



Universidad Nacional Mayor de San Marcos

Universidad del Perú. Decana de América

Dirección General de Estudios de Posgrado
Facultad de Ingeniería de Sistemas e Informática
Unidad de Posgrado

**Herramienta gráfica para la automatización de
historias de usuario desde un modelo orientado a
objetivos con gestión de trazabilidad**

TESIS

Para optar el Grado Académico de Magíster en Ingeniería de
Sistemas e Informática con mención en Ingeniería de software

AUTOR

Delicia Esmeralda LUCERO GUEVARA

ASESOR

Dra. Nora Bertha LA SERNA PALOMINO

Lima, Perú

2022



Reconocimiento - No Comercial - Compartir Igual - Sin restricciones adicionales

<https://creativecommons.org/licenses/by-nc-sa/4.0/>

Usted puede distribuir, remezclar, retocar, y crear a partir del documento original de modo no comercial, siempre y cuando se dé crédito al autor del documento y se licencien las nuevas creaciones bajo las mismas condiciones. No se permite aplicar términos legales o medidas tecnológicas que restrinjan legalmente a otros a hacer cualquier cosa que permita esta licencia.

Referencia bibliográfica

Lucero, D. (2022). *Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de trazabilidad*. [Tesis de maestría, Universidad Nacional Mayor de San Marcos, Facultad de Ingeniería de Sistemas e Informática, Unidad de Posgrado]. Repositorio institucional Cybertesis UNMSM.

Metadatos complementarios

Datos de autor	
Nombres y apellidos	Delicia Esmeralda Lucero Guevara
Tipo de documento de identidad	DNI
Número de documento de identidad	40959243
URL de ORCID	
Datos de asesor	
Nombres y apellidos	Nora Bertha La Serna Palomino
Tipo de documento de identidad	DNI
Número de documento de identidad	07665297
URL de ORCID	https://orcid.org/0000-0002-4292-344X
Datos del jurado	
Presidente del jurado	
Nombres y apellidos	Lenis Rossi Wong Portillo
Tipo de documento	DNI
Número de documento de identidad	10438282
Miembro del jurado 1	
Nombres y apellidos	César Luza Montero.
Tipo de documento	DNI
Número de documento de identidad	06111988
Miembro del jurado 2	
Nombres y apellidos	Carlos Ernesto Chávez Herrera
Tipo de documento	DNI
Número de documento de identidad	40788722
Datos de investigación	
Línea de investigación	C.0.3.22. Ingeniería de software

Grupo de investigación	No aplica.
Agencia de financiamiento	Sin financiamiento.
Ubicación geográfica de la investigación	Universidad Nacional Mayor de San Marcos País: Perú Departamento: Lima Provincia: Lima Distrito: Lima Latitud: -12.056445 Longitud: -77.085994
Año o rango de años en que se realizó la investigación	2019- 2021
URL de disciplinas OCDE	Otras ingenierías y tecnologías https://purl.org/pe-repo/ocde/ford#2.11.02



UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
Universidad del Perú. Decana de América
Facultad de Ingeniería de Sistemas e Informática
Vicedecanato de Investigación y Posgrado
Unidad de Posgrado

**ACTA DE SUSTENTACIÓN VIRTUAL DE TESIS PARA OPTAR EL GRADO
ACADÉMICO DE MAGÍSTER EN INGENIERÍA DE SISTEMAS E
INFORMÁTICA CON MENCIÓN EN INGENIERÍA DE SOFTWARE**

A los dieciocho (18) días del mes de enero de 2022, siendo las 5:00 pm., se reunieron en la sala virtual <https://meet.google.com/mic-gdop-zze> el Jurado de Tesis conformado por los siguientes docentes:

Dra. Lenis Rossi Wong Portillo (Presidente)
Mg. César Luza Montero (Miembro)
Mg. Carlos Ernesto Chávez Herrera (Miembro)
Dra. Nora Bertha La Serna Palomino (Miembro Asesor)

Se inició la Sustentación invitando a la candidata a Magíster **DELICIA ESMERALDA LUCERO GUEVARA**, para que realice la exposición oral y virtual de la tesis para optar el Grado Académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software, siendo la Tesis intitulada:

**"HERRAMIENTA GRÁFICA PARA LA AUTOMATIZACIÓN DE HISTORIAS DE
USUARIO DESDE UN MODELO ORIENTADO A OBJETIVOS CON GESTIÓN DE
TRAZABILIDAD"**

Concluida la exposición, los miembros del Jurado de Tesis procedieron a formular sus preguntas que fueron absueltas por la graduanda; acto seguido se procedió a la evaluación correspondiente, habiendo obtenido la siguiente calificación:

18 - Dieciocho - (Muy bueno)

Por tanto, el presidente del Jurado, de acuerdo al Reglamento General de Estudios de Posgrado, otorga a la Bachiller **DELICIA ESMERALDA LUCERO GUEVARA** el Grado Académico de Magíster en Ingeniería de Sistemas e Informática con mención en Ingeniería de Software.

Siendo las. 18:20 horas, el presidente del Jurado de Tesis, da por concluido el acto académico de Sustentación de Tesis.

Dra. Lenis Rossi Wong Portillo
(Presidente)

Mg. César Luza Montero
(Miembro)

Mg. Carlos Ernesto Chávez Herrera
(Miembro)

Dra. Nora Bertha La Serna Palomino
(Miembro Asesor)

DEDICATORIA

A Dios por permitir que mis padres vean los frutos de su cosecha, ellos son mi mejor ejemplo de perseverancia y excelencia, sus oraciones se convierten en bendiciones (personas) que me enseñan, ayudan y motivan. Aunque están lejos siempre los siento cerquita de mí.

ÍNDICE

ÍNDICE DE TABLAS.....	vii
ÍNDICE DE FIGURAS.....	ix
RESUMEN.....	xii
ABSTRACT	xiv
CAPÍTULO 1: INTRODUCCIÓN.....	1
1.1 Situación Problemática.....	1
1.2 Formulación del problema.	3
1.2.1 Problema General	4
1.2.2 Problemas Específicos.....	4
1.3 Justificación Teórica	4
1.4 Justificación Práctica	5
1.5 Objetivos de la Investigación	6
1.5.1 Objetivo general	6
1.5.2 Objetivos específicos.....	7
CAPÍTULO 2: MARCO TEÓRICO.....	8
2.1 Marco Filosófico o epistemológico de la investigación	8
2.1.1 Epistemología del Modelado.....	8
2.1.2 Epistemología de la Trazabilidad	10
2.2 Antecedentes de la investigación.....	11
2.2.1 Trabajos Relacionados	11
2.2.2 Métodos, modelos, técnicas.....	52
2.2.3 Software existente	56
2.3 Bases Teóricas.....	59
2.3.1 Ingeniería de Requisitos	59
2.3.2 Framework i*.....	62
2.3.3 Historia de Usuario.....	63
2.3.4 Trazabilidad.....	65

2.4	Hipótesis general	66
2.5	Hipótesis específicas	67
2.6	Identificación de variables.....	67
2.7	Operacionalización de variables.....	67
2.8	Matriz de consistencia	68
2.9	Resumen del Capítulo	71
	CAPÍTULO 3: METODOLOGÍA.....	72
3.1	Tipo y Diseño de Investigación	72
3.2	Unidad de análisis	72
3.3	Población de estudio.....	73
3.4	Tamaño de muestra	73
3.5	Técnicas de recolección de Datos.....	73
3.6	Técnica de análisis de los resultados	74
3.7	Resumen del Capítulo	74
	CAPÍTULO 4: APORTE TEÓRICO.....	75
4.1	Comparación de las tecnologías existentes	75
4.1.1	Comparación de Librerías gráficas.....	80
4.2	Definición de la solución	83
4.2.1	Modelado de Requisitos bajo el enfoque Orientado a Objetivos	86
4.2.2	Generación de Historias de Usuario	90
4.2.3	Gestión de la Trazabilidad	91
4.2.4	Correspondencia conceptual para solución de la Herramienta Gráfica MOOTRACE.	92
4.3	Metodología para el desarrollo de la herramienta MOOTRACE .	94
4.3.1	Fase de Planeación	95
4.3.2	Historias de Usuario.	96
4.3.3	Plan de Iteración	97
4.3.4	Fase de Diseño	98

4.3.5	Fase de Codificación	98
4.3.6	Fase de Pruebas	99
4.3.7	Resultados de las Pruebas.....	102
4.4	Resumen del capítulo	103
	CAPÍTULO 5: DESARROLLO DE LA HERRAMIENTA	104
5.1	Tecnología del software	104
5.2	Codificación de las Historias de Usuario	105
5.2.1	El Modelo	106
5.2.2	El Controlador	108
5.2.3	La Vista.....	109
5.3	Resumen del capítulo	113
	CAPÍTULO 6: RESULTADOS Y DISCUSIÓN	115
6.1	Análisis, interpretación y discusión de resultados	115
6.1.1	Definición del Objetivo del Experimento	116
6.1.2	Diseño del Experimento	116
6.1.3	Ejecución	119
6.2	Pruebas de Hipótesis	137
6.2.1	Prueba de Hipótesis del Objetivo Específico 1	137
6.2.2	Prueba de Hipótesis del Objetivo Específico 2 Métrica 1	138
6.2.3	Prueba de Hipótesis del Objetivo Específico 2 Métrica 2	139
6.3	Presentación de resultados	140
	CONCLUSIONES.....	141
	RECOMENDACIONES.....	142
	REFERENCIAS BIBLIOGRÁFICAS.....	143
	ANEXOS	153
	Anexo 1: Siglario	153
	Anexo 2: Historias de Usuario de la construcción de la Herramienta	
	MOOTRACE.....	154

Anexo 3: Especificaciones de prueba	156
--	------------

ÍNDICE DE TABLAS

Tabla 1 Factores sociales y humanos agrupados por categorías.	20
Tabla 2 Desafíos de la ingeniería de requisitos en metodologías ágiles.	23
Tabla 3 Aplicaciones Web más recientes y sus características.	25
Tabla 4 Cuadro Comparativo de los trabajos relacionados.	43
Tabla 5 Cuadro resumen de las Etapas del proceso KLEIN.	45
Tabla 6 Cuadro de Relación entre la Gestión del Alcance y Grupo de Procesos.	46
Tabla 7 Cuadro que muestra la Exactitud de los valores de las métricas.	50
Tabla 8 Trabajos realizados para representar modelos conceptuales desde documentos del negocio	53
Tabla 9 Reglas de mapeo Basadas en la propiedad POS en el XML entre Elemento en Modelo, Elemento en Texto	55
Tabla 10 Matriz de Consistencia	69
Tabla 11 Cuadro Comparativo de Tecnologías existentes relacionadas a la investigación.	77
Tabla 12 Cuadro Comparativo de Enfoques de Objetivos para modelado.	78
Tabla 13 Librerías para Editores Gráficos.	81
Tabla 14 Historia de Usuario 1.	97
Tabla 15 Plan de Iteración	97
Tabla 16 Especificación de Prueba: Crear Editor Gráfico (Historia 1)	100
Tabla 17 Tiempo de transformación de requisitos a funcionalidades del Proyecto de Trámites del Ciudadano.	123
Tabla 18 Tiempo de transformación de requisitos a funcionalidades del Proyecto de Bienes Patrimoniales.	124
Tabla 19 Tiempo de transformación de requisitos a funcionalidades del Proyecto Planificador de Sala de Reuniones.	125
Tabla 20 Tiempo de transformación de requisitos a funcionalidades del Proyecto de Calificación de Desempeño Profesional	126
Tabla 21 Tiempo de transformación de requisitos a funcionalidades del Proyecto Gestión Archivística.	127

Tabla 22 <i>Tiempo de transformación de requisitos a funcionalidades del Proyecto Emisión Tarjeta Entidad Laboral.</i>	128
Tabla 23 <i>Tiempo de transformación de requisitos a funcionalidades del Proyecto de Control de Citas.</i>	129
Tabla 24 <i>Tiempo de transformación de requisitos a funcionalidad del Proyecto de Índice de Masa Corporal.</i>	130
Tabla 25 <i>Métricas de Calidad del requisito.</i>	132
Tabla 26 <i>Datos de cálculo de la M1 en los proyectos del experimento.</i>	133
Tabla 27 <i>Datos de cálculo de la M2 en los proyectos del experimento.</i>	135
Tabla 28 <i>Historia de Usuario 2.</i>	154
Tabla 29 <i>Historia de Usuario 3.</i>	154
Tabla 30 <i>Historia de Usuario 4.</i>	155
Tabla 31 <i>Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico (Historia 2).</i>	156
Tabla 32 <i>Especificación de Prueba: Automatización de las relaciones de los objetos del modelo SR para generar automáticamente las Historias de Usuario establecidas en la correspondencia de la Figura 31 (Historia 3)</i>	157
Tabla 33 <i>Personalización de Historia de Usuario (Historia 4)</i>	159

ÍNDICE DE FIGURAS

Figura 1 <i>Método de trazabilidad basado en el comportamiento</i>	12
Figura 2 <i>Prueba de Aceptación del ejemplo de prueba EventCompany</i>	13
Figura 3 <i>Fragmento de Salida para la Historia de Usuario de prueba EventCompany</i>	13
Figura 4 <i>Modelo conceptual del marco de trabajo Merlenn</i>	14
Figura 5 <i>Modelo conceptual del marco de trabajo Merlenn</i>	16
Figura 6 <i>Rutas de transformación del conocimiento tácito y explícito en la dinámica organizacional</i>	18
Figura 7 <i>Propuesta de Modelo Web Adaptativo para soportar todas las características</i> 26	
Figura 8 <i>Configuración de los métodos que componen el nuevo modelo adaptativo</i>	27
Figura 9 <i>Configuración de los métodos que componen el nuevo modelo adaptativo</i>	29
Figura 10 <i>Configuración de los métodos que componen el nuevo modelo adaptativo</i>	30
Figura 11 <i>Componentes del Framework para obtener y especificar los Requisitos de Interoperabilidad</i>	34
Figura 12 <i>Requisito de Interoperabilidad y su configuración dentro de un pool</i>	35
Figura 13 <i>Flujos de mensajes y sus respectivas configuraciones para su envío y recepción.</i>	36
Figura 14 <i>Primer Subproceso de establecer el alcance de la elicitación de un RI</i>	37
Figura 15 <i>Segundo Subproceso para capturar atributos de un RI</i>	37
Figura 16 <i>Tercer Subproceso para identificar condiciones para utilizar los datos.</i>	38
Figura 17 <i>Guía propuesta para el registro de atributos de la vista de emisores, receptores y Tipos de interacción.</i>	39
Figura 18 <i>Guía propuesta para el registro de características asociadas a la vista de propiedades del flujo de datos.</i>	40
Figura 19 <i>Etapas del proceso de Desarrollo KELIN.</i>	44
Figura 20 <i>Guía propuesta para el registro de atributos asociados a la vista de propiedades del flujo de datos.</i>	47
Figura 21 <i>Algoritmo basado en aprendizaje automático para la selección de técnicas de elicitación.</i>	49

Figura 22 <i>Resultados de la puntuación obtenida en la ejecución del Modelo de Machine Learning</i>	51
Figura 23 <i>Formularios de preguntas-respuestas previo a la generación del Diagrama de Casos de Uso y el Diagrama de Clases.</i>	57
Figura 24 <i>Diagrama de Casos de Uso y el Diagrama de Clases generados</i>	58
Figura 25 <i>Transformación PIM a PSM</i>	58
Figura 26 <i>Transformación PIM a Código</i>	59
Figura 27 <i>Impacto en la fase de Mantenimiento ocasionado por la reparación de un error no identificado tempranamente en la Elicitación de Requisitos.</i>	61
Figura 28 <i>Ranking de las mejores tecnologías para desarrollo de software.</i>	80
Figura 29 <i>Propiedades y métodos de la librería GoJS.</i>	83
Figura 30 <i>Esquema conceptual de la Herramienta Gráfica</i>	85
Figura 31 <i>Notación de Framework i*</i>	87
Figura 32 <i>Strategic Dependency.</i>	88
Figura 33 <i>Strategic Relational.</i>	88
Figura 34 <i>Correspondencia entre el Modelo Strategic Rationale (SR) y la Historia de Usuario</i>	93
Figura 35 <i>Metodología Xtreme Programming.</i>	94
Figura 36 <i>Arquitectura Conceptual MOOTRACE.</i>	104
Figura 37 <i>Arquitectura Tecnológica de la herramienta gráfica.</i>	105
Figura 38 <i>Modelo de la Base de Datos de MOOTRACE.</i>	106
Figura 39 <i>Arquitectura Conceptual</i>	107
Figura 40 <i>Modelo Conceptual</i>	108
Figura 43 <i>Controladores MOOTRACE</i>	108
Figura 42 <i>Referencia a librería gráfica Go.js</i>	110
Figura 43 <i>Creación de la Caja de Herramientas</i>	111
Figura 44 <i>Referencia a librería gráfica Go.js</i>	111
Figura 45 <i>Resultado en la vista final.</i>	111
Figura 46 <i>Vistas MOOTRACE</i>	112
Figura 47 <i>Creación del Modelo Gráfico</i>	112
Figura 48 <i>Automatización de las Historias de Usuario</i>	113

Figura 49 <i>Proceso del Diseño Experimental en Software</i>	115
Figura 50 <i>Elementos de un experimento</i>	117
Figura 51 <i>Proceso de Desarrollo de Requisitos en la ingeniería de requisitos</i>	120
Figura 52 <i>Tiempo del proceso de ingeniería de requisitos por fase en cada sistema</i>	131
Figura 53 <i>Métrica 1: Grado de Especificidad de los Requisitos</i>	134
Figura 54 <i>Métrica 2: Grado de Validez de los Requisitos</i>	136
Figura 55 <i>Gráfico del Porcentaje de eficiencia en la ejecución del proceso de ingeniería de requisitos en el Grupo de Control y Experimental</i>	137
Figura 56 <i>Gráfico del Porcentaje de eficiencia en la ejecución de la Métrica 1 del Objetivo Específico 2</i>	138
Figura 57 <i>Gráfico del Porcentaje de eficiencia en la ejecución de la Métrica 2 del Objetivo Específico 2</i>	139

RESUMEN

La fase de elicitación de requisitos se apoya de métodos y técnicas para alcanzar un nivel de calidad que refleja la satisfacción del usuario, actualmente siendo las metodologías ágiles una alternativa no sólo para la fase del desarrollo, sino que además en su fase previa hace uso de la historia de usuario para identificar los requisitos; este es un artefacto exclusivamente técnico cuyo uso y entendimiento corresponde al equipo de proyecto. Al respecto, esta investigación implementa el modelado orientado a objetivos en una plataforma para el modelado gráfico de los requisitos identificados el cual utiliza la notación i* con estereotipos sencillos y entendibles al usuario que posteriormente automatiza los requisitos en el formato de una historia de usuario.

