Una revisión de Trazabilidad de Requerimientos en el contexto de Gestión de Proyectos

Pocladova Victoria[1] , Menchón Magalí [1] ,Olsowy Verena[2] ,
Antonelli Leandro[3] , Thomas Pablo[2]
[1] UNLP Informática, Buenos Aires, Argentina
[2] Instituto de Investigación en Informática LIDI (III-LIDI). Facultad de Informática, Universidad Nacional de La Plata.
Centro Asociado CIC. Buenos Aires, Argentina.
[3] LIFIA, Facultad de Informática, Universidad Nacional de La Plata.
CAETI- Facultad de Tecnología Informática- Universidad Abierta Interamericana
{victoria.pocladova, magalimenchon} @gmail.com
volsowy@lidi.info.unlp.edu.ar
leandro.antonelli@lifia.info.unlp.edu.ar
pthomas@lidi.info.unlp.edu.ar

Abstract. En la ingeniería de requerimientos, la "trazabilidad" se refiere a la capacidad de seguir y comprender los efectos de los cambios en las especificaciones y requerimientos hacia otros elementos del proyecto. Según el Project Management Institute (PMI), el "alcance" se establece como una de las tres restricciones principales, lo que subraya la importancia de evaluar adecuadamente el impacto de los cambios en los requerimientos y en los productos finales mediante una trazabilidad efectiva. El Capability Maturity Model Integration (CMMI)[1] enfatiza en la práctica SP 1.4-2 la necesidad de mantener una trazabilidad bidireccional de los requerimientos, integrando productos de trabajo típicos en la matriz de trazabilidad y en los sistemas de seguimiento de requerimientos. Este trabajo presenta una revisión bibliográfica de las concepciones y prácticas de la inclusión de la trazabilidad de requerimientos, en el contexto de la gestión de proyectos de software.

Keywords: trazabilidad, gestión de proyectos de software, gestión de requerimientos

1. Introducción

La gestión de proyectos de software es crucial debido a las restricciones de presupuesto y tiempo. Los administradores de proyectos deben garantizar que se cumpla con estas restricciones y se entregue un producto de alta calidad. Una gestión eficiente, aunque no garantiza el éxito, reduce el riesgo de retrasos, costos excesivos y expectativas incumplidas. Los criterios de éxito incluyen la entrega oportuna del software, el control de costos, la satisfacción del cliente y un equipo de desarrollo eficaz. La triple restricción, también conocida como el triángulo de la gestión de

proyectos de software, abarca tres limitaciones principales: alcance, tiempo y costo. Estos elementos están interrelacionados, y cualquier cambio en uno afecta inevitablemente a los otros dos. Por ello, deben equilibrarse cuidadosamente, siendo clave la trazabilidad para rastrear los efectos de los cambios en especificaciones y requerimientos [2, 3]. Según el PMI, una adecuada trazabilidad permite dimensionar el impacto de los cambios en los requerimientos y productos finales [4]. El CMMI también destaca la importancia de mantener una trazabilidad bidireccional de los requerimientos para asegurar una gestión eficaz.

Complementariamente, existen diversas herramientas y metodologías disponibles para apoyar la gestión de proyectos de software, como el enfoque Agile, herramientas de gestión de tareas y proyectos, sistemas de control de versiones y plataformas de colaboración en línea.

Este artículo presenta un mapeo sistemático sobre la trazabilidad de requerimientos en proyectos de software.

El trabajo se estructura de la siguiente forma: en la sección 2 se introduce el concepto de trazabilidad en la gestión de proyectos. En la sección 3 se presentan los trabajos relacionados. En la sección 4 se detallan las preguntas de investigación, el protocolo de búsqueda y los criterios de inclusión/exclusión. En la sección 5 se resumen los enfoques principales y cómo integran la trazabilidad. Finalmente, se presenta la discusión final, conclusiones y trabajos futuros.

2. Trazabilidad en la gestión de proyectos

La trazabilidad en el desarrollo de software se refiere a la capacidad de seguir y documentar cada etapa del ciclo de vida del software, desde los requerimientos iniciales hasta su implementación y mantenimiento. Esto permite una visión clara y comprensible de cómo se desarrolló y evolucionó el software, facilitando la identificación de problemas y asegurando que todas las partes del sistema estén alineadas con los objetivos y requerimientos establecidos.

Existen varias técnicas que abordan la trazabilidad, destacando la gestión de requerimientos, el uso de herramientas de gestión de configuraciones y la integración continua [4,5].

