAVORABLE _____

3.3. Observaciones: Sin observaciones

2

Tacna,

Mgtr. Gianfranco A. Málaga Tejada
Ingeniero de Sistemas - CIP 94021
Prof. Asociado DAIS/FAIN

Firma

	UNIVERSIDAD PRIVADA DE TACNA Escuela de Postrado Centro de Investigación Formato de Validación por expertos		
Codificación CEIN fve - 001	Versión 00	Vigencia 2015	Páginas 02

INFORME DE OPINIÓN DE EXPERTOS DEL INSTRUMENTO DE INVESTIGACIÓN

I. DATOS GENERALES:

1.1. Apellidos y nombres del informante (Experto): Nelson Abrahan Pablo Mollo Condori
1.2. Grado Académico: MAESTRO EN CIENCIAS
1.3 Profesión: INGENIERO EN INFORMATICA Y SISTEMAS
1.4. Institución donde labora: U.N. JORGE BASADRE GROHMANN
1.5. Cargo que desempeña: DOCENTE
1.6 Denominación del Instrumento:
Cuestionario sobre gestión de riesgos y calidad de software realizado por profesionales del CIP Tacna
1.7. Autor del instrumento: Enrique Waldo Condori Siles
1.8 Programa de postgrado: Ingeniería de Software

II. VALIDACIÓN

1

INDICADORES DE EVALUACIÓN DEL INSTRUMENTO	CRITERIOS Sobre los ítems del instrumento	Muy Malo 1	Malo 2	Regular 3	Bueno 4	Muy Bueno 5
1. CLARIDAD	Están formulados con lenguaje apropiado que facilita su comprensión					X
2. OBJETIVIDAD	Están expresados en conductas observables, medibles					X
3. CONSISTENCIA	Existe una organización lógica en los contenidos y relación con la teoría					X
4. COHERENCIA	Existe relación de los contenidos con los indicadores de la variable					X
5. PERTINENCIA	Las categorías de respuestas y sus valores son apropiados					X
6. SUFICIENCIA	Son suficientes la cantidad y calidad de ítems presentados en el instrumento				X	
	SUMATORIA PARCIAL					
	SUMATORIA TOTAL					29

	UNIVERSIDAD PRIVADA DE TACNA Escuela de Postrado Centro de Investigación Formato de Validación por expertos		
Codificación CEIN fve - 001	Versión 00	Vigencia 2015	Páginas 02

III. RESULTADOS DE LA VALIDACIÓN

3.1. Valoración total cuantitativa: 29

3.2. Opinión: FAVORABLE X DEBE MEJORAR _____

NO FAVORABLE _____

3.3. Observaciones: Sin observaciones

2

Tacna, 16 de Enero 2025

Firma

	UNIVERSIDAD PRIVADA DE TACNA Escuela de Postrado Centro de Investigación Formato de Validación por expertos		
Codificación CEIN fve - 001	Versión 00	Vigencia 2015	Páginas 02

INFORME DE OPINIÓN DE EXPERTOS DEL INSTRUMENTO DE INVESTIGACIÓN

I. DATOS GENERALES:

1.1. Apellidos y nombres del informante (Experto): Oliver Israel Saucedo Corbacho
1.2. Grado Académico: Magister
1.3 Profesión: Ingeniero de Sistemas
1.4. Institución donde labora: UNJBG
1.5. Cargo que desempeña: Docente
1.6 Denominación del Instrumento:
Cuestionario sobre gestión de riesgos y calidad de software realizada por profesionales del CIP Tacna
1.7. Autor del instrumento: Enrique Waldo Condori Siles
1.8 Programa de postgrado: Ingeniería de Software