El enfoque de la Metodología Orientada a Objetivos tiene como fundamento la variabilidad con la que se puede concretar un objetivo, pero que este último en sí, no cambia. En relación con la variabilidad, se propone correlacionarla con los fundamentos de la metodología ágil, de manera automática a través de las trazas e información generada desde el modelo gráfico elaborado con la participación del usuario en la que expresa sus necesidades y, se puede interpretar en el lenguaje natural de la Metodología Orientada a Objetivos.

Respecto al procedimiento anterior y, la información generada se constituye en el insumo para a partir de ahí generar automáticamente las historias de usuario que al llegar a manos del diseñador o programador no existan ambigüedades en ninguna de las plataformas: la gráfica expresada en el modelo gráfico del MOO y la narrativa representada por la Historia de Usuario.

Los casos de estudio en la presente investigación demostraron que la herramienta propuesta cumple con los principios de las metodologías ágiles que destacan la participación del usuario en esta etapa para definir de manera precisa los requisitos por parte del equipo de proyecto, en ese sentido, se garantiza que sean completos y consistentes, optimizando la comunicación y el tiempo empleado por los miembros del equipo para elaborar el artefacto a su cargo y de los usuarios finales aportándonos

su conocimiento en la fase de captura de requisitos, promoviendo de esta manera la importancia y el impacto realizar una adecuada elicitación desde etapas tempranas.

Palabras Clave: requisitos, historia de usuario, metodologías ágiles, modelamiento orientado a objetivos

ABSTRACT

The requirements elicitation phase is supported by methods and techniques to achieve a level of quality that reflects user satisfaction, currently being agile methodologies an alternative not only for the development phase, but also in its previous phase makes use of the user story to identify user requirements. However, it is not an artifact whose use and understanding by the project team is exclusively technical. In this regard, this research implements objective-oriented modeling in a platform for graphic modeling of the identified requirements, which uses the i* notation with simple and understandable stereotypes for the user, which subsequently automates the requirements in the format of a user story.

The Objective-Oriented Methodology approach is based on the variability with which an objective can be specified, but that the latter in itself does not change. In relation to variability, it is proposed to correlate it with the fundamentals of the agile methodology, automatically through the traces and information generated from the graphic model prepared with the active participation of the user in which they express their needs and, it can be interpreted in the natural language of the MOO.

Regarding the previous procedure and becoming the input to from there automatically generate user stories that when reaching the hands of the designer or programmer there are no ambiguities in any of the platforms: the graph expressed in the graphic model of the MOO and the narrative represented by the User Story.

The case studies in this research showed that the proposed tool complies with the principles of agile methodologies that highlight the participation of the user in this stage to precisely define the requirements by the project team, in that sense, it is guaranteed that are complete and consistent, optimizing communication and the time spent by team members to develop the artifact in their charge and by end users, providing us with their knowledge in the requirements capture phase, thus promoting the importance and impact perform adequate elicitation from early stages.

Keywords: requirements, user history, agile methodologies, objective-oriented modeling

1 CAPÍTULO: INTRODUCCIÓN

1.1 Situación Problemática

La ingeniería de software como disciplina estructura principios verificables para la construcción de aplicaciones útiles y viables para el usuario (Bauer, 1971; Bourque y Fairley, 2014; Serna, 2012). En el desarrollo de un sistema, el equipo de proyecto con el fin de asegurar la fiabilidad del software debe elaborar cada uno de los componentes desde la fase de análisis, sin embargo, Serna y Serna (2020) concluyeron que las razones de la no formalización de los requisitos son: “falta de formación académica, falta de aceptación en la industria, realmente no hay demostraciones importante de sus ventajas, los costos, la falta de capacitación personal, los ingenieros tienen fobia a las matemáticas”.

El tiempo empleado para interpretar las necesidades del usuario y transferirlas a capacidades que debe cumplir un sistema, representa un esfuerzo que ha sido valorado conforme el campo de la ingeniería del software ha ido evolucionando y estandarizando las fases que un sistema debe cumplir, a través del uso de buenas prácticas, prácticas efectivas y, uso de herramientas según el contexto de su desarrollo, las cuales permitan asegurar el éxito del producto (Sommerville, 2011), sin embargo, el desarrollo de herramientas como apoyo para la elicitación de requisitos no representa un área de mucho interés para investigadores (Carrizo, Dante, y Rojas, Jorge, 2018).

En los últimos años, el involucramiento y empoderamiento del cliente dentro del proyecto ha significado un aporte importante para la gestión de proyectos de desarrollo de software; en ese sentido, en el Manifiesto for Agile Software Development indica “Los procesos Ágiles promueven el desarrollo sostenible. Los promotores, desarrolladores y usuarios debemos ser capaces de mantener un ritmo constante de forma indefinida”.

Consecuentemente, las relaciones no especificadas entre los artefactos implicados en un ciclo de vida software, especialmente entre productos que tienen una relación predecesor-sucesor o maestro-subordinado nos coloca en un escenario poco entendible en donde existen distintas definiciones para un mismo proceso (Jiménez, 2012; Ramírez, Giraldo y

Anaya, 2017). Aunque estos activos son para vistas distintas (usuarios o miembros del equipo de proyecto), aparentemente justificaría la poca legibilidad o información disponible; pero hay que tener en cuenta que si estos cumplen un mismo propósito esta información debería permitirnos identificar el origen del requerimiento con el objetivo de alcanzar la satisfacción del usuario y el posterior mantenimiento de la aplicación desarrollada, sin que se desnaturalice su principal objetivo.

Asimismo, hay enfoques que soportan la reutilización (Sommerville, 2011), en el diseño de los componentes y documentación del desarrollo de software, con la finalidad de disminuir los tiempos de entrega del producto de software y por consiguiente incrementar el nivel de satisfacción del usuario; sin embargo, aun cuando se hayan automatizado los procesos del negocio mediante herramientas de modelado (Jiménez, 2010; Loucopoulos y Karakostas, 1995; StarUML, 2020) que ya son útiles al usuario dentro de una organización su poca fiabilidad o entendimiento del acervo tecnológico existente, resulta “más sencillo” volver a crear estos artefactos cuando se va a desarrollar un nuevo producto de software.

Actualmente, el desarrollo de las aplicaciones web, se realiza haciendo uso de metodologías que permiten el desarrollo individual o adquisición a través de terceros generando sus propios activos. En consecuencia, la diversidad de soluciones que funcionan en un negocio se compone de distintos artefactos dependiendo de la metodología y/o tecnología con la que se hayan desarrollado, una mejora u actualización en el proceso de negocio ocasiona que los administradores tengan que actualizar cada uno de los componentes o documentación existente.

Los artefactos intermedios o finales, por ejemplo: Realización de caso de uso, Análisis de clase, Modelo de Clase, Diagramas de Secuencia, Modelo de Despliegue, Modelo de Análisis, Modelo de Diseño y Documentación de Arquitectura entre otros; que cubren la cadena de producción de software bajo un enfoque tradicional, los cuales son operacionalmente útiles para el especialista que lo elabora, sin embargo, su interpretación y comunicación es sesgada para los otros. Por otro lado, existen empresas que sustentan la adopción de un enfoque de reutilización por el impacto en su nivel de productividad de software, Urroz, (2012), otra propuesta, sugiere recoger a través del tiempo, información

referida a cambios o mejoras, que permitan estimar la pertinencia de continuar dando mantenimiento al producto, u optar por uno nuevo (Farfán y Vega-Zepeda, 2020).

En ese sentido, tenemos la documentación para sustentar cada fase de la elaboración del producto de software, cada uno con sus estereotipos o formato estándar; sin embargo, se sabe que los cambios o mejoras son inminentes durante el desarrollo de una aplicación y se realizarán dentro de su ecosistema de manera independiente, al respecto no tendremos los elementos adecuados para identificar el origen de cada uno de los requisitos o ubicarlo en alguna de las diferentes etapas del proceso de producción y que al intentar realizar esta tarea terminará impactando en los tiempos asociados a la ejecución del cronograma.

Un proceso de elicitación ineficaz, demanda recursos y actividades que no ofrecen valor al sistema en desarrollo, por lo que se infiere que las funcionalidades no estén acorde a lo requerido por el usuario, como consecuencia podrían manifestar su insatisfacción, porque asumen que no toman en cuenta sus necesidades o la interpretación de lo que expresaron está alejada de la realidad, en consecuencia, estaríamos frente a un desarrollo caótico porque la información no es fidedigna, se toman decisiones incorrectas, los costos y el tiempo exceden lo proyectado en el cronograma.

1.2 Formulación del problema.

Las metodologías de desarrollo de software existentes, concentran sus técnicas y métodos en etapa del desarrollo de la aplicación específicamente y, no se profundiza en el empleo de modelos que ofrezcan la alternativa del reúso de los activos del software generados en la etapa de elicitación; por ejemplo, para reducir los costos de ingeniería relacionados a los defectos del software, la entrega a tiempo y principalmente la dificultad de los usuarios para describir y transmitir las tareas propias a su cargo o roles, quienes pueden dejar la información importante sin especificar o con poca claridad y en algunos casos poco dispuestos a colaborar con el analista; posteriormente, el desarrollador omite cosas que no considera relevantes o no entiende lo interpretado en el proceso anterior y, así es como un requisito finalmente pierde su esencia.

Al respecto, el uso de lenguajes muy técnicos u operativos en la comunicación entre el usuario y el equipo de desarrollo, aumenta el riesgo que la transferencia de la información vaya cambiando según el entendimiento o léxico de cada miembro aun tratándose del mismo proceso de negocio, que los administradores cuenten con información diferente para definir un mismo proceso o sistemas con funcionalidades similares para resolver una necesidad lo que genera confusión, ambigüedades y duplicidad de esfuerzo.

En este contexto la trazabilidad se pierde entre los elementos y, en ocasiones un artefacto termina siendo reemplazo del que le antecede, al punto que la legibilidad de los requisitos de los usuarios es imposible de transferir a un lenguaje o vista que sea útil para otro colaborador o cliente del proyecto, perdiéndose en el transcurso información relevante que ayude a mejorar el nivel de satisfacción del cliente.

1.2.1 Problema General

¿Qué mejoras se obtienen en el proceso de elicitación de requisitos mediante la automatización de historias de usuarios con una herramienta gráfica desde un modelo orientado a objetivos con gestión de la trazabilidad?

1.2.2 Problemas Específicos

- ¿Cómo se reduce el tiempo de transformación de requisitos a funcionalidades mediante la automatización de historias de usuarios con la herramienta gráfica desde un modelo orientado a objetivos con gestión de la trazabilidad?
- ¿Cómo se incrementa la calidad del requisito con la aplicación de la automatización de historias de usuarios con la herramienta gráfica desde un modelo orientado a objetivos con gestión de la trazabilidad?

1.3 Justificación Teórica

La investigación propuesta, busca mediante los conceptos de agilidad y reutilización reafirmar la importancia de la transferencia iterativa e incremental del conocimiento

utilizado en el desarrollo de una aplicación (Durán, et al., 2013; Verma, Kamboj y Kaur, 2017) y el involucramiento del usuario (Donoso y Vega, 2017). Para ello, el equipo de proyecto debe tener capacidad de respuesta frente a las necesidades del cliente las cuales son tan dinámicas como la evolución de la tecnología en estos tiempos.

En las organizaciones existen componentes u activos que son desarrollados para la solución de una casuística, que bien podrían ser aplicados en otras, esta reutilización de modelos y conocimiento de los usuarios que ya se encuentran familiarizados a los entornos compensa los tiempos en costo y esfuerzo (Nieves-Guerrero. Ucán-Pech. Menéndez-Domínguez, 2014; Verma et al., 2017).

Esta investigación demostrará, que el uso de modelos en el proceso de desarrollo de aplicaciones permite generar una ventaja competitiva y, sobre todo, permiten su crecimiento sostenible en el tiempo convirtiéndose en activos para la reutilización y reducción de tiempos en la producción de software (Durán et al., 2013 y, Fiallos, Suarez y Villavicencio, 2016)

La utilidad de la reutilización queda en evidencia cuando se vuelven a plantear necesidades o escenarios que ya fueron abordados tanto por el equipo de proyecto como por el cliente, puesto que su implementación incurre en tiempos y costos adicionales, en tanto el margen de resistencia al cambio incrementa (Durán et al., 2013; Verma et al., 2017).

1.4 Justificación Práctica

Esta investigación se realiza porque existe la necesidad de mejorar el proceso de elicitación de requisitos para reducir las ambigüedades e inconsistencias.

Las principales contribuciones que ofrecerá la herramienta desarrollada y denominada **MOOTRACE** son:

- Facilitará la generación del modelado de requisitos, con un lenguaje más natural e intuitivo para los colaboradores a través de un modelo gráfico

(Loucopoulos y Karakostas, 1995; Gonçalves, de Oliveira, Monteiro, Castro y Araújo, 2019).

- Validación dinámica de información del modelado, para solucionar las inconsistencias visualizadas en el modelo gráfico y tener una visión integrada.
- Incorporación de mecanismos de automatización (Model Driven Architecture, 2000), que a partir del modelo generen las historias de usuario, esta incorporación permitiría hacer más productivas las entrevistas con los usuarios y afinar de manera más efectiva sus necesidades (Escobar, Velandia, Ordoñez y Cobos, 2015; Jiménez et al., 2014).
- Mejora de la calidad durante la producción del software, identificando tempranamente los aspectos comunes y variables en el modelo generado en el punto anterior, fomentando la reutilización y el diseño para la reutilización Verma et al. (2017).

Este entorno permitirá que la información se solidifique y sea lo más real en medida el equipo de trabajo ágil (multidisciplinar) del proyecto, en este sentido las Historias de Usuario como el artefacto bandera de las metodologías ágiles (Qasaimah, 2011) y que, siendo automatizadas a partir del modelo de requisitos orientado a objetivos, reducirán los errores de transferencias manuales, simplificando las comunicaciones y haciéndolas más productivas y asertivas.

La reutilización de la información de un artefacto y el esfuerzo para generar otro dentro sobre la base de este en un proyecto, o el empleo de estos componentes en otros con alcance semejante, compensa la inversión inicial tanto de costo y tiempo, alcanzando rentabilidad a largo plazo. Además, se sabe que la automatización es una de las formas más sofisticadas para hacer más competitivas a las empresas (Durán, Troya y Vallecillo, 2013; Grigera y Nava, 2021).

1.5 Objetivos de la Investigación

1.5.1 Objetivo general

Desarrollar la Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad en la fase de elicitación de requisitos en el proceso de desarrollo de software para reducir las ambigüedades e inconsistencias, y en consecuencia incrementar la calidad del software.

1.5.2 Objetivos específicos

1. Demostrar que la Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad reduce el tiempo de transformación de requisitos en funcionalidades del sistema.
2. Determinar que la Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad incrementa la calidad de requisitos.

2 CAPÍTULO: MARCO TEÓRICO

Dado que este trabajo de investigación busca desarrollar una herramienta gráfica para la generación automática de Historias de usuario desde un modelo gráfico orientado a objetivos basado en el framework i* con gestión de la trazabilidad, es preciso definir el marco epistemológico en referencia al modelado de requisitos y de la trazabilidad, antecedentes similares a la investigación y, además aclarar conceptos como: Requisitos destacando su impacto en el desarrollo de un proyecto, el alcance del framework i*, la Historia de Usuario en el ámbito de la especificación requisitos y la Gestión de la Trazabilidad como requisito para garantizar la calidad del producto de software.

2.1 Marco Filosófico o epistemológico de la investigación

La epistemología como análisis del conocimiento científico, la estructura lógica de sus teorías, los métodos empleados en la investigación y en la explicación o interpretación de sus resultados y la aseveración y/o falsabilidad de sus teorías (G. Briones, 1996; Rendón, 2021), da cuenta de su causalidad, índole y validez.

Al respecto, mientras la opinión es un punto de vista subjetivo, sin rigor ni fundamento, por el contrario, el conocimiento tiene fundamentos y justificación que, luego de un proceso estricto se ha logrado comprobar y validar apoyado de un método científico para obtener el conocimiento de la verdad, en el ámbito de determinada disciplina científica.

2.1.1 Epistemología del Modelado

El ingeniero de requisitos, en la inherente complejidad de interpretación de las necesidades de los usuarios debe emplear estrategias que permitan identificarlas efectivamente, para su posterior exitosa materialización, es decir, entregar una especificación de requisitos de software correcta y completa, para lograr un producto de software sea de calidad; en donde el margen de ambigüedad e inconsistencia en la definición de los requisitos sean nulos.

La vulnerabilidad del lenguaje textual frente al tecnicismo del equipo de desarrollo, en donde el primero utiliza un lenguaje básico y natural que crea una barrera entendimiento con el segundo luego de realizar la transferencia de necesidad a requerimiento. Al respecto, (Loucopoulos y Karakostas, 1995; Verma et al., 2017), precisan que el uso de un lenguaje natural deja posibilidad de caer en ambigüedades o requisitos erróneos y a fin de mitigar estos riesgos (Leite, Rossi, Balaguer, Majorana, Kaplan, Hadad y Oliveros, 1997; Verma et al., 2017) proponen como mejora utilizar el Léxico Extendido del Lenguaje (LEL), esta técnica tiene como propósito describir los símbolos específicos de comunicación de un grupo social (palabras, frases propias de una cultura) a través, de escenarios plasmando posibles comportamientos del sistema donde el usuario está familiarizado y su nivel de comprensión es mayor; en consecuencia, cualquier validación u observación será entendible y de mucho aporte a la mejora del sistema.

Un factor crítico en el análisis de requisitos exitoso parece ser la comprensión no solo de lo que debe hacer el sistema en consideración, sino también por qué. Para captar el propósito de un sistema de información, se necesita un mecanismo para describir el comportamiento de la organización en la que operará el sistema. Este enfoque sugiere una mayor comprensión y modelado de los objetivos organizacionales y la forma en que estos objetivos se vuelven operativos (Loucopoulos, 1995; Verma et al., 2017), es por ello la importancia de involucrarlo e interpretar sus necesidades mediante el uso de modelos los cuales generen puentes de comunicación a través de estereotipos que representan de manera natural.

Cuanto más real el bosquejo, el producto digital cubrirá sus necesidades y el tan ansiado nivel de satisfacción será alcanzado porque un proceso de la cadena de producción que es consistente y preciso nos da la certeza para obtener un software de calidad.

La inclusión activa de los usuarios da legitimidad a los requisitos del sistema, y su sostenibilidad durante el desarrollo de este.