Se entiende a la Gestión de Configuración, como el proceso de manejar cambios en el proyecto de manera sistemática a través de herramientas de control de versiones y repositorios de código [4]. Además, la Integración Continua (CI) y Entrega Continua (CD) permiten la integración y entrega frecuente de código. Facilitan la trazabilidad al integrar cambios continuamente y permitir pruebas automáticas que verifican que los requerimientos se cumplan con cada nueva versión [5,6].

Otra técnica es el uso de Matrices de Trazabilidad. El PMBOK define la matriz de trazabilidad como un cuadro que vincula los requerimientos del producto desde su origen hasta los entregables que los satisfacen [2,7].

La importancia de mantener una buena trazabilidad radica en su capacidad para mejorar la calidad y confiabilidad del software, así como en la facilitación del cumplimiento normativo y auditorías. Una trazabilidad adecuada permite a los equipos de desarrollo entender el impacto de los cambios, identificar y corregir

errores rápidamente y asegurarse que todos los componentes del sistema estén correctamente alineados con los requerimientos iniciales. Además, es fundamental en entornos regulados, donde es necesario cumplir con los procedimientos y estándares requeridos para garantizar la seguridad y efectividad del software.

3. Trabajos Relacionados

Diversos estudios han abordado la trazabilidad de requerimientos desde distintas perspectivas. En [25] realizaron una revisión sistemática sobre las tecnologías de trazabilidad y su aplicación en la ingeniería de requerimientos, destacando la importancia de la trazabilidad en el soporte a la toma de decisiones durante el proceso de desarrollo de software. En [26] centraron el estudio de la trazabilidad entre la arquitectura de software y el código fuente, subrayando la necesidad de mantener la coherencia entre los artefactos de alto nivel y su implementación. En [27] exploraron el papel de la trazabilidad en sistemas heterogéneos, proponiendo una serie de prácticas y herramientas que permiten mejorar el seguimiento de los requerimientos en entornos de desarrollo complejos. Estos trabajos proporcionan una base para entender cómo diferentes enfoques y herramientas pueden mejorar la trazabilidad en proyectos de software, lo cual complementa la presente revisión.

El presente trabajo se enfoca en la trazabilidad de requerimientos en el contexto específico de la gestión de proyectos de software, proporcionando una visión actualizada y comparativa de los enfoques más recientes, y explorando su aplicación en equipos ágiles y distribuidos geográficamente.

4. Método de investigación

Considerando que la trazabilidad es una práctica propuesta por el CMMI desde sus ediciones tempranas, en este trabajo se optó utilizar la metodología de mapeo sistemático, definiendo los criterios de selección detallados a lo largo de esta sección.

Para abordar el estudio se plantearon las siguientes preguntas de investigación:

PI1: ¿Qué consideraciones se deben tener en cuenta para implementar la trazabilidad en un equipo ágil?
PI2: ¿Es viable implementar la trazabilidad sin una documentación exhaustiva, adhiriéndose al principio de documentar solo lo necesario?
PI3: ¿Es posible conservar la trazabilidad en equipos distribuidos geográficamente?
PI4: ¿Qué herramientas y procesos pueden ayudar?
PI5: ¿Qué prácticas de agilidad se pueden aprovechar para reforzar la trazabilidad?
PI6: ¿De qué manera invertir tiempo en la trazabilidad puede contribuir a ser más eficientes?

Para la búsqueda de los artículos, se han consultado diversos repositorios, los cuales son SEDICI[8], IEEE[9], Google Scholar[10] y ResearchGate[11]. Para una primera selección, se hizo uso de las siguientes palabras claves con la unión de subcadenas de texto: *trazabilidad de requerimientos, gestión de proyectos de software, trazabilidad hacia adelante y hacia atrás, gestión de riesgos*.

Se obtuvieron 20 artículos de la búsqueda realizada, de los cuales finalmente se seleccionaron 13 que cumplen con los criterios planteados en la tabla 1. El proceso de búsqueda y selección es reflejado en la tabla 2.