II. VALIDACIÓN

1

INDICADORES DE EVALUACIÓN DEL INSTRUMENTO	CRITERIOS Sobre los ítems del instrumento	Muy Malo 1	Malo 2	Regular 3	Bueno 4	Muy Bueno 5
1. CLARIDAD	Están formulados con lenguaje apropiado que facilita su comprensión					X
2. OBJETIVIDAD	Están expresados en conductas observables, medibles				X	
3. CONSISTENCIA	Existe una organización lógica en los contenidos y relación con la teoría					X
4. COHERENCIA	Existe relación de los contenidos con los indicadores de la variable				X	
5. PERTINENCIA	Las categorías de respuestas y sus valores son apropiados					X
6. SUFICIENCIA	Son suficientes la cantidad y calidad de ítems presentados en el instrumento					X
	SUMATORIA PARCIAL					
	SUMATORIA TOTAL				28	

	UNIVERSIDAD PRIVADA DE TACNA Escuela de Postrado Centro de Investigación Formato de Validación por expertos		
Codificación CEIN fve - 001	Versión 00	Vigencia 2015	Páginas 02

III. RESULTADOS DE LA VALIDACIÓN

3.1. Valoración total cuantitativa: 28

3.2. Opinión: FAVORABLE X DEBE MEJORAR _____

NO FAVORABLE _____

3.3. Observaciones: Sin observaciones

2

Tacna, 16 de enero 2025

Firma

Anexo 04: Matriz de datos

N°	x1	x2	x3	x4	x5	x6	x7	x8	x9	y1	y2	y3	y4	y5	y6	y7	y8	y9	y10	y11	y12
1	3	3	4	3	4	3	3	4	3	3	3	3	3	4	4	4	4	3	3	4	4
2	4	4	5	3	4	4	3	2	3	3	2	3	3	3	3	3	3	3	3	3	3
3	4	4	5	2	4	4	4	5	4	3	4	4	4	4	5	4	4	3	4	4	2
4	2	1	4	3	4	4	2	2	3	2	2	3	4	4	2	4	4	3	2	2	2
5	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
6	3	3	3	2	2	4	4	3	4	4	3	4	4	4	3	4	4	4	4	4	4
7	3	4	4	3	3	4	4	3	3	3	4	4	4	3	2	3	4	3	2	3	3