2.1.2 Epistemología de la Trazabilidad

La trazabilidad en la Ingeniería de Software es un método de seguimiento que viabiliza la obtención de un producto automatizado que responda a las necesidades expresadas por los usuarios lo más exacto y fiable posible para responder un problema específico. La trazabilidad se retroalimenta desde el análisis hasta la puesta en producción de un producto de software, por cada uno de los actores/roles que forman el equipo de proyecto.

Esta capacidad para inferir la vida de un requisito en ambas direcciones, desde su fase de elicitación, desarrollo e incluso hasta su uso después de la implementación y, en el caso de las metodologías ágiles este proceso debe ser dinámico para cada iteración dentro de alguna de estas fases (Lindvall y Sandahl, 1996; Lucassen, Dalpiaz, van der Werf, Brinkkemper y Zowghi, 2017). Al respecto, hay menciones que sólo abordaban la fase de desarrollo, definiéndola como la relación que se generaba entre dos o más productos del proceso, sin embargo, en la cadena de producción del software, los cambios o propuestas de mejoras son inminentes e inevitables, en ese sentido, se requiere “identificar” de manera precisa y óptima el impacto causado por los cambios, por lo tanto, las trazas generadas en el proceso de producción generan valor a corto o largo plazo indudablemente (Lucassen et al., 2017; Ramírez et al., 2017).

En la especificación de requisitos para las metodologías tradicionales se emplea el documento de Especificación de Requisitos mientras que las metodologías ágiles usan las Historias de Usuario, el uso de cualquiera de los dos artefactos atañe en las propuestas de mejora, anuencia del producto, cumplimiento del procedimiento y del proyecto, representación de la línea base y aprendizaje organizacional (Espinoza y Garbajosa, 2011), en ese sentido, independientemente de la metodología a utilizar queda claro que la documentación forma un recurso importante para una adecuada gestión en el proceso de desarrollo de software (Ramírez et al., 2017).

2.2 Antecedentes de la investigación

2.2.1 Trabajos Relacionados

2.2.1.1 *Trazabilidad de requisitos basada en el comportamiento a través de pruebas de aceptación automatizadas*

La utilidad de implementar la trazabilidad en los requisitos está en que establece las bases para el análisis de los efectos de un cambio, Aumento de comprensión de la aplicación, El proceso de desarrollo de software es más rápido (Mader y Egyed, 2015).

Las Historias de Usuario (HU) como artefacto bandera de las metodologías ágiles, sirve para la definición de requisitos que utilizados en un equipo de proyecto con un lenguaje natural existiendo el riesgo que cause errores en la interpretación y, en consecuencia, que la comprensión de las necesidades sea atendida con soluciones incorrectas.

La valoración de los usuarios y las relaciones existentes entre los procesos y sus instrumentos, la relevancia de una solución sobre su acervo documental, la relevancia del aporte del usuario en la definición del proceso y; la capacidad de respuesta frente a la celeridad de los procesos sobre el objetivo empresarial (Manifiesto for Agile Software Development, 2001).

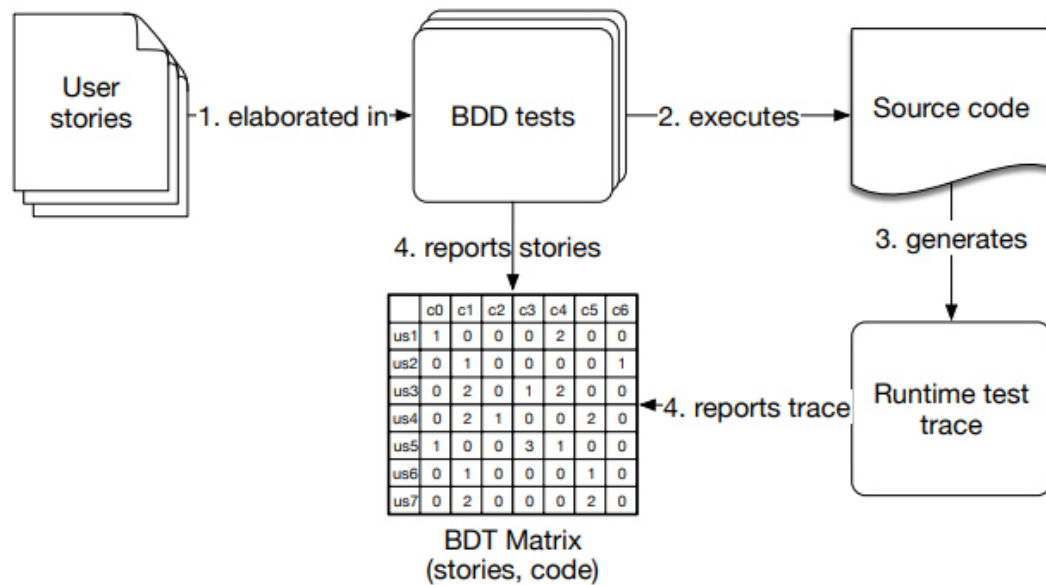
En la metodología ágil XP las historias de usuario suplen el documento de especificación funcional y los casos de uso de la metodología tradicional como en RUP. Las “historias” son descripciones en primera persona de las acciones realizadas por un rol en específico de forma breve y con un lenguaje coloquial (Beatty y Chen, 2012), su construcción se compone de expresiones cortas que narran funcionalmente las acciones en primera persona del usuario que es el que ejecuta algún rol en particular. Su adopción representa el 50% debido a su aplicabilidad en prácticas de Desarrollo Ágil como Scrum (Cao y Ramesh, 2008).

En 2017, Lucassen, Dalpiaz, van der Werf, Brinkkemper y Zowghi plantean un método de trazabilidad de requisitos basado en el comportamiento en un contexto de desarrollo ágil, estos deben ser abordados con métodos no invasivos desde sus etapas iniciales (elicitación) para garantizar el éxito de su propuesta, en ese sentido, el “estar siempre ahí” requiere estar implícitamente durante el proceso de ingeniería. El despliegue de su propuesta se realiza entre los requisitos de la historia del usuario y el código fuente

aprovechando las pruebas de aceptación automatizadas que se crean como parte del proceso de desarrollo basado en el comportamiento.

Figura 1

Método de trazabilidad basado en el comportamiento



Nota: La figura representa el flujo de comunicación entre los componentes que conforman el método de trazabilidad basado en el comportamiento. Tomado de *Trazabilidad de requisitos basados en el comportamiento a través de pruebas de aceptación automatizadas*, por Lucassen, Dalpiaz, van der Werf, Brinkkemper y Zowghi, 2017. pág.2

En la Figura 1, se visualiza las Historias de Usuario formuladas que contienen en detalle especificaciones convirtiéndose en la entrada para codificar breves pruebas en el bloque denominado BDD tests, la ejecución de estas pruebas genera código fuente el cual se introduce en el bloque Runtime test trace que finalmente genera un reporte de trazabilidad. La Figura 2, muestra un ejemplo de pruebas de Aceptación del Caso de Estudio de la Investigación EventCompany (Aplicación web de venta de entradas a eventos) las líneas de 1 al 4 describen la especificación de una Historia de Usuario y los numerales del 5 al 14 describen un escenario, esta información será utilizada en Python tiene traceback, .Net viene con Environment.StackTrace y Java admite getStackTrace según sea el caso.

Figura 2

Prueba de Aceptación del ejemplo de prueba EventCompany

```
1: Feature: Contact Form
2:   As a Visitor
3:   I want to use the contact form
4:   so that I can contact the organizer.

5: Scenario: Submit contact form
6:   Given I go to the "contact" page
7:     visit_path_to(contact_page)
8:   When I enter "Hello World" in the "Question" field
9:     fill_in('Question', with: 'Hello World')
10:  And I submit the form
11:    click_button('Submit')
12:  Then the organizer receives a message
13:    open_email('organizer@webcompany.com')
14:    expect(current_email).to have_content 'Hello World'
```

Nota: Tomado de Trazabilidad de requisitos basados en el comportamiento a través de pruebas de aceptación automatizadas, por Lucassen, Dalpiaz, van der Werf, Brinkkemper y Zowghi, 2017. pág. 2

Finalmente, en la Figura 3, se muestra el seguimiento generado en Ruby BDT Tracer, luego de la ejecución del caso de prueba EventCompany.

Figura 3

Fragmento de Salida para la Historia de Usuario de prueba EventCompany

	File	Class:method	us#	BDD scenario	BDD test step
1	classes/user.rb:10	User:current_user	us1	features/contact.feature:5	features/contact.feature:6
2	classes/user.rb:22	User:track_visitor	us1	features/contact.feature:5	features/contact.feature:6
3	classes/form.rb:15	Form:process	us1	features/contact.feature:5	features/contact.feature:10
4	classes/user.rb:22	User:track_visitor	us1	features/contact.feature:5	features/contact.feature:10

Nota: Tomado de Trazabilidad de requisitos basados en el comportamiento a través de pruebas de aceptación automatizadas, por Lucassen, Dalpiaz, van der Werf, Brinkkemper y Zowghi, 2017. pág. 23.

Lucassen etl., 2017 demostraron que el procesamiento de rastros en tiempo de ejecución a través de las pruebas de aceptación automatizadas, enfatizando en las clases y métodos de la historia de usuario.

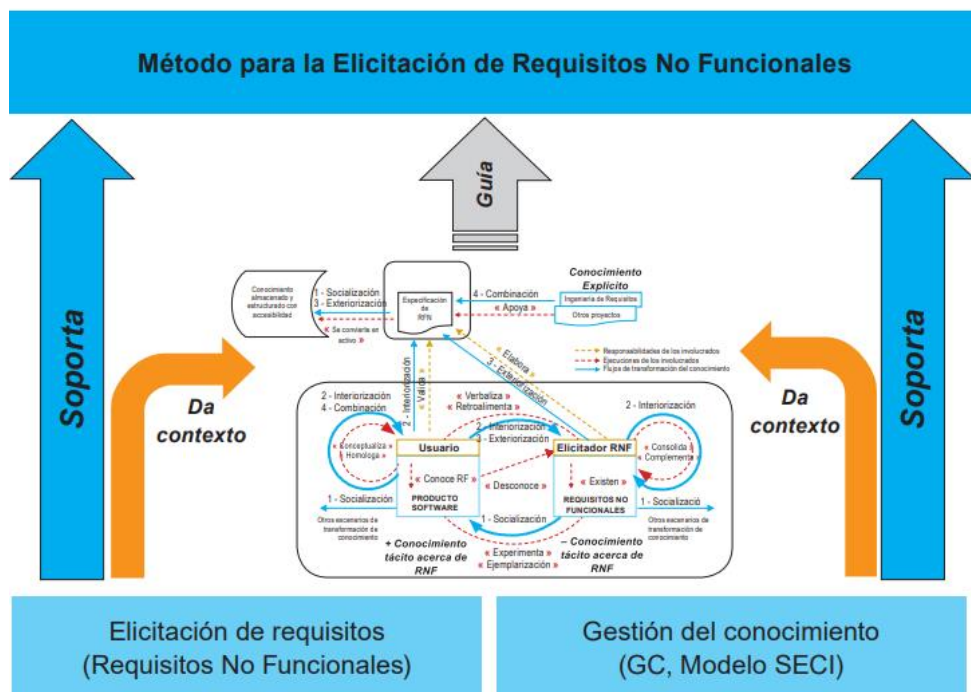
2.2.1.2 Elicitación de requisitos no funcionales basada en la gestión de conocimiento: el marco de trabajo Merlinn

Siendo los Requisitos Funcionales (RF) y los Requisitos No Funcionales (RNF) los 2 productos a obtener en la fase de Elicitación de Requisitos, los RNF deben enmarcarse en un contexto específico para que el software pueda ser usado y cumpla con un nivel adecuado de aceptación del usuario, al respecto, se precisa que para (Cysneiros y Yu, 2004) la calidad de un producto de software está determinado en tanto los 2 productos se realicen con la misma diligencia es decir los Requisitos Funcionales (RF) y los Requisitos No Funcionales (RNF).

En ese sentido, al tener la misma relevancia estos 2 elementos, sin embargo, la revisión de la literatura da cuenta que existe poco desarrollo de mecanismos adecuados para el relevamiento de los RNF.

Figura 4

Modelo conceptual del marco de trabajo Merlinn



Nota: La figura representa los componentes que conforman la propuesta del marco de trabajo Merlinn. Tomado de *Elicitación de requisitos no funcionales basada en la gestión de conocimiento: el marco de trabajo Merlinn*, por Buitrón, et al., 2017, Revista Ingenierías Universidad de Medellín. pág. 163

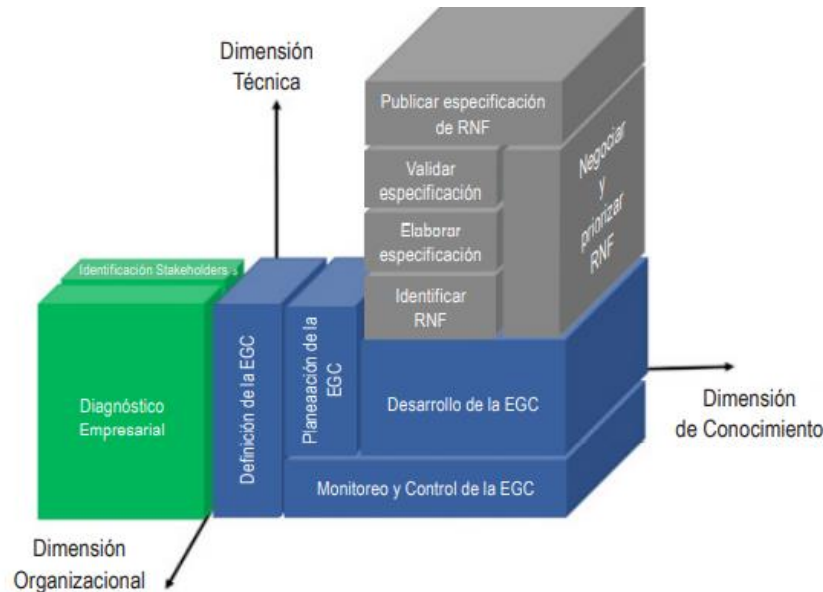
Al respecto, (Buitrón Ruiz, S. L., Flores Ríos, B. L. y Pino Correa, F. J., 2017) proponen un marco de trabajo denominado Merlinn basado en la gestión del conocimiento que permitiera relevar los Requisitos No Funcionales para ser integrados a la fase de Elicitación de Requisitos.

La Figura 5, muestra los 3 componentes que forman parte del marco conceptual de Merlinn:

1. Núcleo conceptual de transformación del conocimiento en la fase de Elicitación de Requisitos, basado en el modelo SECI el cual plantea que a partir de los activos institucionales se pueden generar nuevos conocimientos que en un primer momento se enfoca en sociabilizar la información entre el usuario y el elicitor para luego externalizar el conocimiento generado en un documento que plasme los Requisitos No Funcionales que finalmente deben ser validados por los usuarios (finales y técnicos).
2. Dimensiones para la Especificación de Requisitos No Funcionales basada en la gestión de conocimiento enmarcadas en las siguientes dimensiones:
 - 1) Dimensión de Conocimiento que plantea la valoración del conocimiento por niveles, la evolución o trayectoria del conocimiento implícito al explícito o su refinamiento para un mejor entendimiento de los Requisitos No Funcionales.
 - 2) Dimensión Organizacional que sirve para establecer los mecanismos en gestión y transferencia del conocimiento y los recursos existentes en las organizaciones para el tratamiento de los Requisitos No Funcionales, asimismo, considera identificar los procesos de negocio asociados al dominio pues ellos delimitan el alcance de un proyecto de software.
 - 3) Dimensión Técnica que en la figura 5 se visualiza la adaptación del proceso de Especificación de Requisitos No Funcionales (ERNF) bajo el estándar ISO 12207, asimismo aborda el enfoque de Gestión del Conocimiento para identificar, elaborar, validar, consensuar, priorizar y publicar la Especificación de los Requisitos No Funcionales.

Figura 5

Modelo conceptual del marco de trabajo Merlinn



Nota: La figura representa el método propuesto para el proceso de Elicitación de Requisitos No Funcionales basado en la Gestión del Conocimiento. Tomado de *Elicitación de requisitos no funcionales basada en la gestión de conocimiento: el marco de trabajo Merlinn*, por Buitrón, et al.,2017, Revista Ingenierías Universidad de Medellín. pág. 166.

3. Método para la ERNF basado en GC

El método propuesto requiere como entrada un Diagnóstico empresarial y determinar quiénes son los stakeholder para evaluar el nivel de implementación de los procesos de Gestión de Conocimiento a nivel institucional o la dificultad del dominio a implementar esperando como resultado las siguientes salidas: tamaño y tipo de la organización, mecanismos de almacenamiento de la información, lista de procesos de negocio involucrados y lista de mecanismos de comunicación utilizados por los stakeholders dentro de la organización (Buitrón et al.,2017). A continuación, estas salidas se convierten en insumo para la Definir y Planear la Estrategia de Gestión del Conocimiento y posteriormente aplicarlas en el proceso de Especificación de Requisitos No Funcionales obteniendo la conversión de conocimiento tácito a explícito, y viceversa. Por último, se

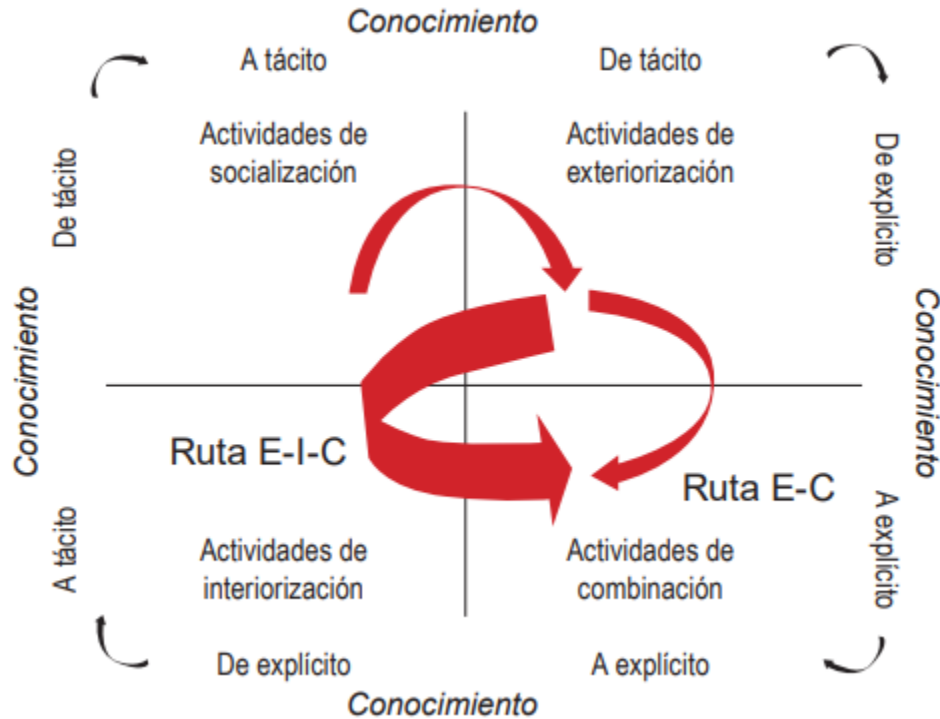
debe desarrollar la Estrategia de Gestión del Conocimiento definida y planificada en la fase anterior empleando el Núcleo conceptual de transformación del conocimiento en la fase de Elicitación de Requisitos.