Tabla 1. Criterios de inclusión de material bibliográfico

Orden	Criterio
1º	Revisión del título, palabras claves y abstract
2º	Idioma español o inglés
3º	Contenga información relacionada a la trazabilidad de requerimientos
4º	Enfocados en la gestión de software
5º	Contenido relacionado a las preguntas de investigación
6º	Literatura contemporánea con las metodologías ágiles, mediados de 1990 en adelante

Tabla 2. Proceso de búsqueda y selección

Base de datos	Primera selección	Segunda selección
SEDICI	9	5
IEEE	5	4
Google Scholar	3	2
ResearchGate	3	2

En la sección 5, se presenta un breve resumen sobre distintas publicaciones seleccionadas para responder a las preguntas de investigación.

5. Análisis de trabajos seleccionados

El enfoque de Klein [12] se basa en los conceptos establecidos en el Manifiesto de Desarrollo Ágil, habiendo aspectos del desarrollo tradicional que también forman parte de él. El proceso contiene seis etapas, tres de las cuales tienen un documento asociado que es firmado por el cliente para garantizar la trazabilidad de los requerimientos: el documento de “Iniciación”, “Definición preliminar” y “Revisión”.

Se basa en los siguiente principios:

1) documentar solo lo necesario 2) software funcionando es la medida principal de progreso 3) conversación fluida 4) flexibilidad ante el cambio 5) validar la calidad durante todo el proceso 6) diseño incremental.

Un aporte diferencial de la tesis es que demuestra la viabilidad de adaptar la práctica de trazabilidad con un equipo muy reducido y distribuidos geográficamente.

Los autores Vera, Hadad y Doorn [13] describen un mecanismo de generación de trazas que acompañe a cada actividad del proceso de requerimientos junto con un mecanismo de versionado de trazas con el fin de mantener la historia de los cambios.

Definen el tipo de traza a utilizar, existiendo dos tipos posibles: trazas incrementales y trazas largas.

En cuanto al campo del versionado, existe una gran variedad de estrategias, que abarcan desde la forma de generación de versiones hasta el mantenimiento y control de cada cambio. La estrategia debe determinar cómo guardar cada cambio realizado

sobre cada elemento, y para ello existen distintas formas de almacenamiento de cambios. Debe analizarse la relación beneficio-costo de cada opción para entender cual es conveniente aplicar particularmente en cada proyecto.

Bersano en [14] plantea “Una de las principales tareas para la correcta finalización de un proyecto es la gestión de requerimientos, debido a que a partir de esto se definirá el alcance final y preciso del software a construir. Uno de los resultados de esta gestión es la matriz de trazabilidad de los requerimientos, la cual brinda las relaciones existentes entre ellos. A medida que la cantidad de los requerimientos aumenta se hace más dificultosa su gestión sin una asistencia adecuada. Las numerosas relaciones entre los requerimientos deben gestionarse manualmente, tanto sobre los iniciales del proyecto como los que surjan a lo largo de la vida del software.”

En el marco de estas problemáticas se propone desarrollar una herramienta Web que brinde una ayuda semiautomática en la realización de la gestión de los requerimientos y trazabilidad de un proyecto de software, mediante la utilización de técnicas de búsqueda y recuperación de información (Information Retrieval).

Bersano [14] creó el UML del módulo de trazabilidad. El mismo contiene el componente innovador propuesto, estableciendo las relaciones de trazabilidad de requerimientos, contando con documentos relacionados (tanto casos de test como otros requerimientos) a partir de su contenido textual.

Guerra en [15] define que la trazabilidad de requerimientos en un desarrollo ágil es de suma importancia, al ser esencial para garantizar la calidad, la eficiencia y el éxito de los proyectos de software y proporcionar una visión clara y coherente de cómo se relacionan los requerimientos con la implementación y evolución del sistema.

La propuesta puede contribuir significativamente de las siguientes maneras:

- Claridad y Transparencia: Mejora la comprensión y comunicación dentro del equipo de desarrollo.
- Gestión de Cambios: Facilita la identificación y seguimiento de los cambios en los requerimientos a lo largo del ciclo de vida del software.
- Mejora Continua: Permite una evaluación constante de la cobertura de los requerimientos en el código fuente y en las pruebas.
- Demostración de Cumplimiento: Proporciona una base sólida para demostrar el cumplimiento con los requerimientos establecidos.
- Mantenimiento y Evolución: Facilita la implementación de cambios y mejoras en el software a lo largo de su ciclo de vida.

Se concluye que esta propuesta puede ayudar a mejorar la eficiencia y calidad de los proyectos de software al brindar un marco estructurado y eficaz para la gestión de requerimientos, lo que a su vez promueve una mayor transparencia, comunicación, adaptabilidad y control en el proceso de desarrollo.