8	3	4	4	2	3	4	3	4	3	4	4	4	3	4	3	3	4	3	4	5	3
9	1	1	1	1	1	1	1	1	1	3	2	4	4	1	5	4	2	2	1	3	1
10	2	3	2	3	3	3	3	3	3	4	2	3	3	3	2	3	2	3	2	2	2
11	3	2	3	2	2	4	2	2	2	2	1	4	2	2	4	2	2	2	2	2	2
12	4	4	4	4	4	4	4	5	3	3	3	3	3	4	4	4	5	4	4	4	3
13	3	4	4	3	4	4	4	3	4	4	2	2	3	2	4	3	3	3	2	2	2
14	3	2	2	2	4	4	2	3	3	3	4	4	4	3	4	4	4	3	3	4	4
15	4	3	3	4	3	4	4	4	4	5	4	5	4	4	3	4	4	4	4	4	4
16	4	4	4	3	3	4	4	3	4	4	3	4	4	4	4	4	3	4	3	3	3
17	4	4	4	3	4	4	3	3	3	4	4	4	4	4	4	4	4	4	4	4	4
18	2	2	2	3	3	3	3	4	4	3	3	4	4	4	4	4	4	3	3	3	4
19	2	2	2	2	2	3	2	3	3	1	1	1	1	1	2	1	1	1	1	1	2
20	2	4	3	3	3	3	3	3	3	4	4	4	4	4	4	4	4	2	3	4	4
21	3	4	4	1	4	1	4	1	1	4	4	5	5	4	3	4	4	4	4	3	4
22	4	3	4	3	4	5	3	4	3	4	5	5	4	5	5	4	4	4	4	4	4
23	2	3	3	3	4	4	3	3	2	4	3	4	4	3	3	4	4	3	4	4	3
24	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
25	4	4	4	4	3	3	4	4	4	4	3	4	4	4	4	4	3	4	3	4	3
26	4	2	2	2	4	2	2	4	4	2	4	4	4	4	4	4	4	2	4	4	2
27	4	5	5	4	4	5	4	4	3	4	5	4	4	5	2	5	4	3	4	5	4
28	4	4	3	3	3	3	4	4	4	4	3	4	5	4	2	4	4	3	3	4	2
29	3	2	3	2	3	4	4	4	4	5	3	4	3	4	3	4	4	4	4	4	4
30	4	4	3	2	3	4	4	4	4	4	4	4	4	4	4	3	4	4	3	4	4
31	3	3	4	3	2	4	2	4	3	4	4	4	4	4	4	4	4	4	3	2	4
32	4	3	3	4	3	4	3	3	3	4	3	4	4	4	3	4	4	4	3	4	4
33	3	3	3	2	3	3	4	4	3	3	2	4	4	3	3	4	2	4	3	2	3
34	4	4	5	3	2	3	4	4	4	4	5	4	4	4	3	5	4	4	3	4	3
35	3	4	4	2	4	4	4	4	4	4	4	4	4	4	3	3	4	4	4	4	4
36	1	2	2	2	3	2	4	2	2	2	2	2	2	4	2	4	4	2	4	2	3
37	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
38	3	1	3	4	5	4	3	3	2	2	1	2	1	1	2	1	2	1	2	3	2
39	2	2	3	3	4	4	4	3	4	4	3	3	4	3	4	4	3	4	4	3	4
40	3	3	4	3	3	3	3	3	3	4	4	2	4	4	4	4	4	4	4	3	4
41	4	4	4	4	4	4	4	4	4	4	5	4	5	5	4	5	4	2	4	3	4