La ejecución correcta del método Merlino se garantiza haciendo uso de los procesos e instrumentos que a continuación se detallan: Proceso de Diagnóstico Empresarial y Proceso de identificación de stakeholders (formato FM-DE-IS), Proceso de definición de la estrategia de gestión de conocimiento (Formato FM-DEGC), Proceso de planeación de la estrategia de gestión de conocimiento (Formato FM-PEGC), Proceso de desarrollo de la estrategia de gestión de conocimiento (Formato FM-DESEGC), Proceso de monitorización y control de la estrategia de gestión de conocimiento (Formato FM-MyCEGC)); el Proceso de desarrollo de la estrategia de gestión de conocimiento requerirá la aplicación de los escenarios de transformación que ocurren en la ERNF: i) escenario de instauración del conocimiento , ii) escenario de configuración del conocimiento, iii) escenario de afianzamiento del conocimiento y iv) escenario de institucionalización del conocimiento propuestos por Buitrón, et al., 2017. Para ello según la imagen x define las rutas de transformación.

El marco de trabajo despliega cuatro rutas: 1) Ruta S-E ubicada en el cuadrante superior derecho y que en sentido horario permite obtener un primer producto explícito, 2) Ruta E-C en el cuadrante inferior derecho y en sentido unidireccional obtenemos conocimiento explícito de combinación, 3) Ruta E-I-C que define una nueva combinación sin embargo considera la transformación del conocimiento explícito al tácito y 4) Ruta E-I que según la Figura 6 se logra un refinamiento progresivo durante el proceso de Elicitación de Requisitos No Funcionales bajo la consideración de aumentar el grado de interiorización.

Figura 6

Rutas de transformación del conocimiento tácito y explícito en la dinámica organizacional



Nota: La figura representa Propuesta de Rutas de transformación de conocimiento tácito-explícito del marco de trabajo Merlino. Tomado de *Elicitación de requisitos no funcionales basada en la gestión de conocimiento: el marco de trabajo Merlino*, por Buitrón, et al., 2017, Revista Ingenierías Universidad de Medellín.pp 170.

La implementación del marco de trabajo Merlino se realiza bajo el supuesto que el proceso de Gestión del Conocimiento se encuentra o requiere ser institucionalizado y, para extender su aplicabilidad al proceso de Elicitación de Requisitos dentro del contexto de la ingeniería de software.

Con el fin de validar preliminarmente el marco de trabajo Merlino, el caso de estudio se aplicó en una empresa desarrolladora de software definiéndose roles (un asesor del proceso de investigación, tres equipos técnicos de la empresa, grupo de usuarios finales (operarios y jefes de áreas)) y la ejecución de las actividades consideradas en el Plan de Elicitación de Requisitos.

En la recolección de datos y posterior aplicación del marco de trabajo Merlino de los 3 proyectos seleccionados, se identificó una lista de características no funcionales que obtuvieron mayor número de características asociadas a eficiencia de desempeño y usabilidad. Por ejemplo, para los RNF Eficiencia: tiempo de cargue/consultas de información, Tiempo de obtención de cálculos y/o resultados, Tiempo de almacenamiento en base de datos, etcétera y los RNF Usabilidad: Plataforma intuitiva y de fácil uso, Ordenamiento de información visual según criterios, Definición de información opcional, entre otros.

2.2.1.3 Factores sociales y humanos que afectan el proceso de educación de: requisitos una revisión sistemática

En la Ingeniería de Requisitos, en referencia al proceso de obtención de los requisitos que debe cumplir un software para ser aceptado y alcanzar la calidad deseada, autores como (Carrizo, Dieste, y Juristo, 2014) han descrito diversas técnicas como las entrevistas, Análisis de formularios, prototipado, etcétera que establecen mecanismos para la definición de los requisitos en un contexto puntual. Sin embargo, en la revisión realizada por (Donoso y Vega, 2017) identificaron que la literatura existente no abordaba ampliamente los aspectos humanos y sociales en la fase de elicitación de requisitos, en ese sentido, proponen identificar las características que no son consideradas típicamente, para establecer aquellos factores que afectan negativamente en el proceso de elicitación. Esto hace de la ingeniería de requisitos una disciplina singularmente compleja al intervenir el factor humano factor responsable de que la Ingeniería de Requisitos aborde aspectos sociales y culturales y no solo técnicos (Goguen, 1994).

En el análisis realizado por (Donoso y Vega, 2017) consideran la inminente participación de aspectos sociales y humanos en la comunicación entre los usuarios y especialistas a cargo del proceso denominado por los autores como educación en referencia a su uso semántico en el campo de la Ingeniería de Software y su significado del idioma inglés, describiendo “educación” como la capacidad de transferencia de información de un individuo a otro mediante artefactos, técnicas o metodologías definidas con el objetivo de capturar requisitos y descubrir necesidades de un cliente.

La revisión de la literatura sostiene que la comunicación es la base de la construcción de especificaciones sólidas y para ello se necesita un medio para transmitirla, en ese aspecto, es la entrevista uno de los canales más usados para ese propósito y que exige la capacidad de enfrentar las emociones y gestionarlas, de manera que podamos obtener las necesidades de los usuarios y con la asistencia técnica del especialista establecer cuáles sí se sistematizarán.

En la Tabla 1 se visualiza que las ocurrencias asociadas al factor *Comunicación* suman 18 y es uno de los 6 factores identificados en el presente estudio que pueden afectar negativamente el proceso de educación por lo que los autores recomiendan su consideración, en el “eje y” la lista de factores sociales y humanos agrupados en seis (06) categorías, resultado de la revisión sistemática de los procesos de educación en la que participan stakeholders y los especialistas, en el “eje x” se visualizan los títulos P01 al P25 correspondiente a los códigos asignados a los 25 artículos seleccionados para análisis.

Tabla 1

Factores sociales y humanos agrupados por categorías.

Factores sociales y humanos (categoría)	P01	P02	P03	P04	P05	P06	P07	P08	P09	P10	P11	P12	P13	P14	P15	P16	P17	P18	P19	P20	P21	P22	P23	P24	P25
Ambigüedad	x										x	x			x	x	x			x		x		x	
Conocimiento tácito				x		x							x				x					x			
Comunicación		x	x	x	x				x	x	x	x	x	x	x		x	x	x	x		x		x	x
Entendimiento					x				x	x	x	x		x					x						
Ámbito cognitivo, emocional y motivacional				x		x						x				x									
Resistencia				x							x					x									
Valores humanos							x				x														

Nota. Tomado de *Factores sociales y humanos que afectan el proceso de educación de requisitos: una revisión sistemática*, por Donoso Barraza, B. y Vega Zepeda, V., 2017, p. 77.

La revisión literaria de Donoso et al., 2017 determinó un conjunto de factores con la posibilidad de afectar el proceso de educación y fueron advertidos por otros autores, en ese sentido, el presente estudio concluye que los especialistas a cargo del proceso de educación deben desarrollar aptitudes asociadas a mejorar el nivel de comunicación con el usuario y a disminuir el impacto. Su preparación a nivel emocional debe enfocarse en desarrollar habilidades como:

- Capacidad de identificar información relevante que se encuentra oculta dentro del dominio del producto, para ellos debe considerar técnicas cognitivas o apoyarse con el uso de un artefacto.
- Comunicación efectiva fortaleciendo constantemente la atención del interlocutor pudiendo ser verbal y escrita.
- Capacidad de negociar, relevar y tomar decisiones frente a los stakeholders.
- Diseñar Especificaciones simples, estructuradas, objetivas y sin ambigüedades.
- Gestión de herramientas que promuevan la participación de los involucrados para mantenerlos motivados y neutralizar su resistencia al cambio.
- Sensibilización de los beneficios en el uso de un nuevo software trae consigo como la mejora continua y competitividad de los negocios.

Sin embargo, es conocido que las habilidades listadas en el párrafo anterior describen la dimensión social, un área poco explorada en la Ingeniería software debido a que los especialistas priorizan los aspectos técnicos sobre los sociales (Apsihvalka, Donina, y Kirikova, 2009).

2.2.1.4 Impacto y desafíos de la ingeniería de requisitos en metodologías ágiles: una revisión sistemática

El alcance de la ingeniería de Requisitos se establece en el uso sistemático de métodos, técnicas, lenguajes y herramientas que permitan definir un adecuado análisis, especificación de las condiciones que debe cumplir una solución de software, documentación, gestión de las mejoras funcionales, aseguramiento de la consistencia frente a los cambios, entre otros (Hsia, Davis, y Kung, 1993).

La ingeniería de Requisitos comprende dos principales procesos: 1). -Desarrollo de Requisitos: captura, análisis, especificación y validación. 2). -Gestión de Requisitos: trazabilidad de los requisitos (Bourque y Fairley, 2014). En el primer proceso se debe descubrir los requisitos para desarrollar un sistema, esto se logra comunicándonos con las

partes interesadas (clientes y usuarios del sistema), para obtener información sobre el dominio de la aplicación y los servicios que el sistema debe proporcionar, el rendimiento del sistema requerido, las limitaciones de hardware, etcétera. La eficacia de la Ingeniería de Requisitos requiere una mejor atención del dominio área, las condiciones en las que se ejecutará, requisitos de las partes interesadas del producto, incluidos desarrolladores, diseñadores, usuarios finales y clientes (Sommerville y Sawyer, 1997).

Al respecto, la metodología ágil surge como alternativa a las metodologías tradicionales que son consideradas invariables estableciendo un proceso secuencial para el desarrollo de software volviéndolo predecible y en consecuencia eficiente (Sehrish, Shahid, Asim y Amr, 2017), es decir la captura de requisitos se hace una sola vez hasta que llega al desarrollador para materializar los requisitos y no es hasta su despliegue en donde se evidencia el éxito o no del proyecto, en caso de ser negativo tendría que volverse a realizar cada una de las fases del largo y tedioso proceso.

La aplicación de las metodologías ágiles en la ingeniería de requisitos según la revisión sistemática realizada por (Sehrish et al., 2017), mencionan el uso de los prototipos los cuales responden a consultas de contenido de los requisitos y genera un mejor nivel de entendimiento a los clientes, esto resulta beneficioso porque podrán describir sobre lo que necesitan realmente, además el enfoque ágil tiene la flexibilidad de “reajustar”, los requisitos no válidos que son previamente son evaluados pero que inminentemente generan un coste adicional, riesgo que según (Cao y Ramesh, 2008) puede disminuir siempre que se aseguren las siguientes condiciones: la especificación en forma escrita debe ser cara a cara, empleo de reuniones de revisión y pruebas de aceptación, prototipos, desarrollo de pruebas base, planificación constante para la gestión de requisitos de cambio y la priorización de requisitos.

Tabla 2

Desafíos de la ingeniería de requisitos en metodologías ágiles.

Características	Detalles	Desafío
Comunicación Directa	Comunicación entre clientes y miembros del equipo, documentación mínima en Ágil	¿Cómo rastrear cambios en requisitos?
Gran interacción de clientela.	La interacción con el cliente es requerido para grandes cantidad de comentarios	Gran carga de trabajo
Cambio requisitos	Manejar cambios	Hacer de nuevo
Omisión de los Requerimientos No Funcionales	Las historias de usuario solo almacenan requerimientos funcionales	Seguridad y usabilidad
Estimación de tiempo y presupuesto	No puedo estimar debido a cambios de requisitos	Alto sobrecoste de proyecto
Innovación en Ingeniería de requisitos	Creatividad en Ingeniería de requisitos	Problema de Creatividad en agilidad y actualización
Desaparición de requisitos	¿Cómo descubrir requisitos desaparecidos?	Gran cantidad de iteraciones para requisitos desaparecidos

Nota. Esta tabla analiza algunas características del enfoque ágil y los desafíos de cara a cubrir sus deficiencias. Adaptado de *Impacto y desafíos de la ingeniería de requisitos en metodologías ágiles: una revisión sistemática*. Sehrish, Shahid, Asim y Amr (2017).

De acuerdo con la característica que se muestra en la Tabla 2, descrita como la Omisión de los Requisitos No funcionales, se da detalle que en el enfoque ágil sólo se almacena requisitos funcionales, en ese sentido los autores plantean como desafío promover un marco de solución que extienda la implementación de requisitos no funcionales como la seguridad y usabilidad en enfoques ágiles.

2.2.1.5 Un nuevo modelo adaptativo para los métodos de ingeniería web para desarrollar una web moderna Aplicaciones)

El desarrollo de las aplicaciones web y sus constantes propuestas de mejoras están enfocadas en la fase de desarrollo e implementación, no lográndose cubrir aspectos referidos al aseguramiento de la calidad con una temprana definición de los requisitos no funcionales, en ese sentido, Aguilar, Garrigós, Casteleyn y Mazón (2011) proponen la creación de un algoritmo basado en el óptimo de Pareto para identificar las configuraciones cliente/servidores deseables a través de la optimización de requisitos no funcionales .

En el campo de Ingeniería Web, se puede encontrar un bagaje de metodologías tradicionales de definición de requisitos, que se han sido adaptadas para soportar el aspecto interactivo de las interfaces en las RIAs (UML-based Web Engineering y Web Modeling Language), así como las que actualmente se han creado exclusivamente para la fase de desarrollo (Object-Oriented Approach for Web Solutions Modeling, Object-Oriented Hypermedia Design Method, Internet Application Modelling Language) Aguilar et al.,2011.

Wakil y Jawawi (2015) propusieron una categorización para las Aplicaciones Web Modernas en 4 características fundamentalmente 1) Enriquecidas de Internet: Las Aplicaciones Enriquecidas de Internet o RIA por sus siglas en inglés, proponen mejorar el aspecto de aplicaciones de escritorio bajo un entorno web, con tecnologías de comunicación dándoles mayor flexibilidad y haciéndolas más intuitivas además permiten novedosas operaciones, como distribución de datos; cálculo de página parcial; y trabajo desconectado (Comai y Carughi, 2007). 2) Web Ubicua, característica que requiere que las aplicaciones tengan la capacidad de desplegarse en cualquier dispositivo móvil atendiendo a las características como tamaño de pantalla, almacenamiento local, capacidad de red, otras necesidades más específicas son las que requieren información acerca de la ubicación, el tiempo y servicios personalizados, es decir constituirse como parte del medio ambiente en el que se despliega de una forma personalizada (Finkelstein, Kappel y Retschitzegger, 2002). 3) Web Semántica: Permite crear soluciones tecnológicas que permitan publicar datos legibles y estándares a través de Resource Description

Framework (RDF) y la eliminación de referencias al contenido y, 4) Web Inteligente: Su escala de transformación es inminentemente debido a la accesibilidad a la web semántica, asimismo, la implementación de infraestructura innovadora para acelerar la ejecución de aplicaciones web semánticas.

Tabla 3

Aplicaciones Web más recientes y sus características.

Features	Web applications
Client/Server, Rich UI, Rich Client	RIA
Anywhere, Anytime, Anyplace, Anymedia, Anydevice, Adaptation, Context-aware	UWA
Ontology, Rich UI, RDF, semantic hyperlink, behaviour	SWA
Web Personalization, Web Mining, Semantic Web, Intelligent Agents	IWA

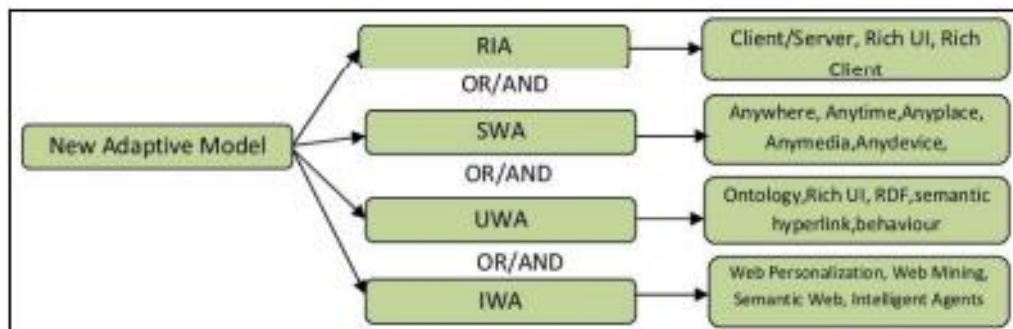
Nota. Esta tabla analiza algunas características de las aplicaciones web inteligentes. Tomado de *A New Adaptive Model for Web Engineering Methods to Develop Modern Web Applications*. Wakil, y Jawawi (2015), pág. 36.

De acuerdo con la información de la Tabla 3, las aplicaciones web inteligentes incluyen características como minería web, web semántica, Personalización web y agentes inteligentes.

La ausencia de un método de ingeniería web único para admitir nuevas características de aplicaciones web modernas debido a la evolución de la tecnología web, el presente estudio, en la Figura 7 se define un nuevo modelo adaptativo para soportar todas las características

Figura 7

Propuesta de Modelo Web Adaptativo para soportar todas las características

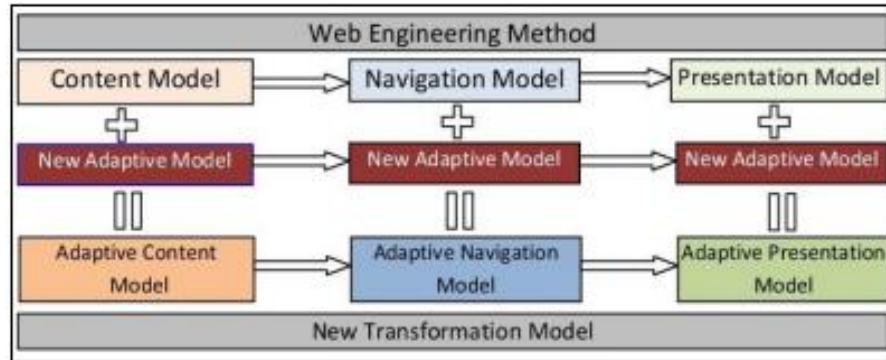


Nota. Esta tabla analiza algunas características de las aplicaciones web inteligentes. Tomado de *A New Adaptive Model for Web Engineering Methods to Develop Modern Web Applications*. Wakil, y Jawawi (2015), pág. 36.