Los autores Gotel y Finkelstein en [16] examinan la trazabilidad de requerimientos, destacando su importancia antes y después de la especificación de requerimientos. Mediante estudios empíricos, los autores identifican desafíos y barreras en las prácticas actuales de trazabilidad.

1. Herramientas de propósito especial. 2. Workbenches. 3. Environments.

Como parte del desarrollo se propone [16]:

1. Centrarse en la trazabilidad de la especificación de requerimientos previos: Los esfuerzos de investigación deberían reorientarse en la trazabilidad de la especificación de requerimientos previos.
2. Modelar la infraestructura social: Debe reflejar todos los cambios y facilitar la rápida ubicación y acceso de las personas involucradas en la especificación y refinamiento de los requerimientos.

Jacobsson en [17] parte de investigar cómo se suponía que debía realizarse la trazabilidad en los métodos ágiles, analizando estas preguntas básicas: ¿Necesitamos realmente trazabilidad?, ¿Puede la trazabilidad ser ágil?, ¿Se puede agregar sin demasiada sobrecarga administrativa?, ¿Debe el equipo escribir documentos que corren el riesgo de no estar actualizados?, ¿Es necesario que el rastreo sea costoso?, ¿Puede el proyecto ganar algo con la trazabilidad?, ¿Es mejor no tener información de trazabilidad que tener información incorrecta?

El objetivo fue investigar si se puede agregar la trazabilidad a las tecnologías ágiles, y en caso afirmativo, cómo podría hacerse. Se expone [17] cuáles serían los métodos a incorporar para lograr la trazabilidad en entornos ágiles.

Para finalizar destaca que “Hay un dicho: "El todo es mayor que la suma de las partes" cuando se trata de trazabilizar. Cada técnica individual descrita anteriormente posee algunas ventajas y desventajas. Sin embargo, cuando se combinan las técnicas, el equipo (y en la mayoría de los casos, todo el proyecto) ganará más.

Los autores Rempel, Mçder, y Kuschke. en [18] investigan el impacto de las estrategias de trazabilidad específicas de proyectos en el desarrollo de software y la calidad del producto. En la práctica, muchos profesionales no siguen estrategias explícitas de trazabilidad, lo que puede llevar a problemas significativos.

Para abordar este problema, se realizó un estudio cualitativo mediante entrevistas con profesionales de 17 empresas. Se observó que la calidad general de las estrategias de trazabilidad sugiere la necesidad de definirlas desde el principio del proyecto. Además, se destaca que la decisión de incluir o no relaciones de trazabilidad entre artefactos requiere una comprensión detallada del proceso de ingeniería y de los objetivos del proyecto de software.

Se concluye que es esencial contar con un procedimiento orientado a objetivos para evaluar y definir estrategias de trazabilidad. Estrategias bien definidas desde el inicio del proyecto son cruciales para alcanzar una trazabilidad efectiva que apoye las necesidades específicas del proyecto.

Los autores Cleland-Huang, Chang y Christensen en [19] abordan los desafíos asociados con la trazabilidad de requerimientos en el desarrollo de software, especialmente en contextos donde los sistemas evolucionan constantemente. Por ello, proponen un método de trazabilidad basado en notificaciones de eventos, llamado "Event-Based Traceability" (EBT). En este enfoque, los artefactos de software no están estrechamente acoplados; en su lugar, están vinculados a través de un servicio de eventos. Cuando se realiza un cambio en un requisito u otro artefacto, se publica un evento en un servidor de eventos, y las notificaciones se envían a todos los artefactos dependientes. Este mecanismo permite manejar los cambios de manera más eficiente y mantener los artefactos y sus enlaces en un estado reversible.

Los autores Lago, Muccini y Van Vliet en [20] proponen un enfoque para personalizar la trazabilidad mediante la definición de caminos de trazabilidad necesarios para las actividades específicas de los interesados. Presentan dos ejemplos: derivación de productos en líneas de productos de software y planificación de lanzamientos en la gestión de procesos de software. Se demuestra que el enfoque puede ser utilizado para apoyar tanto la trazabilidad de productos como de procesos de manera pragmática y eficiente.