42	2	4	3	3	3	3	3	3	3	3	3	3	3	4	4	4	4	3	3	4	4
43	3	4	4	1	4	1	4	1	1	3	2	4	4	1	5	4	2	2	1	3	1
44	4	3	4	3	4	5	3	4	3	4	5	4	4	4	3	5	4	4	3	4	3
45	2	3	3	3	4	4	3	3	2	3	4	4	4	3	2	3	4	3	2	3	3
46	3	3	3	2	2	4	4	3	4	3	3	4	4	4	4	4	4	3	3	3	4
47	3	4	4	3	3	4	4	3	3	4	4	4	4	4	4	4	4	4	3	2	4
48	4	2	2	2	4	2	2	4	4	3	2	4	4	3	3	4	2	4	3	2	3
49	4	5	5	4	4	5	4	4	3	4	5	5	4	5	5	4	4	4	4	4	4
50	3	4	4	2	3	4	3	4	3	4	3	4	4	4	3	4	4	4	4	4	4
51	3	2	3	2	3	4	4	4	4	4	4	4	3	4	3	3	4	3	4	5	3
52	4	3	3	4	3	4	3	3	3	5	3	4	3	4	3	4	4	4	4	4	4
53	3	3	4	3	2	4	2	4	3	3	3	4	4	4	4	4	4	3	3	3	4
54	4	3	3	4	3	4	3	3	3	4	4	4	3	4	3	3	4	3	4	5	3
55	3	2	3	2	3	4	4	4	4	4	4	4	3	4	3	3	4	3	4	5	3
56	4	4	4	3	3	4	4	3	4	4	5	4	4	4	3	5	4	4	3	4	3
57	3	2	3	2	3	4	4	4	4	4	4	2	4	4	4	4	4	4	4	3	4
58	5	4	5	5	4	4	5	4	4	4	4	5	5	5	4	4	4	4	4	4	3
59	3	4	4	1	4	1	4	1	1	2	1	2	1	1	2	1	2	1	2	3	2
60	3	4	4	2	4	4	4	4	4	4	5	4	4	4	3	5	4	4	3	4	3
61	1	2	2	2	3	2	4	2	2	2	2	2	2	2	2	2	2	2	2	2	2
62	5	4	5	5	4	4	5	4	4	4	4	5	5	5	4	4	4	4	4	4	3
63	3	1	3	4	5	4	3	3	2	4	3	3	4	3	4	4	3	4	4	3	4
64	2	2	3	3	4	4	4	3	4	3	3	3	3	4	4	4	5	4	4	4	3
65	3	3	4	3	3	3	3	3	3	4	3	4	4	3	3	4	4	3	4	4	3
66	4	4	4	4	4	4	4	4	4	4	5	4	4	5	2	5	4	3	4	5	4
67	3	3	4	3	4	3	3	4	3	3	3	3	3	4	4	4	5	4	4	4	3
68	3	2	3	2	3	4	4	4	4	3	4	4	4	4	5	4	4	3	4	4	2
69	4	4	5	2	4	4	4	5	4	4	5	4	4	5	2	5	4	3	4	5	4
70	2	1	4	3	4	4	2	2	3	2	2	3	4	4	2	4	4	3	2	2	2
71	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
72	3	3	3	2	2	4	4	3	4	4	3	3	4	3	4	4	3	4	4	3	4
73	3	4	4	3	3	4	4	3	3	4	3	4	4	4	3	4	4	4	3	4	4
74	3	2	3	2	2	4	2	2	2	2	1	2	1	1	2	1	2	1	2	3	2
75	4	4	4	4	3	3	4	4	4	4	4	5	5	4	3	4	4	4	4	3	4
76	5	4	5	5	4	4	5	4	4	4	4	5	5	5	4	4	4	4	4	4	3
77	4	5	5	4	4	5	4	4	3	4	5	5	4	5	5	4	4	4	4	4	4
78	2	2	3	3	4	4	4	3	4	4	4	4	4	4	4	4	4	2	3	4	4
79	3	2	3	2	3	4	4	4	4	3	4	4	4	3	4	4	4	3	3	4	4
80	3	1	3	4	5	4	3	3	2	4	5	4	4	4	3	5	4	4	3	4	3
81	3	2	3	2	2	4	2	2	2	2	1	2	1	1	2	1	2	1	2	3	2
82	4	4	4	4	3	3	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
83	5	4	5	5	4	4	5	4	4	4	5	4	5	5	4	5	4	2	4	3	4
84	4	5	5	4	4	5	4	4	3	4	5	5	4	5	5	4	4	4	4	4	4
85	3	3	4	3	2	4	2	4	3	4	4	4	4	4	4	3	4	4	3	4	4
86	3	2	3	2	3	4	4	4	4	3	3	4	4	4	4	4	4	3	3	3	4