En la ingeniería Web las cualidades de presentación (hipermedia adaptativa) y navegación (ingeniería web) que forman parte de una propuesta de aplicación de web moderna representan un pre-requisito crítico para encarar la diversidad de usuarios que la usaran. Los requisitos de los métodos de ingeniería web del nuevo modelo adaptativo se visualizan en la Figura 8, además (Wakil, y Jawawi, 2015) incorporan a la estructura los métodos de la Ingeniería Web con la finalidad que sea adaptable a todos los modelos.

Figura 8

Configuración de los métodos que componen el nuevo modelo adaptativo



Nota. Tomado de *A New Adaptive Model for Web Engineering Methods to Develop Modern Web Applications*. Wakil, y Jawawi (2015), pp 37.

2.2.1.6 Ingeniería de requisitos orientada a trabajos por hacer: un método para definir historias de trabajo

En el enfoque de Modelado Orientado a Objetivos propuesto en (Yu, 1995), establece 2 modelos en los que aplica reglas semánticas respecto a los enlaces entre sus elementos: 1) Modelo de dependencias estratégicas (SD) define las posibles dependencias entre cada pareja de actores sociales. 2) Modelo de Justificación estratégica (SR) que establece posibles metas duras, metas blandas, planes, y recursos dentro de los límites de cada actor social y describiendo cómo estos objetivos pueden afectarse unos a otros y los requisitos para alcanzarlos. El planteamiento de (Agra. et al., 2015) analiza el comportamiento de estas relaciones y su impacto cruzado para comprender cómo los elementos pueden verse afectados.

La comprensión sustancial a través del Modelado Orientado a Objetivos pone en evidencia los “por qué” detrás del “qué” y el “cómo” representando su organización en términos de relación de dependencia entre actores (SD) y el modelo que captura las motivaciones, intenciones y fundamentos que implícitos a un software en proceso de construcción (SR), dentro de los límites (boundary) de un actor para refinar el modelo SD y añadirle capacidad de razonamiento.

Al comparar la ER “orientada a objetivos” y los “trabajos por hacer”, se identifican algunos puntos en común: ambos se centran en aspectos motivacionales y por otro lado se evidencian las diferencias: en la ER orientada a objetivos representa “lo que quiere hacer un actor” y se centra en la solución, los trabajos por hacer se escriben desde la perspectiva de un cliente para comprender sus problemas, sin embargo, considerando sus similitudes (Lucassen, Keuken, Dalpiaz, Brinkkemper, Sloof y Schlingmann, 2018) consideran que podrían servir para investigaciones futuras.

Las historias de trabajo se fundamentan en el método de la teoría de la innovación disruptiva. Jobs-to-be-Done (JTBD) (Clayton, 2016), una colección de principios que ayudan a descubrir y comprender las interacciones entre los clientes, sus motivaciones y los productos físicos que utilizan. En ese sentido, Lucassen et al., 2018 proponen un método para realizar Ingeniería de Requisitos denominado Jobs-to-be-Done (JTBD) y consta de cinco fases.

La Figura 9, muestra en el lado izquierdo el conjunto de actividades o fases y en el lado derecho los artefactos que componen la propuesta del Método Integrado para JTBD para desarrollo de software, las 5 fases tienen el siguiente alcance:

Fase 1: Realizar entrevistas, se establece diálogo con los potenciales clientes para identificar los problemas que estén afrontando y la forma de alcanzar sus objetivos.

Fase 2: Fase de Análisis, la información obtenida en la fase anterior es contextualizada, así como las motivaciones de los clientes para realizar los trabajos e historias iniciales.

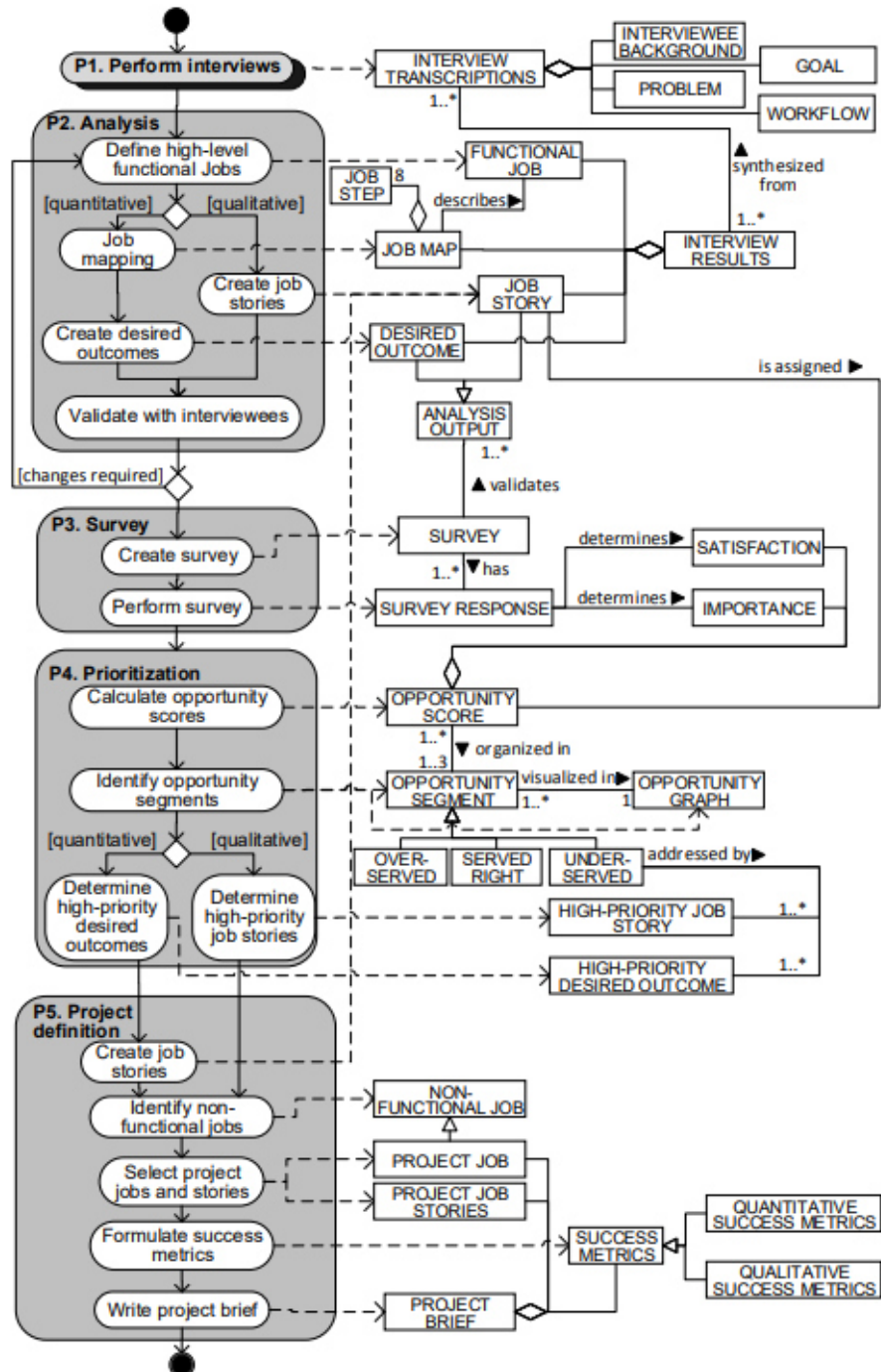
Fase 3: Fase de encuesta, este tipo de diálogo se realiza con una audiencia más amplia y sirve para validar lo avanzado hasta este punto.

Fase 4: Fase de priorización, de aquellos trabajos que fueron validados correctamente se identifican los que tienen mayor probabilidad para la innovación.

Fase 5: Fase de definición del proyecto, se seleccionan los trabajos e historias de trabajo para el desarrollo y se crea un resumen del proyecto que puede facilitar el seguimiento del proyecto de desarrollo.

Figura 9

Configuración de los métodos que componen el nuevo modelo adaptativo

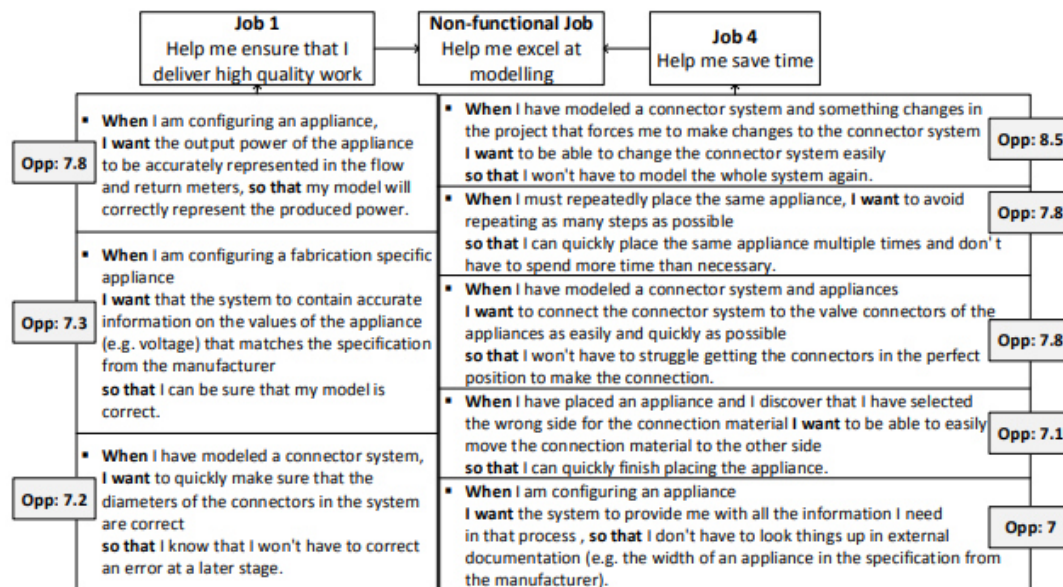


Nota. Tomado de *Ingeniería de requisitos orientada a trabajos por hacer: un método para definir historias de trabajo*, por Lucassen et al., 2018, pp 8.

Stabiplan es una empresa en el mercado que implementa soluciones para instalaciones mecánicas, electrónicas y de plomería con software de diseño asistido por computadora (CAD), los autores Lucassen et al., 2018 desarrollaron un caso práctico con el objetivo de utilizar su método que determine previamente qué funcionalidad ya sea nueva o una existente que permita incorporar un producto independiente a un nuevo segmento de clientes e incitar su uso. Se seleccionó trabajos no funcionales de bajo nivel 1 y 4: "Help me ensure that I deliver high quality work" y "Help me save time". El resultado total fue de 10 historias de trabajo, el criterio de inclusión corresponde sólo aquellas con puntuación 7,0 o superior. Esto da como resultado 8 historias de trabajo como se muestra en la Figura 10.

Figura 10

Configuración de los métodos que componen el nuevo modelo adaptativo



Nota. Tomado de *Ingeniería de requisitos orientada a trabajos por hacer: un método para definir historias de trabajo*, por Lucassen et al., 2018, pp 12.

El enfoque disruptivo del método enmarca a las empresas que no se definen por lo que producen, sino por las necesidades de sus clientes, las cuales se convierten en “la problemática a resolver”. La propuesta del método Jobs-to-Be-Done intenta resolverlo a través de la contextualización del problema que impulsa el comportamiento del cliente de acuerdo con la situación y sus motivaciones.

Lucassen et al., 2018 consideran su propuesta JTBD como una adaptación orientada a la industria de la metodología orientada a Objetivos que favorece el pragmatismo (usando texto y diagramas simples) sobre la solidez conceptual (por ejemplo, la distinción entre tipos de actores, metas y tareas, tipos de metas y las distinciones que permiten para el razonamiento formal sobre los gráficos de objetivos).

2.2.1.7 Conceptualización de requisitos utilizando historias de usuarios y Casos de uso: un experimento controlado

Los estereotipos usados para representar los requisitos de un sistema deben ser de permitir una aproximación sencilla a lo que el usuario realmente quiere, por ejemplo, los casos de uso de UML e historias de usuarios (Cohn, 2004), también se usan oraciones en lenguaje natural (Dalpiaz, Ferrari, Franch y Palomares, 2018), modelos semiformales (Mylopoulos, Chung, Yu, 1999 y Van, 2009), y lenguajes formales (Ciancarini, Cimato, Mascolo, 1997 y Ottensooser, Fekete, Reijers, Mendling, y Menictas, 2012).

La selección de una notación puede estar motivada para garantizar la comunicación entre las partes interesadas, para tareas analíticas, para establecer inconsistencias y detectar viabilidad, desarrollo de especificaciones, etc.

La cuestión de la investigación es conceptualizar los requisitos a partir de las notaciones más utilizadas: Casos de Uso e Historias de Usuario (Dalpiaz, Ferrari, Franch y Palomares, 2018).

En el presente estudio la comparación realizada acerca de la preferencia por parte de los usuarios entre el Caso de Uso y la Historia de Usuario, se afirma que la comprensión de los Casos de uso al tener un enfoque secuencial y orientado a proceso lo hace entendible, por otro lado, las historias de Usuario al poseer descripciones breves y que tiene un formato estándar facilita la comprensión de cada requisito de forma separada.

Los resultados indican que, para derivar modelos conceptuales, las historias de los usuarios se ajustan mejor que los casos de uso, por ejemplo, en los casos de uso, los actores generalmente se mencionan solo al principio de cada caso de uso (en la sección "actor"), y las acciones descritas se refieren implícitamente a la interacción entre el sistema y el actor. En las historias de usuario, por el contrario, los actores se expresan en cada historia de usuario en la parte "Como". Al igual que los actores, parece que en las historias de usuarios hay entidades que se repiten varias veces, ya que se utilizan en muchas operaciones, esto también llevó a una mejor identificación de tales entidades al derivar los modelos conceptuales.

2.2.1.8 Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales

El enfoque organizacional de las empresas actualmente debe atender un panorama de gestión por procesos de negocio que impacten a nivel transversal a todas las áreas de la entidad, con la finalidad de trabajar de manera integral a través del trabajo colaborativo (Daclin, Chen, y Vallespir, 2016).

En ese sentido, el trabajo colaborativo y capacidad de integración institucional debe garantizar la implementación de la interoperabilidad clasificada y definida como un atributo de calidad por la ISO/IEC 25010 (ISO/IEC 25010, 2011) con la capacidad de intercambiar información apoyado de herramientas tecnológicas que generaran un canal de comunicación adecuado y seguro, sin embargo, (Cenci, Estevez y Fillottrani, 2017) proponen un enfoque holístico para asegurar que la información fluya efectivamente.

Asimismo, la norma ISO 11354 (ISO 11354-1, 2011) define el alcance de la interoperabilidad en 4 niveles: 1) datos, gestión del intercambio de la data que incluye la absolución de diferencias sintácticas y semánticas; 2) servicios, determina cuáles son los servicios que el sistema debe tener, y solicitar si su fuente de consulta de datos es externa; 3) procesos, la institucionalización de la interoperabilidad ocasiona el trabajo colaborativo de todos los miembros de una empresa y 4) negocio, gestionar las relaciones al interno de la institución y con miembros externos e incorporar oportunamente en los procesos de negocio el resultado de la interacción. Al respecto, de los niveles mencionados

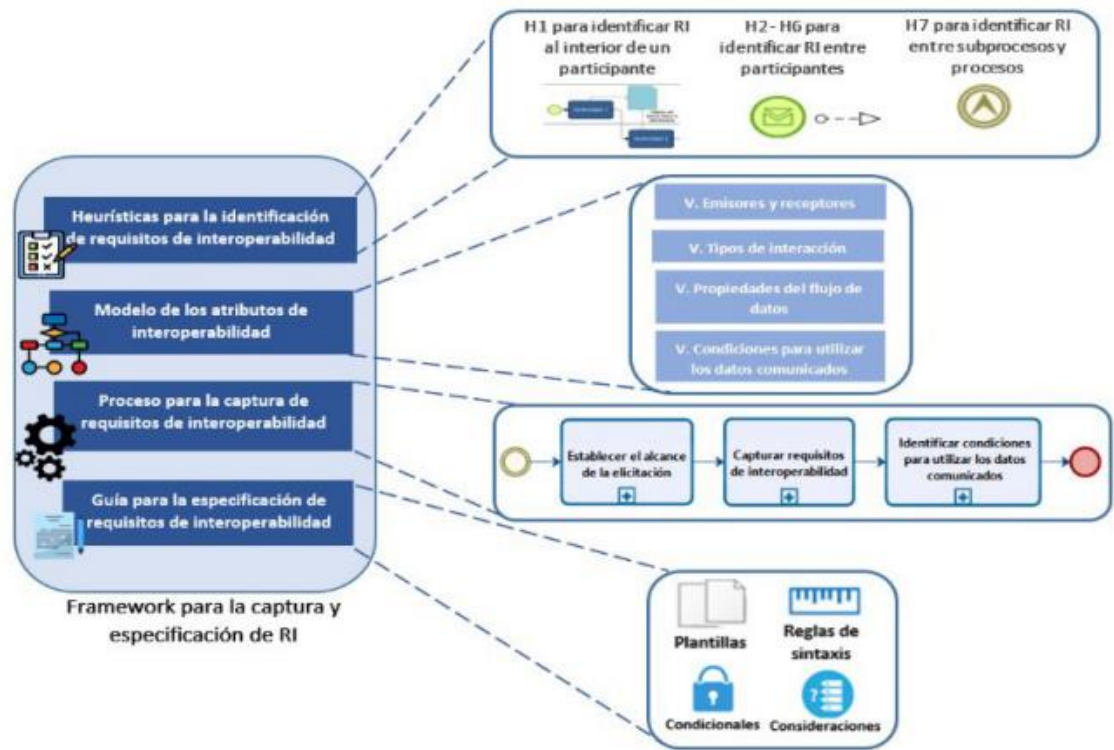
podemos entender que establece la hoja de ruta para una adecuada implementación de la interoperabilidad, la cual no sólo tiene componentes técnicos, sino que además el factor social que pueden influir en el éxito de su implantación (Motta, Marçal y Horta, 2017).

En la implementación o implantación de un sistema, resulta indispensable prestar atención a las actividades de elicitación de requisitos de interoperabilidad para especificar su carencia de intercomunicación y uso de información entre sistemas, representadas dentro de la institución como: roles, grupos de personas, áreas funcionales, software o hardware. La criticidad de esta fase se fundamenta en la precisión de la obtención y especificación de los requisitos en correlación a la carencia de intercomunicación y manejo de datos por parte de los usuarios de dirección de la organización, usuarios técnicos, usuarios finales del negocio, etc.