Los autores Dapozo, Greiner, Irrazábal, Medina, Ferraro, Lencina, Chiapello y Mascheroni en [21] plantean que "el desafío es sin duda, acompañar la gestión de cambios con el suficiente nivel de trazabilidad que logre el equilibrio entre calidad de software y tiempo dedicado a la gestión." Así analiza diferentes líneas de investigación, donde se concluye que: [21]

1. Estimación: mientras más específico es el método mejor es el ajuste y una mayor disponibilidad de datos históricos mejora la precisión para estimar.
2. Entrega Continua: la funcionalidad es la característica percibida como más importante. Esto tiene relación directa con los tipos de pruebas que se realizan de manera habitual antes de una entrega de software: las pruebas funcionales.
3. Gestión cuantitativa: en particular, la gestión de incidentes y bugs, contribuirá a la eficiencia en los procesos de desarrollo.
4. Trazabilidad: se determinó mediante encuestas que existe una carencia de trazabilidad de requerimientos en las empresas.

Los autores Kattz Villa, Martínez Moreno, Vergara Camacho y Morales Velázquez en [22] presentan una revisión sistemática de la literatura para relevar un análisis empresarial sobre Matriz de Trazabilidad de Requerimientos (RTM), en busca de definir una matriz estandarizada "compuesta de conceptos que son claves y precisos para el seguimiento de los requerimientos, agilizando y haciendo que se produzcan productos de software de calidad".

Si bien establecen que cada proyecto debe diseñar su propia herramienta en función de los requerimientos establecidos en las fases de diseño y de la complejidad de la ejecución de las tareas [22] presentan que la diferente literatura relevada posee los siguientes puntos en común, de los cuales elaboraron la matriz, incluyendo: Requerimiento, Casos de prueba asociados, Prioridad, Artefactos Asociados, Estado, Situación del requerimiento, Responsable y Fecha de entrega.

Se concluye que implementar en la trazabilidad de requerimientos una RTM estandarizada beneficia a la gestión de un proyecto de software "ya que permite una mejor planificación, seguimiento y control, lo que se traduce en un proyecto más eficiente, eficaz y exitoso."

Los autores Roldán, Vegetti, Marciszack y Gonnet en [23], proponen la construcción de una ontología de requerimientos funcionales, de la cual se concluyen las siguientes ventajas de uso:

i) Posibilita documentar que las características relevantes del sistema como artefactos de tipo plantilla de casos de uso;

ii) Propone una técnica para la generación de un modelo de caso de uso basado en AFD que posibilita su validación.

iii) Utiliza modelos de pruebas basados en AFD, lo que permite derivar un conjunto de casos de pruebas, especificando cierto criterio de cobertura. Esta técnica además, posibilita detectar fallas o defectos en requerimientos;

iv) Propicia la interoperabilidad entre herramientas de soporte HECU y HGCP, ya que se basa en un modelo conceptual que soporta los metamodelos de ellas. [23]

Los autores Helming, Koegel y Naughton en [24] definen: “modelo de proyectos” a los artefactos planificación, gestión y estructura organizativa, y “modelo de sistema” a los artefactos de requerimientos, diseño e implementación.

Destacan que “existen muchas relaciones de gran importancia que se dan entre la gestión de proyectos de software y el modelo del sistema, que tienen una influencia directa en el proyecto.” [24]. Se analizan los siguientes conceptos claves:

(1) Capacidad para navegar entre tareas y la parte correspondiente de la especificación: Refiere a una trazabilidad bidireccional, desde el punto de vista del modelo del sistema hasta el modelo del proyecto.

(2) Agregación del estado de gestión de proyectos de software por artefactos de especificación del sistema: Util para modelos de ciclo de vida donde se utiliza planificación, por ejemplo SCRUM.

(3) Uso de entidades del modelo del sistema para la planificación de proyectos: La integración entre los artefactos de modelos de sistema y de proyecto proporciona una mejor planificación.

(4) Validación del modelo unificado: Aplicando técnicas de validación usadas en el modelo de sistema por medio de criterios de consistencia o integridad al modelo unificado.

Por medio del abordaje del estudio con Unicase, siendo una herramienta CASE (Ingeniería Asistida por Computadora) diseñada para un modelo unificado se expone la posibilidad de gestionar la integración de dicho modelo unificado.

La tabla 3 presenta un resumen de la propuesta de cada artículo.

Tabla 3: Resumen de la propuesta de cada artículo.