87	3	3	3	2	3	3	4	4	3	4	4	4	4	4	3	3	4	4	4	4	4
88	3	3	3	2	2	4	4	3	4	5	3	4	3	4	3	4	4	4	4	4	4
89	4	4	5	2	4	4	4	5	4	4	5	4	4	5	2	5	4	3	4	5	4
90	2	1	4	3	4	4	2	2	3	2	2	3	4	4	2	4	4	3	2	2	2
91	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
92	3	3	3	2	2	4	4	3	4	4	3	4	4	3	3	4	4	3	4	4	3
93	3	4	4	3	3	4	4	3	3	4	4	4	4	4	4	4	4	2	3	4	4
94	2	1	4	3	4	4	2	2	3	2	2	3	4	4	2	4	4	3	2	2	2
95	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
96	4	4	4	4	4	4	4	5	3	4	5	4	5	5	4	5	4	2	4	3	4
97	3	3	3	2	3	3	4	4	3	4	4	4	4	4	4	3	4	4	3	4	4
98	5	4	5	5	4	4	5	4	4	4	5	4	4	5	2	5	4	3	4	5	4
99	3	3	4	3	3	3	3	3	3	5	3	4	3	4	3	4	4	4	4	4	4
100	3	3	4	3	3	3	3	3	3	4	4	4	4	4	4	3	4	4	3	4	4
101	5	4	5	5	4	4	5	4	4	4	5	4	4	5	2	5	4	3	4	5	4
102	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
103	5	4	5	5	4	4	5	4	4	4	5	4	4	5	2	5	4	3	4	5	4
104	3	1	3	4	5	4	3	3	2	4	3	4	4	4	4	4	3	4	3	3	3
105	3	4	4	2	4	4	4	4	4	4	3	4	4	4	3	4	4	4	4	4	4
106	4	4	4	4	3	3	4	4	4	4	4	5	5	4	3	4	4	4	4	3	4
107	5	4	5	5	4	4	5	4	4	4	5	4	5	5	4	5	4	2	4	3	4
108	4	5	5	4	4	5	4	4	3	4	5	5	4	5	5	4	4	4	4	4	4
109	4	3	4	3	4	5	3	4	3	5	3	4	3	4	3	4	4	4	4	4	4
110	3	2	3	2	3	4	4	4	4	4	3	4	4	3	3	4	4	3	4	4	3
111	3	2	3	2	3	4	4	4	4	3	4	4	4	4	5	4	4	3	4	4	2
112	3	3	4	3	2	4	2	4	3	3	3	4	4	4	4	4	4	3	3	3	4
113	4	3	3	4	3	4	3	3	3	3	3	3	3	4	4	4	5	4	4	4	3
114	3	1	3	4	5	4	3	3	2	4	3	3	4	3	4	4	3	4	4	3	4
115	2	2	3	3	4	4	4	3	4	4	3	4	4	3	3	4	4	3	4	4	3
116	3	3	4	3	3	3	3	3	3	3	3	3	3	4	4	4	4	3	3	4	4
117	4	4	4	4	4	4	4	4	4	5	4	5	4	4	3	4	4	4	4	4	4
118	3	3	4	3	4	3	3	4	3	3	4	4	4	3	4	4	4	3	3	4	4
119	3	2	3	2	3	4	4	4	4	4	4	4	4	4	4	4	4	4	3	2	4
120	4	4	5	2	4	4	4	5	4	5	4	5	4	4	3	4	4	4	4	4	4
121	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
122	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
123	3	3	3	2	2	4	4	3	4	3	3	3	3	4	4	4	4	3	3	4	4
124	3	4	4	3	3	4	4	3	3	4	4	4	4	4	4	4	4	2	3	4	4
125	3	4	4	2	3	4	3	4	3	3	4	4	4	3	4	4	4	3	3	4	4
126	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
127	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
128	3	2	3	2	2	4	2	2	2	4	2	3	3	3	2	3	2	3	2	2	2
129	4	4	4	4	4	4	4	5	3	4	5	4	4	5	2	5	4	3	4	5	4
130	3	3	4	3	4	3	3	4	3	4	4	4	4	4	4	3	4	4	3	4	4
131	3	4	4	2	3	4	3	4	3	4	3	4	4	4	3	4	4	4	4	4	4