Paz, Pino y Buitrón (2020), proponen un framework que desde una vista empresarial se realice tanto la obtención como la especificación de requerimiento de interoperabilidad. El framework está compuesto por 4 componentes que se visualizan en la Figura 11.

Figura 11

Componentes del Framework para obtener y especificar los Requisitos de Interoperabilidad



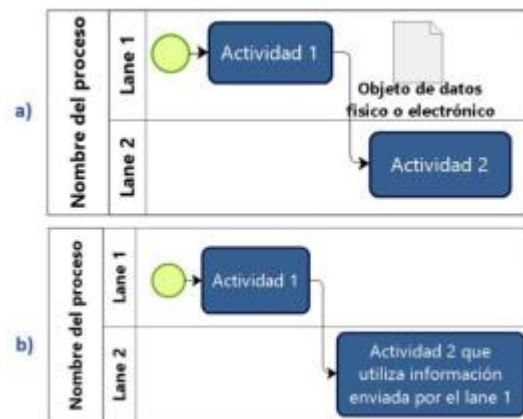
Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-242

A. Heurísticas para la identificación de requisitos de interoperabilidad, Tiene como objetivo abstraer los Requisitos de Interoperabilidad a partir de un modelo de proceso de negocio que para este caso de uso la notación BPMN que es el diagrama más usado para estos fines según (Pillat, Santos y Oliveira y 2019). Así mismo, el caso desarrolla 3 heurísticas:

- 1) Comunicación al interior del pool, como se visualiza en la Figura 12 estas comunicaciones pueden ser de forma implícita (escenario a) o explícita (escenario b).

Figura 12

Requisito de Interoperabilidad y su configuración dentro de un pool.

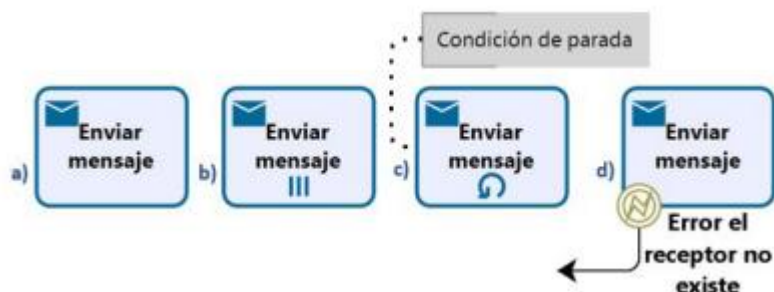


Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-242

- 2) Comunicación mediante actividades de envío, en la Figura 13 se visualizan los procesos para la obtención y especificación del Requisito de Interoperabilidad se debe evaluar si es un solo receptor (literal a), si tiene varios receptores (literal b o c), además el proceso del literal b analiza la condición de parada que posibilita detener el envío de mensajes; y además en el literal d) establecer las excepciones al enviar un mensaje.

Figura 13

Flujos de mensajes y sus respectivas configuraciones para su envío y recepción.



Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-243

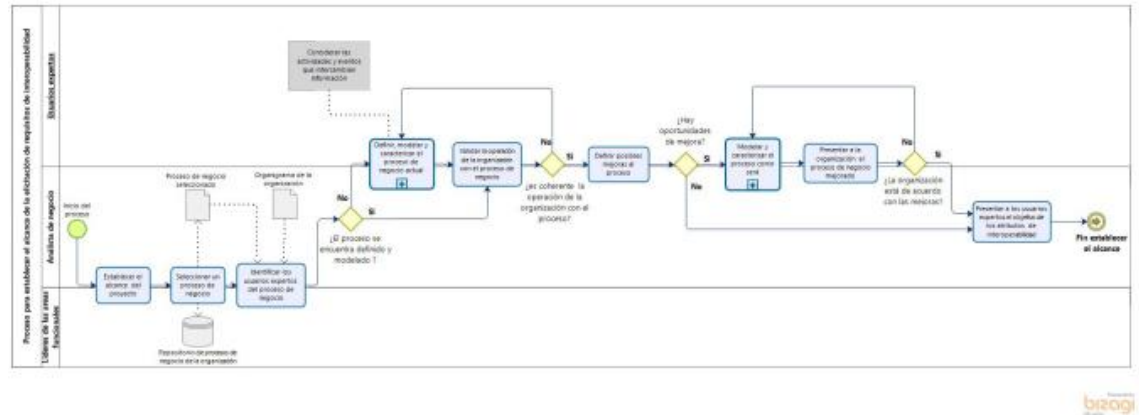
- 3) Comunicación mediante actividades simples con flujos de mensaje, de acuerdo con la Figura 14 el literal (a) existen 2 RI tanto para el flujo de salida como el de entrada, en el caso del literal (b) representa un envío asíncrono ya que el emisor debe esperar respuesta de su envío sin antes poder actuar sobre otra.

B. Modelo que describe los atributos que conforman un requisito de interoperabilidad; este modelo está enfocado a mostrar los requisitos de interoperabilidad a nivel de negocio y presenta 3 vistas: Vista de propiedades de flujo de datos, Vista de emisores y Receptores, Vista de tipo de interacción, Vista de restricciones para utilizar los datos comunicados.

C. Proceso que guía la captura de Requisitos de Interoperabilidad; los Requisitos de Interoperabilidad y sus atributos son establecidos en propiedades del flujo de datos, condiciones para utilizar los datos comunicados, las vistas de emisores y receptores y, tipos de interacción. Debido a la complejidad de la captura de los RI, el proceso se ha dividido en 3 subprocesos:

Figura 14

Primer Subproceso de establecer el alcance de la elicitación de un RI.

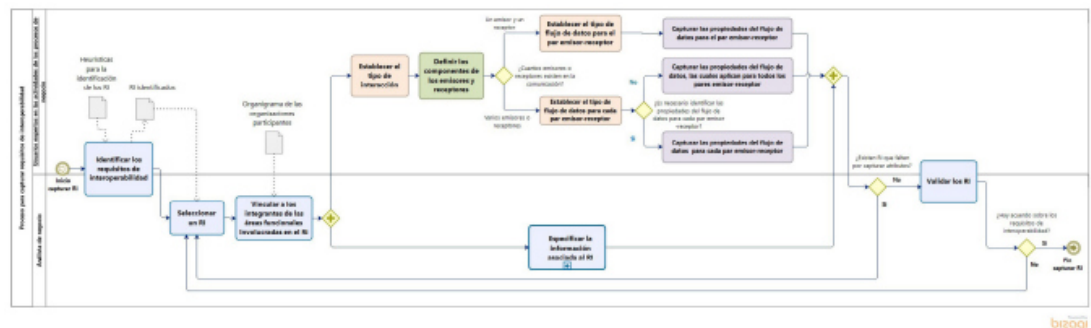


Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-245

El primer subproceso mostrado en la Figura 14 tiene como objetivo elegir un proceso de negocio a partir de cual capturan los requisitos de interoperabilidad y posteriormente se hacen propuestas de mejora; y finalmente son presentados a los usuarios expertos.

Figura 15

Segundo Subproceso para capturar atributos de un RI.

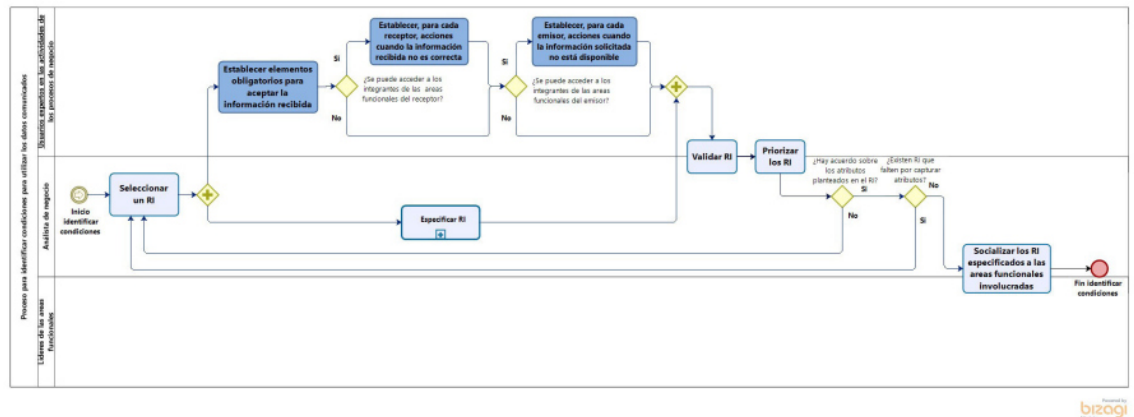


Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-245

El Segundo subproceso mostrado en la Figura 15 tiene como objetivo identificar un RI a partir de cual captura el requisito de interoperabilidad y sobre el cual, y sobre el son capturados y especificados los atributos de las vistas de emisores y receptores.

Figura 16

Tercer Subproceso para identificar condiciones para utilizar los datos.



Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-246

El Tercer subproceso mostrado en la Figura 16 tiene como objetivo identificar un RI sobre el cual serán capturados los atributos de la vista denominada “Condiciones para utilizar los datos comunicados”, finalmente son se priorizan, ratifican y sociabilizan.

La investigación en análisis desarrolla una Guía que muestra en detalle la información necesaria para especificar un Requerimiento de Interoperabilidad a nivel de negocio.

Figura 17

Guía propuesta para el registro de atributos de la vista de emisores, receptores y Tipos de interacción.

Vista de emisores, receptores y tipos de interacción		
*Tipo de flujo de datos	Seleccione los tipos de flujos de datos de las siguientes opciones: 1. Entre áreas funcionales de una organización ☐ 2. Entre áreas funcionales de diferentes organizaciones ☐ 3. Dentro de un área funcional de una organización ☐ 4. Entre un área y un individuo externo a la organización ☐	
*Tipo de interacción	Seleccione un tipo de interacción de las siguientes opciones: 1. Envío punto a punto ☐ 2. Enviar / recibir ☐ 3. Envío uno a muchos ☐ 4. Recibir muchos a uno ☐ 5. Enviar/recibir uno a muchos ☐ 6. Respuestas múltiples ☐ 7. Solicitudes contingentes ☐ 8. Solicitud con referencia ☐ 9. Solicitud transmitida ☐	
Componentes del emisor	*Organización:	<i>Escribir un nombre significativo de la organización</i>
	Área funcional:	<i>Escribir un nombre significativo del área funcional</i>
	Recurso que ejecuta la actividad o evento emisor:	<i>Escribir este campo de acuerdo a la regla 2</i>
	Actividad o evento:	<i>Comenzar con un verbo en infinitivo y corresponder a lo planteado en el proceso de negocio.</i>
	Repositorio de datos desde el cual se consulta la información:	<i>Escribir este campo de acuerdo con la regla 3.</i>
	Individuo externo:	<i>Escribir en este campo el rol del individuo externo a la organización</i>
Componentes del receptor	*Organización:	<i>Escribir un nombre significativo de la organización</i>
	Área funcional:	<i>Escribir un nombre significativo del área funcional</i>
	Recurso que ejecuta la actividad o evento receptor:	<i>Escribir este campo de acuerdo a la regla 2</i>
	Actividad o evento:	<i>Comenzar con un verbo en infinitivo y corresponder a lo planteado en el proceso de negocio.</i>
	Repositorio de datos en el cual se almacena la información:	<i>Escribir este campo de acuerdo con la regla 3.</i>
	Individuo externo:	<i>Escribir en este campo el rol del individuo externo a la organización</i>
Nota: En caso de seleccionar el tipo de flujo de datos entre un área y un individuo externo a la organización, debe diligenciar el campo individuo externo dentro de los componentes del emisor o del receptor.		

Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-246

Guía propuesta para el registro de características asociadas a la vista de propiedades del flujo de datos.

Nota. Tomado de *Framework para la elicitación de requisitos de interoperabilidad en el contexto de sistemas organizacionales*, por Paz et al., 2020, INGE CUC. pp-247

Para la implementación del framework propuesto los autores utilizaron los procesos de negocio de 3 casos, y distintos objetivos para usar los RI elicitados.

2.2.1.9 Klein: Proceso de desarrollo de software con trazabilidad de requisitos, documentación mínima y aplicable con equipos pequeños y distribuidos

El desarrollo de los proyectos de software bajo el enfoque de las metodologías tradicionales apoyadas del Proceso Unificado Rational (RUP) y el Lenguaje Unificado de Modelado, componen una arquitectura compleja que demanda documentación en esa misma proporción con la idea de dejar un producto sólido y estable, sin embargo, a finales de la década de los 90 los resultados mostraban un efecto contrario. A fin de aligerar los mecanismos del ciclo de desarrollo de un producto software cobran relevancia las metodologías ágiles (Agile Alliance, 2001), enfocándose en las relaciones interpersonales del equipo de desarrollo y los interesados para obtener información fehaciente de manera práctica con menos documentación y mayor énfasis en el valor generado hacia el cliente.

Metodologías como Scrum (Sutherland, 2017), Extreme Programming (Beck, 2004) y Kanban (Anderson, 2016) están basadas en el enfoque ágil requiere, aunque por tiempos breves un alto nivel de comunicación siendo necesario contar con equipos que trabajen juntos físicamente. Sin embargo, la nueva tendencia no genera restricción alguna respecto a su ubicación geográfica a fin de mantener equipos multidisciplinarios (Robinson, 2019) y comúnmente definidos en la literatura como “Global Software Development” (GSD).

Al respecto, las empresas hacen uso de las tecnologías de la información como código fuente distribuido, tableros en la nube, VPN, herramientas de mensajería instantánea y videoconferencias y, otras herramientas de colaboración en línea para asegurar el canal de comunicación.

Por otro lado, (Claus, Pablo, 2020) advierte que la tendencia acerca de la disposición a la lectura de informes largos, documentos con diagramas complejos técnicamente y que el usuario final no podría emitir opinión, sino hasta que vea algo “material” como por

ejemplo las funcionalidades del sistema, sin embargo, se requiere su aprobación por escrito a fin de encontrarse preparado frente a un proceso de auditoría.

En ese sentido, la carencia de documentación representa una problemática de los enfoques ágiles, sin usuarios que deseen leer documentos complejos, ni usuarios que deseen escribir, (Claus, Pablo, 2020) propone una documentación mínima que permita a los usuarios, clientes y equipo de proyecto mantener su atención en la lectura, basado en 4 pilares: Trazabilidad de los requisitos, documentación mínima, equipos pequeños y equipos con integrantes distribuidos geográficamente.

En la Tabla 4, se observa los trabajos relacionados a la materia de esta tesis analizando si atienden o no a los pilares propuestos: TR (Trazabilidad de los requisitos), EP (Equipos Pequeños), ID (Integrantes Distribuidos) y DM (Documentación Mínima).

Tabla 4

Cuadro Comparativo de los trabajos relacionados.

	Propuesta	TR	EP	ID	DM	Evidencia Empírica	P
1	InterMod (Losada & Urretavizcaya, 2013)	Si	No	Si	No	Si. Tres equipos de 7 integrantes en un mismo proyecto con 8 iteraciones.	2.50
2	Waterfall (Royce, 1970)	Si	Si	No	No	Si	2.50
3	XWebProcess (Sampaio & Vasconcelos, 2004)	Si	No	No	Si	Si. Estudiantes. 2 iteraciones de 10 días.	2.50
4	RUP Reduce Model (Monteiro & Borges, 2012)	Si	No	No	No	Si. 7 equipos de estudiantes.	2.00
5	CASDPF (Al-Maharmeh M. , A composite application software development process framework (CASDPF), 2010)	Si	No	-	No	Si. Un proyecto piloto	2.00
6	ADAPT (Vallon & Strobl, 2016)	No	No	Si	Si	Si. Proyectos de 7 equipos distribuidos, por un periodo de entre 9 y 15 meses.	2.00
7	Extreme Programming (Beck & Andres, 2004)	No	Si	No	Si	Si	2.00
8	Faat (Bernabé, Navia, & García-Peñalvo, 2015)	No	Si	-	Si	Si. Un proyecto web.	2.00
9	Kanban (Anderson & Carmichael, 2016)	No	Si	No	Si	Si	2.00
10	Distributed Scrum-based process (Dos Santos & L'Erario, 2018)	No	No	Si	Si	Si. Un Proyecto real, por 2 meses.	2.00
11	Adaptive Process Model (Rizwan & Hussain, 2008)	Si	Si	Si	No	No	2.00
12	PADRE (Rettig & Simons, 1993)	Si	Si	No	No	No	1.50
13	Scrum (Sutherland & Schwaber, 2017)	No	No	No	Si	Si	1.50
14	BRIDGE (Mandal, 2009)	Si	No	-	No	No	1.00
15	Layer-based Development Method (Lin & Yang, 2012)	No	Si	-	Si	No	1.00
16	Mobile-D (Abrahamsson, 2004)	No	No	No	Si	No	0.50
17	Proceso Súper Ágil (Cerqueira & Magalhães, 2018)	No	No	-	Si	No	0.50
Cantidad (Si)		8	7	4	10	11	
Referencias: TR: Trazabilidad de requerimientos. EP: Equipos de menos de 5 integrantes. ID: Integrantes Distribuidos. DM: Documentación mínima. P: Puntaje							

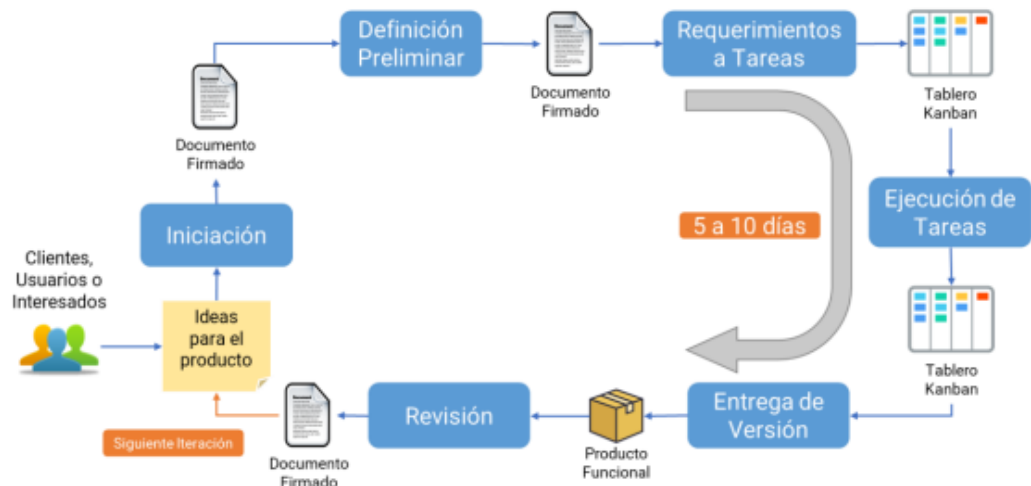
Nota. Tomado de *Proceso de desarrollo de software con trazabilidad de requisitos, documentación mínima y aplicable con equipos pequeños y distribuidos*, por Claus, 2020, Universidad Nacional de la Plata, Argentina.