Referencia	Propuesta
[12]	Proceso basado en el Manifiesto de Desarrollo Ágil
[13]	Mecanismo de generación y versionado de trazas
[14]	Herramienta web para la gestión de requerimientos
[15]	Importancia de la trazabilidad en el desarrollo ágil
[16]	Trazabilidad de requerimientos antes y después de especificación
[17]	Trazabilidad en métodos ágiles
[18]	Impacto de estrategias de trazabilidad en calidad de software
[19]	Nuevo método de trazabilidad Event-Based Traceability (EBT)
[20]	Personalización de trazabilidad
[21]	Gestión de cambios con trazabilidad
[22]	Matriz de Trazabilidad de Requerimientos (RTM)
[23]	Ontología de requerimientos funcionales
[24]	Modelo unificado de proyectos y sistemas

6. Discusión

De la totalidad de los 20 artículos analizados se descartaron 7 debido a tener un abordaje específico a código o la presentación de software particulares asegurando que los mismos cumplan con correctitud la administración pero sin hacer foco en la trazabilidad, desviándose de las preguntas de investigación planteadas.

La mayoría de los artículos tiene un abordaje fuertemente enfocado en el eje de las problemáticas que encuentran las diferentes organizaciones al integrar y aplicar la trazabilidad, proponiendo distintas alternativas de mejora. Se dispone entonces de las siguientes categorías en común y distribuciones en la tabla 4.

Dicho relevamiento refleja los puntos en común abordados por una variedad de planteos de diferentes autores. Todos los artículos coinciden en que la trazabilidad es fundamental para la gestión de software. Siendo el mismo, un campo donde existe gran diversidad de propuestas para su aplicación e integración.

Tabla 4: Distribución de categorías.

Categoría	Cantidad de artículos
Propuestas de mejora a los problemas de aplicación de trazabilidad	6
Trazabilidad integrada a metodologías ágiles	3
Herramientas y técnicas de aplicación	4

De dichos artículos se concluyen las siguientes respuestas a las preguntas de investigación:.

PI1: En [12], [15], [17] se discute la integración de la trazabilidad en metodologías ágiles y la personalización de la trazabilidad según las necesidades del proyecto. Se mantiene la trazabilidad a partir de documentos claves como el de “Iniciación”, “Definición preliminar” y “Revisión” o la matriz de trazabilidad en el levantamiento de requerimientos, útil para equipos ágiles.

PI2: En [17], [24] y [23] se proponen mantener la trazabilidad sin necesidad de tener documentación exhaustiva, alineándose con los principios ágiles de documentar solo lo necesario y priorizar el software en funcionamiento. Se puede mantener la trazabilidad usando herramientas automatizadas, integración continua, documentación mínima viable, mantenimientos de un backlog, reuniones regulares, etc

PI3: En [12] se menciona que la trazabilidad puede mantenerse en equipos distribuidos geográficamente mediante el uso de herramientas de colaboración y gestión de proyectos, así como políticas claras sobre información de rastreo.

PI4: Los art. [13], [14], [16], [18], [19], [20] y [22] mencionan diversas herramientas y procesos que pueden facilitar la implementación de la trazabilidad, incluyendo herramientas, técnicas y mecanismos automáticos de generación de trazas que son esenciales para gestionar la trazabilidad de manera eficiente.

PI5: En [12] y [15] las prácticas ágiles como reuniones diarias, revisiones de sprint y retrospectivas se utilizan como métodos efectivos para reforzar la trazabilidad, todos los miembros del equipo deben estar al tanto de los cambios en los requerimientos.

PI6: En [15] y [21] se resalta como la trazabilidad puede aumentar la eficiencia del equipo, proporcionando una visión clara de los requerimientos y permitiendo la identificación rápida de desviaciones y la toma de medidas correctivas.

7. Conclusiones y trabajo futuro

La trazabilidad de requerimientos es fundamental para contribuir a la calidad y al éxito de los proyectos de software. Los enfoques presentados abarcan desde métodos tradicionales hasta técnicas avanzadas basadas en eventos y recuperación de información, reflejando la importancia de personalizar la trazabilidad según las necesidades específicas del proyecto. Además, integrar la trazabilidad en metodologías ágiles y utilizar herramientas semiautomáticas puede mejorar la gestión de cambios y la documentación de requerimientos. En general, una buena planificación y la aplicación de políticas adecuadas son cruciales para optimizar la trazabilidad y, en consecuencia, la calidad del software.

En este trabajo se presenta un estudio exploratorio y descriptivo sobre la evolución de las prácticas de trazabilidad acompañando las modificaciones de contexto que se inclinan por una fuerte presencia de metodologías ágiles, trabajo en entorno geográficamente distribuido y equipos reducidos.