132	4	3	3	4	3	4	3	3	3	4	4	4	4	4	4	3	4	4	3	4	4
133	3	4	4	2	3	4	3	4	3	5	3	4	3	4	3	4	4	4	4	4	4
134	1	2	2	2	3	2	4	2	2	2	2	2	2	2	2	2	2	2	2	2	2
135	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
136	3	1	3	4	5	4	3	3	2	2	4	4	4	4	4	4	4	2	4	4	2
137	2	2	3	3	4	4	4	3	4	4	3	4	4	3	3	4	4	3	4	4	3
138	3	3	4	3	2	4	2	4	3	4	3	4	5	4	2	4	4	3	3	4	2
139	4	3	3	4	3	4	3	3	3	4	4	4	3	4	3	3	4	3	4	5	3
140	3	3	3	2	3	3	4	4	3	2	4	4	4	4	4	4	4	2	4	4	2
141	4	3	3	4	3	4	3	3	3	4	4	4	4	4	3	3	4	4	4	4	4
142	3	4	4	2	3	4	3	4	3	4	3	4	4	4	4	4	3	4	3	4	3
143	5	4	5	5	4	4	5	4	4	4	5	4	5	5	4	5	4	2	4	3	4
144	3	2	3	2	3	4	4	4	4	4	3	4	4	4	3	4	4	4	4	4	4
145	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
146	3	3	3	2	2	4	4	3	4	4	3	4	5	4	2	4	4	3	3	4	2
147	3	4	4	3	3	4	4	3	3	4	4	4	4	4	4	4	4	2	3	4	4
148	3	4	4	2	3	4	3	4	3	4	3	4	4	4	4	4	3	4	3	4	3
149	5	4	5	5	4	4	5	4	4	5	4	5	4	4	3	4	4	4	4	4	4
150	3	2	3	2	3	4	4	4	4	5	3	4	3	4	3	4	4	4	4	4	4
151	3	2	3	2	2	4	2	2	2	4	2	3	3	3	2	3	2	3	2	2	2
152	4	4	4	4	4	4	4	5	3	4	5	4	5	5	4	5	4	2	4	3	4
153	2	2	3	3	4	4	4	3	4	4	4	4	4	4	4	3	4	4	3	4	4
154	3	2	2	2	4	4	2	3	3	2	2	2	2	4	2	4	4	2	4	2	3
155	3	2	3	2	3	4	4	4	4	5	3	4	3	4	3	4	4	4	4	4	4
156	3	2	3	2	3	4	4	4	4	4	4	4	4	4	4	3	4	4	3	4	4
157	3	2	3	2	3	4	4	4	4	5	3	4	3	4	3	4	4	4	4	4	4
158	4	4	4	3	3	4	4	3	4	4	4	4	4	4	3	3	4	4	4	4	4
159	3	2	3	2	3	4	4	4	4	4	4	2	4	4	4	4	4	4	4	3	4
160	5	4	5	5	4	4	5	4	4	5	4	5	4	4	3	4	4	4	4	4	4
161	3	2	3	2	2	4	2	2	2	1	1	1	1	1	2	1	1	1	1	1	2
162	2	4	3	3	3	3	3	3	3	3	4	4	4	3	2	3	4	3	2	3	3
163	3	4	4	1	4	1	4	1	1	4	2	2	3	2	4	3	3	3	2	2	2
164	4	3	4	3	4	5	3	4	3	4	3	4	4	4	3	4	4	4	4	4	4
165	3	3	3	2	3	3	4	4	3	3	3	3	3	4	4	4	4	3	3	4	4
166	4	4	5	3	2	3	4	4	4	5	3	4	3	4	3	4	4	4	4	4	4
167	3	4	4	2	4	4	4	4	4	4	4	4	4	4	4	3	4	4	3	4	4
168	1	2	2	2	3	2	4	2	2	2	2	2	2	2	2	2	2	2	2	2	2
169	5	4	5	5	4	4	5	4	4	4	4	4	4	4	4	4	4	4	4	4	4
170	4	4	4	3	4	4	3	3	3	4	4	2	4	4	4	4	4	4	4	3	4
171	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
172	2	2	2	2	2	3	2	3	3	2	1	4	2	2	4	2	2	2	2	2	2
173	2	4	3	3	3	3	3	3	3	3	4	4	4	3	2	3	4	3	2	3	3
174	3	4	4	1	4	1	4	1	1	4	2	2	3	2	4	3	3	3	2	2	2
175	4	3	4	3	4	5	3	4	3	5	3	4	3	4	3	4	4	4	4	4	4
176	2	3	3	3	4	4	3	3	2	3	4	4	4	3	2	3	4	3	2	3	3