En la Figura 19, se muestra la propuesta del Proceso de Desarrollo Klein, el cual define 6 etapas: Iniciación, Definición Preliminar, Requisitos a Tareas, Ejecución de Tareas, Entrega de Versión y Documento de Revisión, las cuales deben realizarse por cada iteración.

La trazabilidad en el campo de la ingeniería de software sirve como mecanismo de control para asegurar correcto desarrollo de un requerimiento. La utilidad de la trazabilidad se materializa detectando dependencias y conflictos entre los requisitos, lo cual es de gran importancia porque se detecta a tiempo la no viabilidad de algunos requisitos permitiendo tomar decisiones inmediatas para la mejora del proyecto.

Figura 19

Etapas del proceso de Desarrollo KELIN



Nota. Tomado de *Proceso de desarrollo de software con trazabilidad de requisitos, documentación mínima y aplicable con equipos pequeños y distribuidos*, por Claus, 2020.

Tabla 5

Cuadro resumen de las Etapas del proceso KLEIN.

Etapas	Objetivos	Técnicas	Herramientas	Recursos	Resultados
Iniciación	Viabilidad. Concepto global. Resultados esperados. Límites establecidos. Terceros. Riesgos.	Tormenta de ideas. Entrevistas. Observación <i>in situ</i> .	Diagramas. Aplicaciones de mensajería instantánea y videoconferencias.	Idea inicial o retroalimentación de iteraciones previas.	Documento "Iniciación" completo.
Definición Preliminar	Requerimientos. Precondiciones. Estimación. Arquitectura.	Tormenta de ideas. Entrevistas. Observación <i>in situ</i> .	Diagramas. Historias de usuario. Herramientas CASE. Aplicaciones de mensajería instantánea y videoconferencias.	Documento "Iniciación".	Documento "Definición Preliminar" completo.
Requerimientos a Tareas	Requerimientos en tareas. Responsable. Criterios de aceptación. Refinamiento de tareas.	Tormenta de ideas.	Diagramas. Historias de usuario. Herramientas CASE. Aplicaciones de mensajería instantánea y videoconferencias.	Documento "Definición Preliminar".	Tablero <i>Kanban</i> con todas las tareas a realizar en la columna "Backlog".
Ejecución de Tareas	Complejidad de tareas. Refinamiento. <i>Testing</i> .	Tormenta de ideas. Programación en pareja distribuida. Reuniones.	Diagramas. Herramientas CASE. Aplicaciones de mensajería instantánea y videoconferencias.	Tablero <i>Kanban</i> con las tareas definidas en la etapa anterior.	Nueva versión del producto listo para testear por el usuario.
Entrega de Versión	Entrega del producto. Entrenamiento de usuario. Creación de documentación de usuario.	Reuniones.	Herramientas CASE. Aplicaciones de mensajería instantánea y videoconferencias.	Producto creado en la etapa anterior y documento "Definición Preliminar".	Producto instalado en ambiente del cliente. Documentación.
Revisión	Evaluación del producto. Desvíos.	Reuniones.	Editores de texto colaborativo. Aplicaciones de mensajería instantánea y videoconferencias.	Producto evaluado por el cliente y documento "Definición Preliminar".	Documento "Revisión" completo.

Nota. Tomado de *Proceso de desarrollo de software con trazabilidad de requisitos, documentación mínima y aplicable con equipos pequeños y distribuidos*, por Claus, 2020, Universidad Nacional de la Plata, Argentina.

En la Tabla 5, se presenta un resumen de cada una de las etapas con sus objetivos, técnicas, Herramientas, Recursos y Resultados que forman parte de la propuesta de KLEIN.

Este trabajo enfatizó la deficiencia documental que generan las metodologías ágiles y que la que generan las metodologías tradicionales resultan ser inmanejables. Además, analizó el impacto social de los hábitos de lectura de las personas y su relación con la documentación de los proyectos de desarrollo de software.

Finalmente, como solución a ello (Claus, Pablo, 2020) estableció un proceso de desarrollo de software que cubriera los pilares que definió como determinantes: trazabilidad de requisitos, documentación mínima, aplicable con equipos pequeños, distribuidos.

2.2.1.10 *Selección de técnicas de obtención de requisitos de software: Método para la gestión del alcance del proyecto*

La Gestión del alcance del Proyecto es una de las 10 áreas del conocimiento definidas en el marco de la Gestión de Proyectos y descritas en la Guía de los Fundamentos para la Dirección de Proyectos publicada por el Project Management Institute (PMI) y su versión vigente es la número 6.

La Gestión del alcance del Proyecto está compuesta por 6 procesos: 4 en el grupo de procesos de Planificación y 2 en el grupo de procesos de Monitorización y Control (PMI-PMBOK, 2017).

Tabla 6

Cuadro de Relación entre la Gestión del Alcance y Grupo de Procesos.

ÁREAS DE CONOCIMIENTO	GRUPOS DE PROCESOS				
	INICIO (2)	PLANIFICACIÓN (24)	EJECUCIÓN (10)	MONITORIZACIÓN Y CONTROL (12)	CIERRE (1)
4. Gestión de la Integración (7)	4.1 Desarrollar Acta Constitución del Proyecto	4.2 Desarrollar Plan de Dirección del Proyecto	4.3 Dirigir y Gestionar el Trabajo del Proyecto 4.4 Gestionar el conocimiento del Proyecto	4.5 Monitorear y Controlar el Trabajo del Proyecto 4.6 Realizar Control Integrado de Cambios	4.7 Cerrar Fase o Proyecto
5. Gestión del Alcance (6)		5.1 Planificar Gestión de Alcance 5.2 Recopilar Requisitos 5.3 Definir Alcance 5.4 Crear la EDT		5.5 Validar Alcance 5.6 Controlar Alcance	

Nota. Adaptado de *A Guide to the Project Management Body of Knowledge*, por Project Management Institute, 2017.

La Tabla 6 muestra los procesos del área de conocimiento Gestión del Alcance y su relación con el Grupo de Procesos de Planificación en el cual encontramos: 1) Planificar la Gestión del Alcance, 2) **Recopilar Requisitos**, 3) Definir Alcance y 4) Crear la EDT y con el Grupo de Procesos de Monitorización y Control: 1) Validar el Alcance y 2) Controlar el Alcance.

El proceso de Recopilar Requisitos tiene entradas, las herramientas y técnicas, y las salidas según se visualiza en la Figura 20.

Figura 20

Guía propuesta para el registro de atributos asociados a la vista de propiedades del flujo de datos.



Nota. Tomado de *A Guide to the Project Management Body of Knowledge*, por Project Management Institute, 2017.

Una adecuada recopilación de Requisitos establecerá de manera unívoca el enunciado que define la magnitud del producto a desarrollar, razón por la cual (Gobova y Huchenko, 2021) realizaron el artículo en desarrollo, con la finalidad de analizar a través de una encuesta las prácticas actuales de uso de técnicas de elicitación de requisitos en proyectos de desarrollo de software y construcción de modelos de predicción basados en algoritmos de aprendizaje automático, que pueda apoyar a los expertos en las actividades de técnicas de selección a evaluar la precisión de estos modelos para técnicas de elicitación de

obtención de requisitos, selección y orientación sobre su uso bajo el enfoque del modelo de aprendizaje automático para la selección de técnicas y modelado.

El estudio se dividió en 3 fases de acuerdo con la Figura 21 y la siguiente descripción:

- 1) Diseño del cuestionario y la recopilación de datos: Compuesto por un total de 43 preguntas que se elaboraron utilizando vocabulario de análisis empresarial en su mayoría de BABOK (Business Analysis Body of Knowledge), con preguntas tipo abiertas y cerradas (opción individual y múltiple), luego de ello expertos en análisis de negocios del Industria de TI realizaron observaciones y propuestas de mejora.
- 2) Método basado en el aprendizaje para la selección de técnicas de elicitación y justificación de los resultados obtenidos: Se aplicó MS Azure Machine Learning Studio (Servicio de aprendizaje automático) para la obtención de recomendaciones según la combinación de factores.
- 3) Aplicación del modelo de aprendizaje automático

La aplicación de Maching Learning delimitó el estudio de 4 técnicas de elicitación las cuales son: 1) Lista de partes interesadas, mapa o personas, 2) Creación de prototipos, 3) análisis de reglas comerciales y 4) Observación. Se consideraron las siguientes métricas para la precisión del modelo de evaluación.

Exactitud: Es una métrica que mide el mejor modelo de clasificación en proporción de resultados verdaderos al total de casos.

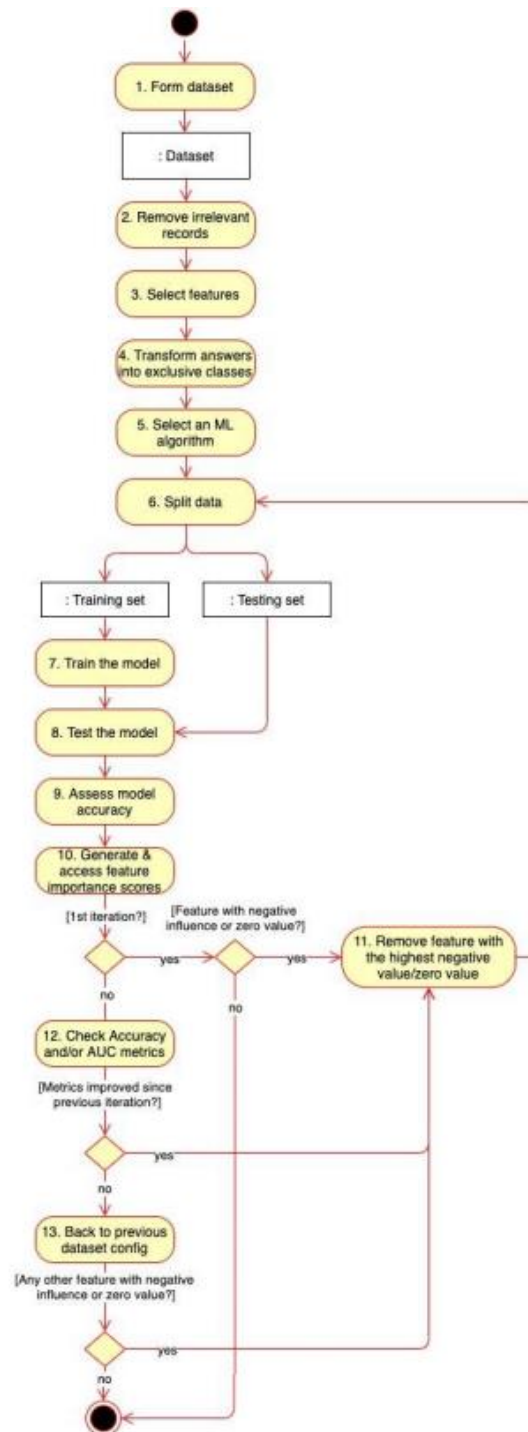
AUC: Métrica que mide el área bajo la curva trazada con verdaderos positivos en el eje y, y falsos positivos en el eje x. Proporciona un único número que nos permite comparar los modelos de diferentes tipos.

Precisión: Es una métrica que mide la proporción de resultados verdaderos en general, resultados positivos.

Re-llamadas: Una métrica que mide la fracción de todos los resultados correctos devueltos por el modelo.

Figura 21

Algoritmo basado en aprendizaje automático para la selección de técnicas de elicitación.



Nota. Tomado de Selección de técnicas de obtención de requisitos de software: Método para la gestión del alcance del proyecto, por Gobova y Huchenko, 2021.pp 6

Tabla 7

Cuadro que muestra la Exactitud de los valores de las métricas.

Techniques	Metric for Two Class Decision Jungle algorithm				
	Accuracy	AUC	Precision	Recall	Threshold
Stakeholder list, map or personas	0.738	0.734	0.714	0.435	0.5
Prototyping	0.778	0.802	0.731	1	0.52
Business Rule Analysis	0.741	0.764	0.829	0.708	0.57
Observation	0.716	0.76	0.605	0.742	0.44

Nota. Tomado de *Selección de técnicas de obtención de requisitos de software: Método para la gestión del alcance del proyecto*, por Gobova y Huchenko, 2021, pág.8

En la Tabla 7 se definen las características más importantes por cada técnica de elicitación de forma independiente, el grado de importancia fue calculado mediante el algoritmo Permutation Feature Importance (PFI) cuya puntuación es indicativa de cuánto el modelo depende de la función. Según el resultado, se visualiza en la Figura 22 que la "categoría de proyecto" se define como una característica importante para todas las técnicas de elicitación excepto para Análisis de reglas de negocio. Por otro lado, las características como "Análisis comparativo y de mercado", "Minería de datos", "Observación", "Rol principal del proyecto", "Clase de sistema" y "Taller y grupos focales" (Gobova y Huchenko, 2021, pág.8), son definidos como una característica importante para dos de las cuatro técnicas de elicitación.

Figura 22

Resultados de la puntuación obtenida en la ejecución del Modelo de Machine Learning

Stakeholder list, map or personas	Prototyping	Business Rule Analysis	Observation
Benchmarking and Market Analysis	Document analysis	Experience	Benchmarking and Market Analysis
Company size	Elicitation responsibility	Process analysis	Data mining
Data mining	Experience	Project category	Design thinking
Industry sector	Multiple project roles		Document analysis
Observations	Observations		Multiple project roles
Primary project role	Stakeholder list, map or personas		Primary project role
Project category	System class		Process analysis
Reuse database	Ways of working		Project category
Secondary project role	Workshops and Focus groups		Survey/Questionnaire
System class			Workshops and Focus groups

Nota. Tomado de *Selección de técnicas de obtención de requisitos de software: Método para la gestión del alcance del proyecto*, por Gobova y Huchenko, 2021, pág.9

La Gestión del Alcance depende de la diligencia con la que se realice la recolección de los requisitos a través de las técnicas de obtención adecuadas, en ese sentido, los resultados del estudio de encuesta por parte de (Gobova y Huchenko, 2021) demostraron que es posible predecir / recomendar, con un alto nivel de precisión, el uso de las siguientes técnicas de obtención según el contexto del proyecto y otras técnicas seleccionadas: lista de partes interesadas, mapa o personas, creación de prototipos, reglas comerciales análisis y observación.

2.2.2 Métodos, modelos, técnicas

2.2.2.1 *Método para la representación semi-automática de modelos conceptuales desde documentos de negocio escritos en lenguaje natural en español*

La obtención de información y su representación gráfica para establecer las condiciones funcionales del sistema a desarrollar, debe considerar los activos existentes (documentos de negocio u otras fuentes) que aporten información útil, incluyendo la comunicación verbal que se adquiere mediante la técnica de la entrevista. Sin embargo, la extracción de la información para su posterior procesamiento resulta compleja debido a no encontrarse estructurada (Marín-Álvarez, Manrique-Losada y Quintero, 2020) y el inminente crecimiento del negocio.

En ese sentido, el uso de Procesamiento de Lenguaje Natural (NLP por sus siglas en inglés) para atender las diferentes fuentes de datos y su volumen resulta útil, complementado con técnicas que permitan su similitud al procesamiento de lenguaje del ser humano en tareas como “leer y entender la información contenida en los textos”. Al respecto, la revisión de la literatura realizada por (Marín-Álvarez et al., 2020), evidencia que los enfoques para la obtención y representación del conocimiento se apoya de los métodos NLP, entre los que más destaca es el método Part of Speech Tagging o PoS Tagging para luego ser representado a través de definiciones soportadas en el conocimiento previo, o mediante modelos UML.

Los beneficios del uso de técnicas NLP, impactan sobre el tiempo empleado por parte de los ingenieros de requisitos para leer y comprender la gran cantidad de información en forma de texto con lenguaje natural con los que cuenta la empresa reduciendo además las ambigüedades generadas por las restricciones y contexto del léxico.

La propuesta de (Marín-Álvarez et al., 2020), se establece sobre el tratamiento de los documentos de negocio escritos en lenguaje natural en idioma español, basado en la información de la Tabla 8 que lista once (11) trabajos de investigación, en la cual se evidencia que la técnica de extracción principal es la del PoS Tagging y es incorporada como parte de su solución, sin embargo, los Modelos UML en los estudios realizados se han utilizado para representar información extraída en idioma inglés, en ese sentido y de acuerdo al análisis de (Ozkaya,M, 2016) determina que los modelos UML son los más

populares porque “son bastante populares gracias a su baja curva de aprendizaje y sus notaciones visuales”, razón por la utiliza el modelo UML para el modelamiento.

Tabla 8

Trabajos realizados para representar modelos conceptuales desde documentos del negocio

Referencia	Idioma del texto procesado	Técnica de extracción principal	Modelo presentado
Automatic information extraction from texts with inference and linguistic knowledge acquisition rules. (2013)	Portugués	Parsing	Ontología
A supervised method to enhance vocabulary with the creation of domain specific lexica. (2013)	Portugués	Herramienta de mapeo	Ontología
Natural language requirements quality analysis based on business domain models. (2013)	Inglés	Parsing	Ontología
A NLP Based Framework to Support Document Verification-as-a-Service. (2016)	Inglés	Machine Learning	Ontología
Software Requirements as an Application Domain for Natural Language Processing. (2017)	Inglés	PoS Tagging	Ontología
Conceptual modeling of natural language functional requirements. (2014)	Inglés	PoS Tagging	Modelo UML
Generating UML activity diagram from SBVR rules. (2016)	Inglés	PoS Tagging	Modelo UML
OCL usability: a major challenge in adopting UML. (2014)	Inglés	PoS Tagging	Modelo UML
Enforcing structure on textual use cases via annotation models. (2014)	Inglés	PoS Tagging	Modelo propio
Automated extraction and validation of security policies from natural-language documents. (2012)	Inglés	PoS Tagging	Lenguaje etiquetado
Natural Language Processing of Requirements for Model-Based Product Design with ENOVIA / CATIA V6. (2015)	Inglés	PoS Tagging	Lenguaje etiquetado

Nota. Adaptado de *Método para la representación semi-automática de modelos conceptuales desde documentos de negocio escritos en lenguaje natural en español*, por Marín-Alvarez, Manrique-Losada y Quintero. 2020, pág.569.