Como trabajo futuro se propone aplicar las técnicas presentadas a un ejercicio práctico a fin de evaluarlas y poder compararlas.

8. Referencias

1.http://sedici.unlp.edu.ar/bitstream/handle/10915/3956/2_-_ideas_generales_de_cmmi-sw.pdf?sequence=10 Accedido julio de 2024.
2. Sommerville, I. (2002) Ingeniería de software [6ta edición], Addison Wesley, México
3. Nicholas J, Steyn H. (2017) Project Management for Engineering, Business and Technology. 5th ed. London: Routledge.
4. https://www.redhat.com/es/topics/automation/what-is-configuration-management. Accedido julio de 2024.
5. https://www.redhat.com/es/topics/devops/what-is-ci-cd. Accedido julio de 2024.
6. https://www.oracle.com/ar/scm/product-lifecycle-management/requirements-management/ . Accedido julio de 2024.
7. Project Management Institute (2013) Guía de los fundamentos para la dirección de proyectos (Guía PMBOK) Quinta edición “9 - GESTIÓN DE LOS RECURSOS HUMANOS DEL PROYECTO”, pp. 262.
8. https://sedici.unlp.edu.ar/. Accedido julio de 2024.
9. https://www.ieee.org/ . Accedido julio de 2024.
10. https://scholar.google.es/schhp?hl=es . Accedido julio de 2024.
11. https://www.researchgate.net/ . Accedido julio de 2024.
12. Klein P. (2021) Proceso de desarrollo de software con trazabilidad de requerimientos, documentación mínima y aplicable con equipos pequeños y distribuido.
13. Vera A. F., Hadad G. D. S. & Doorn J. H. (2013) Trazabilidad de Versiones en Ingeniería de Requisitos. XV Workshop de Investigadores en Ciencias de la Computación,WICC.
14. Bersano D. M. (2014) Administración y trazabilidad inteligente de requerimientos.
15. Guerra, J. (2021) Una propuesta para facilitar la trazabilidad de los requerimientos en un desarrollo ágil,

16. Gotel O. C. Z. & Finkelstein A. C. W. (1994) An Analysis of the Requirements Traceability Problem.17. Jacobsson M. (2009) Implementing Traceability In Agile Software Development-
18. Rempel P., Mçder P. & T. Kuschke (2013) An empirical study on project-specific traceability strategies. 21st IEEE International Requirements Engineering Conference.
19. Cleland-Huang J., Chang C.K. & Christensen M. (September 2003) Event-based traceability for managing evolutionary change. IEEE Transactions on Software Engineering.
20. Lago P., Muccini H. & Van Vliet H. (Enero 2009) A scoped approach to traceability management. Journal of Systems and Software.
21. Dapozo G. N., Greiner C. L., Irrazábal E. , Medina Y., Ferraro M., Lencina A., Chiapello J. & Mascheroni M. A. (2016)Métodos y herramientas de estimación, gestión cuantitativa de proyectos, trazabilidad de requerimientos y entrega continua, orientados a la mejora de la calidad del software.
22. Kattz V. E., Martínez M. P., Vergara C. J.& Morales Velázquez L. A. (2024) Matriz de trazabilidad en el levantamiento de requisitos: una revisión sistemática de la literatura..
23.Roldán M. L., Vegetti M., Marciszack M. M. & Gonnet S. (2017) Un Modelo Conceptual para la Especificación y Trazabilidad de Requerimientos Funcionales basados en Casos de Uso y Casos de Prueba.
24. Helming J., Koegel M. & Naughton H. (2009) Towards traceability from project management to system models. ICSE Workshop on Traceability in Emerging Forms of Software Engineering.
25. Wang, B., Peng, R., Li, Y., Lai, H., & Wang, Z. (2018). Requirements traceability technologies and technology transfer decision support: A systematic review. *Journal of Systems and Software*, 146, 59-79.
26. Javed, M. A., & Zdun, U. (2014, May). A systematic literature review of traceability approaches between software architecture and source code. In *Proceedings of the 18th International Conference on Evaluation and Assessment in Software Engineering* (pp. 1-10).
27. Mustafa, N., & Labiche, Y. (2017, July). The need for traceability in heterogeneous systems: A systematic literature review. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)* (Vol. 1, pp. 305-310). IEEE.