177	2	2	2	2	2	3	2	3	3	2	1	4	2	2	4	2	2	2	2	2	2
178	2	4	3	3	3	3	3	3	3	3	4	4	4	3	2	3	4	3	2	3	3
179	3	4	4	1	4	1	4	1	1	4	2	2	3	2	4	3	3	3	2	2	2
180	4	3	4	3	4	5	3	4	3	4	4	4	4	4	4	3	4	4	3	4	4
181	2	3	3	3	4	4	3	3	2	3	4	4	4	3	2	3	4	3	2	3	3
182	3	3	4	3	4	3	3	4	3	4	3	4	4	4	4	4	3	4	3	4	3
183	4	4	5	3	4	4	3	2	3	3	4	4	4	4	5	4	4	3	4	4	2
184	4	4	5	2	4	4	4	5	4	4	5	4	4	5	2	5	4	3	4	5	4
185	2	1	4	3	4	4	2	2	3	2	2	2	2	4	2	4	4	2	4	2	3
186	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
187	2	3	3	3	4	4	3	3	2	3	4	4	4	3	2	3	4	3	2	3	3
188	4	3	4	3	4	5	3	4	3	4	3	4	4	4	3	4	4	4	4	4	4
189	4	4	4	4	3	3	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
190	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
191	4	5	5	4	4	5	4	4	3	4	5	5	4	5	5	4	4	4	4	4	4
192	4	4	3	3	3	3	4	4	4	4	4	4	4	4	4	4	4	4	3	2	4
193	3	3	4	3	3	3	3	3	3	2	4	4	4	4	4	4	4	2	4	4	2
194	4	4	4	4	4	4	4	4	4	4	5	4	5	5	4	5	4	2	4	3	4
195	3	3	4	3	4	3	3	4	3	3	3	3	3	4	4	4	5	4	4	4	3
196	4	4	5	3	4	4	3	2	3	4	3	4	4	4	3	4	4	4	3	4	4
197	4	4	5	2	4	4	4	5	4	5	4	5	4	4	3	4	4	4	4	4	4
198	2	3	3	3	4	4	3	3	2	3	4	4	4	3	2	3	4	3	2	3	3
199	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
200	4	4	4	4	3	3	4	4	4	4	4	5	5	4	3	4	4	4	4	3	4
201	4	2	2	2	4	2	2	4	4	3	2	3	3	3	3	3	3	3	3	3	3
202	4	5	5	4	4	5	4	4	3	4	4	5	5	5	4	4	4	4	4	4	3
203	4	4	3	3	3	3	4	4	4	3	4	4	4	4	5	4	4	3	4	4	2
204	2	2	3	3	4	4	4	3	4	4	3	4	4	3	3	4	4	3	4	4	3
205	3	3	4	3	3	3	3	3	3	4	3	4	5	4	2	4	4	3	3	4	2
206	4	4	4	4	4	4	4	4	4	5	4	5	4	4	3	4	4	4	4	4	4
207	3	3	4	3	4	3	3	4	3	5	3	4	3	4	3	4	4	4	4	4	4
208	4	4	5	3	4	4	3	2	3	4	4	4	4	4	4	4	4	4	3	2	4
209	3	4	4	2	3	4	3	4	3	4	4	4	4	4	4	3	4	4	3	4	4
210	2	2	2	2	2	3	2	3	3	1	1	1	1	1	2	1	1	1	1	1	2
211	2	3	2	3	3	3	3	3	3	3	2	4	4	1	5	4	2	2	1	3	1
212	3	2	3	2	2	4	2	2	2	4	2	3	3	3	2	3	2	3	2	2	2
213	4	4	4	4	4	4	4	5	3	4	4	4	4	4	4	4	4	4	4	4	4
214	2	1	4	3	4	4	2	2	3	3	2	4	4	1	5	4	2	2	1	3	1
215	2	5	4	4	4	4	5	5	4	4	4	5	5	5	4	4	4	4	4	4	3
216	3	3	3	2	2	4	4	3	4	2	4	4	4	4	4	4	4	2	4	4	2
217	3	4	4	3	3	4	4	3	3	3	4	4	4	4	5	4	4	3	4	4	2
218	3	2	3	2	3	4	4	4	4	4	3	4	4	4	4	4	3	4	3	3	3
219	4	4	3	2	3	4	4	4	4	4	3	4	4	4	3	4	4	4	3	4	4
220	3	3	4	3	2	4	2	4	3	4	3	4	5	4	2	4	4	3	3	4	2
221	4	3	3	4	3	4	3	3	3	4	3	4	4	4	3	4	4	4	4	4	4

222	3	3	3	2	3	3	4	4	3	2	4	4	4	4	4	4	4	2	4	4	2
223	4	4	5	3	2	3	4	4	4	4	4	4	4	4	3	3	4	4	4	4	4
224	3	4	4	3	4	4	4	3	4	4	3	4	4	4	3	4	4	4	4	4	4
225	3	2	2	2	4	4	2	3	3	4	2	2	3	2	4	3	3	3	2	2	2
226	4	3	3	4	3	4	4	4	4	5	3	4	3	4	3	4	4	4	4	4	4
227	4	4	4	3	3	4	4	3	4	4	4	4	4	4	4	3	4	4	3	4	4
228	4	4	4	3	4	4	3	3	3	4	4	2	4	4	4	4	4	4	4	3	4
229	2	2	2	3	3	3	3	4	4	2	2	3	4	4	2	4	4	3	2	2	2
230	3	4	4	2	4	4	4	4	4	4	3	4	4	4	3	4	4	4	4	4	4
231	1	2	2	2	3	2	4	2	2	2	2	2	2	2	2	2	2	2	2	2	2
232	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
233	3	1	3	4	5	4	3	3	2	4	3	4	5	4	2	4	4	3	3	4	2
234	2	2	3	3	4	4	4	3	4	4	3	3	4	3	4	4	3	4	4	3	4
235	2	2	2	2	2	3	2	3	3	2	2	2	2	2	2	2	2	2	2	2	2
236	2	4	3	3	3	3	3	3	3	3	2	4	4	3	3	4	2	4	3	2	3
237	3	4	4	1	4	1	4	1	1	3	2	4	4	1	5	4	2	2	1	3	1
238	3	3	4	3	4	3	3	4	3	4	4	4	4	4	3	3	4	4	4	4	4
239	4	4	5	2	4	4	4	5	4	4	4	4	4	4	4	4	4	4	4	4	4
240	2	1	4	3	4	4	2	2	3	3	2	4	4	1	5	4	2	2	1	3	1
241	2	5	4	4	4	4	5	5	4	4	5	4	5	5	4	5	4	2	4	3	4
242	3	3	3	2	2	4	4	3	4	3	3	3	3	4	4	4	4	3	3	4	4
243	3	4	4	3	3	4	4	3	3	4	4	4	4	4	4	4	4	2	3	4	4
244	3	4	4	2	3	4	3	4	3	5	3	4	3	4	3	4	4	4	4	4	4
245	2	2	2	2	2	3	2	3	3	1	1	1	1	1	2	1	1	1	1	1	2
246	3	2	3	2	3	4	4	4	4	4	3	4	4	4	4	4	3	4	3	3	3
247	4	4	3	2	3	4	4	4	4	3	4	4	4	4	5	4	4	3	4	4	2
248	3	3	4	3	2	4	2	4	3	3	3	3	3	4	4	4	4	3	3	4	4
249	4	3	3	4	3	4	3	3	3	3	4	4	4	3	4	4	4	3	3	4	4
250	3	3	3	2	3	3	4	4	3	2	4	4	4	4	4	4	4	2	4	4	2
251	3	4	4	2	3	4	3	4	3	4	3	4	4	4	3	4	4	4	4	4	4
252	4	3	3	4	3	4	3	3	3	4	4	4	4	4	3	3	4	4	4	4	4
253	3	4	4	1	4	1	4	1	1	2	1	2	1	1	2	1	2	1	2	3	2
254	5	4	5	5	4	4	5	4	4	5	5	5	5	5	4	5	5	4	5	5	5
255	3	1	3	4	5	4	3	3	2	4	3	4	5	4	2	4	4	3	3	4	2