El marco metodológico a partir del cual plantearon su propuesta (Marín-Álvarez, et al., 2020) para desarrollar proyectos de analítica de texto desde documentos de requisitos, es el implementado por (Manrique, Quintero, Marín y Morales, 2018) en el cual consideran a los documentos de negocio, documento de requisitos e historias de usuario como documentos de requisitos y principal insumo a procesar para finalmente obtener como resultado documento de contexto del proyecto, el corpus, el modelo, el análisis de resultados y el informe final del proyecto. Asimismo, se propone 3 tipos de herramientas automáticas: 1) Preparador (pre-procesador) que a partir del corpus inicial realiza la transformación, esta herramienta de software puede “compilar documentos y/o entrenar de NLP o Text Mining”; 2) Compilador (procesador) realiza un análisis sintáctico, gramatical, semántico con la finalidad de extraer los conceptos más importantes para el negocio, obteniendo como resultado una especificación en lenguaje controlado o llamada también ontología, las herramientas de este tipo son capaces de procesar “lenguaje natural o Text Mining, como Freeling, Stanford NLP y otras, las cuales permiten ejecutar procesos como POS Tagging y Lemmatization”, y 3) Modelador: Herramientas que reciben los conceptos de la especificación y los representan en modelos gráficos más estructurados como Enterprise Architect y/o Ed Graph Editor.

Asimismo, define los 3 roles implicados: 1) Director del proyecto, quien define los objetivos y el alcance contextualizándolos en un documento; 2) Documentador, quien se encarga de las actividades centrales que permiten generar el modelo a partir de los documentos, y 3) implementador, encargado de verificar que el modelo corresponda a las necesidades de los desarrollos y, realizar el seguimiento del uso de los modelos resultantes.

El Método de extracción propuesto desarrolla 5 fases: Contextualización, Recolección, Preparación, Modelamiento y Evaluación el cual se visualiza en la Figura X, las cuales se describen a continuación:

Contextualización: Fase en la que se define el problema e hipótesis para su posterior representación en un modelo gráfico.

Recolección: Fase en la que se compila los documentos del negocio para obtener las “palabras clave” que permitirán definir un dominio en particular que compone el “corpus inicial” y, además se identifican los documentos que se usarán en la fase de evaluación.

Preparación: Fase en la que implica 2 acciones: la primera permite determinar las herramientas más idóneas para la implementación, luego de ello al “corpus inicial” generado en la fase anterior se asigna un nivel de importancia a los términos de negocio, donde el 1 es menos importante y 10 es más importante esta clasificación es conocida como Information content (IC) que mide el grado de importancia de un término en el “corpus”, esta labor es ejecutada con el apoyo de un analista experto en el dominio. En este caso, como artefacto de output se tiene la definición de pesos de negocio para el dominio específico Information content (IC), Reglas de negocio y el texto depurado del documento.

Modelamiento: Fase en la que se ejecuta el procesamiento del texto a través de 2 acciones:

- 1) Generar POS Tags: Mediante herramientas de clasificación como Freeling, se obtiene una clasificación de partes de oración de los términos del texto extraídos, lo cual se representa en un documento XML;
- 2) Procesar POSTags: El XML resultante es analizado de modo que se transforme a elementos de un modelo conceptual, que represente conceptos y relaciones, y
- 3) Generar salida gráfica: Los conceptos a mostrar y sus relaciones identificados en el paso anterior, se pasan en forma automática a un formato gráfico. Para el desarrollo de estas 3 acciones anteriormente descritas se hace uso de las reglas de mapeo definidas en la Tabla 9.

Tabla 9

Reglas de mapeo Basadas en la propiedad POS en el XML entre Elemento en Modelo, Elemento en Texto

Elemento en Modelo	Elemento en Texto	Nota
Concepto	noun	Basadas en la propiedad POS en el XML
Relación	pos = adjective, pos = adposition, pos = verb	Patrones que se identifican en el XML. Se definen propiedad y valor del XML. Patrón “verboconjugado” genera una acción codificada en la herramienta.
	pos = verb, pos = adposition	
	pos = adposition, pos = verb	
	pos = adposition	
	form = que	
	verboconjugado	

Nota. Tomado de *Método para la representación semi-automática de modelos conceptuales desde documentos de negocio escritos en lenguaje natural en español*, por Marín-Álvarez, Manrique-Losada y Quintero. 2020, pág.574.

Evaluación: En esta fase se estableció el valor de las 2 variables independientes definidas: Comparativo del tiempo que toma procesar y Comparativo del nivel de interpretación de la información contenida en los documentos, a fin de establecer la validez de la propuesta. En ese sentido, Marín-Álvarez, et al. (2020) luego de ejecutar el caso un caso de estudio sus resultados concluyeron que “se evidenció claramente que el tiempo para procesamiento usado por el grupo de control es superior al tiempo tomado por el grupo con la aplicación del método” y, “los analistas interpretan previamente el modelo, antes de proceder a responder las preguntas, proceso que resultó más rápido”, en ese sentido, los autores aclaran que para el proceso de búsqueda de conceptos en el modelo aún se realiza de forma manual pero que en muchos casos resultaba intuitiva.

2.2.3 Software existente

2.2.3.1 *CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información*

La propuesta MDA se organiza básicamente en torno a: los modelos independientes de computación (CIM, Computation Independent Model), modelos independientes de la plataforma (PIM, platform independent models) y modelos específicos de la plataforma (PSM, platform specific models).

A través de los modelos CIM, se grafican la conceptualización del dominio problema; los modelos PIM precisan la arquitectura y funcionalidad del sistema sin considerar aspectos tecnológicos de la plataforma sobre la que se desarrolla; siendo los PSM los que contemplan de manera integrar el soporte tecnológico. En ese sentido, en el DSDM, se determina la correspondencia entre los modelos en mención (Peña-Moreno 2011 y Jiménez, 2012).

“Con este framework el Object Management Group (OMG) demuestra que el enfoque de modelado es una forma potente de especificar sistemas, permitiendo que se obtengan beneficios importantes en aspectos fundamentales como son la productividad, la portabilidad, la interoperabilidad y el mantenimiento” (Moreno-Rodríguez y Rosendo, 2020, pág. 3).

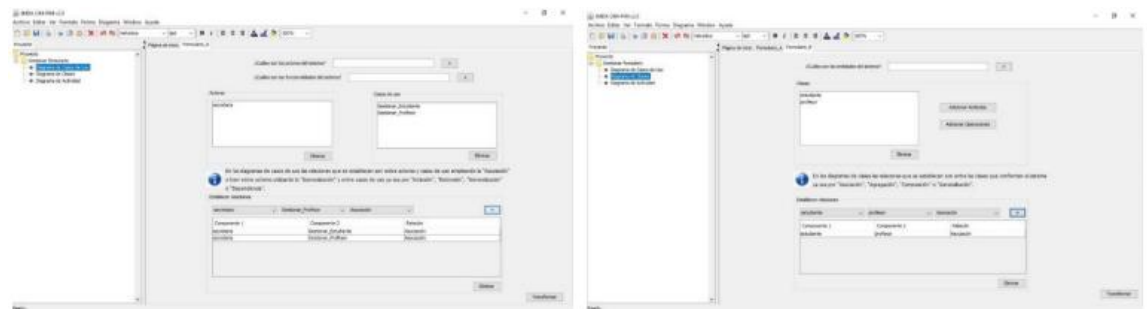
jMDA es una herramienta CASE basada en el framework Arquitectura Dirigida por Modelos (MDA), con 3 módulos a disposición:

Módulo CIM-PIM

En el primer paso se visualiza un grupo de preguntas – respuestas a través de un formulario, según se muestra en la Figura 23, a partir del cual se generará las Figuras

Figura 23

Formularios de preguntas-respuestas previo a la generación del Diagrama de Casos de Uso y el Diagrama de Clases.



Nota. Tomado de CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información, por Moreno. 2020.

Una vez que los formularios sean respondidos por el analista pulsará en el botón “Transformar”, generando automáticamente el cambio del modelo CIM al PIM, teniendo como resultado los diagramas de casos de uso y de clases obtenidos al realizar la transformación correspondiente, según la Figura 24.

Figura 24

Diagrama de Casos de Uso y el Diagrama de Clases generados



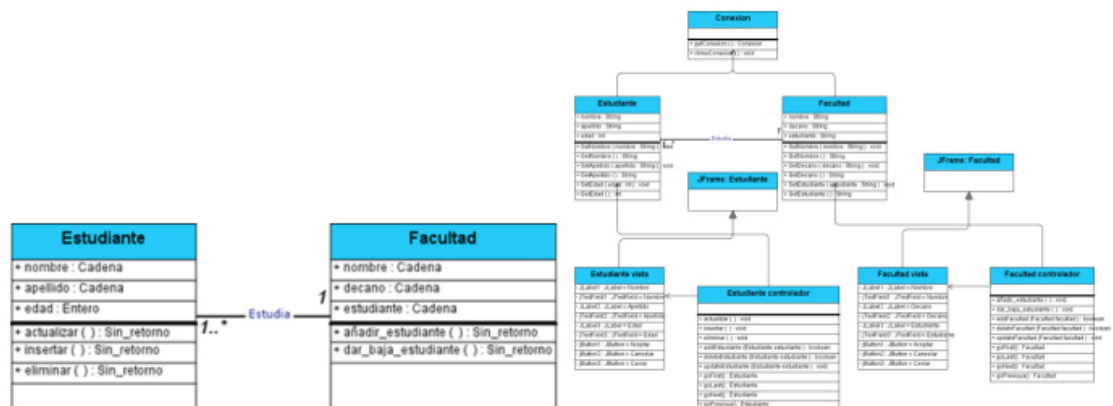
Nota. Tomado de CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información, por Moreno. 2020.

Módulo PIM- PSM

En este módulo los posibles diagramas a generar son los diagramas de casos de uso, clases, actividades, secuencia, y estados, para un contexto para el desarrollo de Sistemas de Información con Bases de Datos relacionales, que pueden ser generados de la información del módulo anterior o generarse de forma independiente.

Figura 25

Transformación PIM a PSM



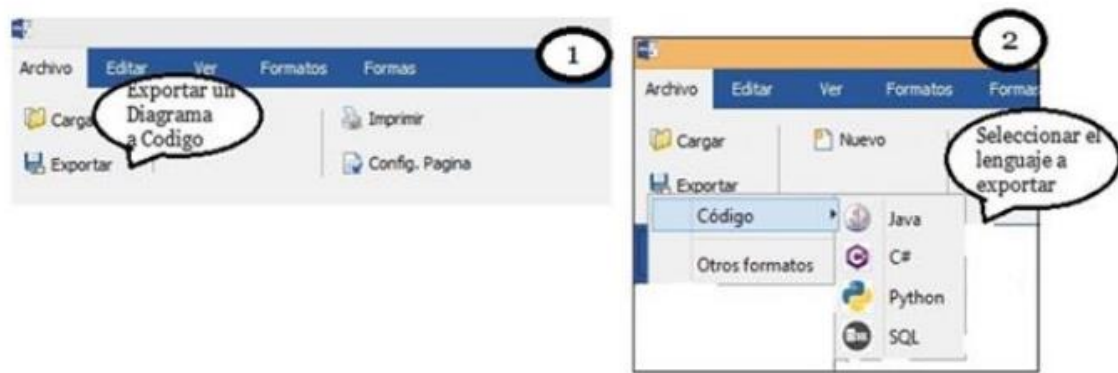
Nota. Tomado de CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información, por Moreno. 2020.

Módulo PSM-Código

Este módulo tiene como base reglas de transformación definidas, para los lenguajes Java, Python y C#, además, este módulo cuenta con un algoritmo de transformación que incluye 4 plantillas de programas (para Java, Python, C# y SQL) y 13 reglas con subreglas, según se visualiza en la Figura 26.

Figura 26

Transformación PIM a Código



Nota. Tomado de CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información, por Moreno. 2020.

2.3 Bases Teóricas

2.3.1 Ingeniería de Requisitos

La ingeniería de Requisitos comprende dos principales procesos: 1). -Desarrollo de Requisitos: captura, análisis, especificación y validación. 2). -Gestión de Requisitos: trazabilidad de los requisitos (Bourque y Fairley, 2014; Ramírez et al., 2017). En el primer proceso se debe identificar los requisitos que tendrá el sistema a desarrollar, esto se logra comunicándonos con las partes interesadas (clientes y usuarios del sistema), para obtener

información sobre el dominio de la aplicación y los servicios que el sistema debe proporcionar, el rendimiento del sistema requerido, las limitaciones de hardware, etcétera. El objetivo es desarrollar el producto correcto en el primer intento. Cuando se reducen los errores en las primeras etapas del proyecto su mitigación es con mayor intensidad y más sencilla la reanudación. El método de ensayo y error, además de ser más caro y lento, crea insatisfacción en el cliente por lo que se busca evolucionar en la calidad del trabajo en la gestión de requisitos para lograr un impacto positivo en el costo, el tiempo y la satisfacción del proyecto.

La elicitación (del latín *elicitus*, "inducido" y *elicere*, "atrapar") es un término asociado a la psicología que se refiere al traspaso de información de forma fluida de un ser humano a otro por medio del lenguaje (Elicitar, 2019).

En este contexto, el proceso de elicitación de requisitos, incluye actividades como descubrir e interpretar adecuadamente los requisitos del usuario para desarrollar un sistema, razón por la cual la comunicación con las partes interesadas desde el inicio del proyecto (Gobova y Huchenko, 2021), a fin de obtener un producto que cubra las necesidades reales del mismo, así por ejemplo identificamos y clasificamos tempranamente a usuarios y sus necesidades las cuáles pueden encajar en un software empaquetado o a medida.

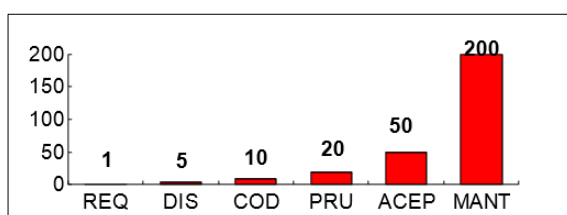
En esta fase es importante invertir gran parte de esfuerzo para afinar y encontrar la correlación entre lo que quieren los stakeholders, y su viabilidad a través de funcionalidades que debe cubrir el software y finalmente satisfacer al ente inicial que ocasionó este proceso, es decir el usuario.

La elicitación de requisitos establece primigeniamente actividades como la identificación de los interesados, la elicitación de los requisitos funcionales, la identificación de restricciones, el establecimiento de los escenarios y la identificación de los requisitos de calidad (ISO/IEC, 2007).

Al respecto, la elicitación de requisitos es una de las fases más importantes debido a que representa la base inicial del software que se va a desarrollar y con ellos perfil a su naturaleza.

Figura 27

Impacto en la fase de Mantenimiento ocasionado por la reparación de un error no identificado tempranamente en la Elicitación de Requisitos.



Nota. Tomado de *Requirements Engineering A good practice guide*, por Sommerville y Sawyer, 1997.

En la Figura 27, deducimos la importancia de la Elicitación de Requisitos, ya que mientras más precisos son, los costos se reducen y por ende el nivel de satisfacción del usuario es mayor varios estudios han replicado estos resultados (Boehm, 1987; Serna y Serna ,2020; Sommerville y Sawyer , 1997).

Los requisitos se clasifican en funcionales y no funcionales o también llamados requisitos de calidad, los cuales representan las restricciones que deben de cumplir los requisitos funcionales para la entrega de un software.

La calidad del producto software se puede interpretar como el nivel en que dicho producto satisface los requisitos de sus usuarios entregando valor al negocio. Son precisamente estos requisitos los que se encuentran representados en el modelo de calidad ISO/IEC 25030, el cual categoriza la calidad del producto en características y sub-características (ISO/IEC, 2007), las ocho características de calidad son: Eficiencia de desempeño, Adecuación Funcional, Compatibilidad, Usabilidad, Fiabilidad, Seguridad, Mantenibilidad y Portabilidad.

Por ejemplo, un error no detectado por el equipo, y encontrado por los usuarios finales tras la entrega se conoce como “defecto evadido” (escaped defect). El número de defectos evadidos es una métrica importante para determinar la calidad del trabajo (Agile Testing Metrics, 2018).

El segundo proceso, surge como respuesta a los cambios que se presentan o solicitan en relación con la definición de requisitos de un sistema, estas solicitudes deben ser administradas a través de mecanismos implementados desde el momento que se empiezan a documentar los requisitos y, se evalúa su relación con los demás, el impacto en costo y tiempo, asimismo, se registran y monitorean los cambios o mejoras propuestas (Ramírez et al., 2017).

2.3.2 Framework i*

En 1995, Yu en su tesis doctoral desarrolló un marco conceptual más versátil para modelar procesos que involucre a múltiples participantes, incluidos humanos y computadoras, para que uno pueda tener una reingeniería de procesos más sistemática. Esto incluye identificar un conjunto de características de modelado para describir procesos y los fundamentos detrás de ellos, proporcionando construcciones representativas formales y su semántica, identificando algunas capacidades analíticas y delineando un marco de diseño; y su aplicación en cuatro áreas: Ingeniería de requisitos, Reingeniería de Procesos de Negocio, Análisis de Impactos Organizacionales y Modelado de Procesos de Software (Yu, 1995). El marco i* y sus extensiones metodológicas se han utilizado en una amplia gama de dominios de aplicaciones, como el modelado de negocios, desarrollo de software orientado a objetos, obtención de requisitos de software, orientado a agentes desarrollo de software, modelado y análisis de requisitos no funcionales, seguridad requisitos, requisitos de confianza y privacidad, y más. En todas estas aplicaciones, los conceptos de i* se han utilizado para capturar los elementos sociales e intencionales de cada dominio específico, apoyando así el desarrollo de software (Estrada, Martínez, Pastor y Mylopoulos, 2006). Estando en la fase inicial del desarrollo del producto, debemos poner todos los esfuerzos en elaborar modelos/notaciones gráficas con un mayor nivel de abstracción para que puedan utilizarse en la comunicación con el cliente y que representen el comportamiento de sus procesos y los supuestos en los que se basa con la finalidad de asegurar un mayor nivel de satisfacción; así, también reducir posibilidades de interpretaciones inexactas de los requisitos que plantean, porque en ocasiones escapa de nuestro conocimiento habitual.

En el marco conceptual i* establece 2 modelos, Modelo de Dependencia Estratégica (SD por sus siglas en inglés), en el cual se representan las relaciones y dependencias entre actores en un contexto organizacional cuyo uso estaría más aplicado a la representación gráfica de procesos de negocio y, el Modelo de Justificación de Dependencia (SR por sus siglas en inglés), representa el comportamiento del actor en el negocio mediante objetivos y tareas se considera que este modelo aplica a un contexto de análisis de requisitos utilizando los objetivos del negocio, los cuales expresan qué esperan los interesados del sistema, y el ingeniero de requisitos debe traducirlo en tareas que finalmente representan las maneras de alcanzarlos.

(González-Baixaui et al., 2004), aseveran que es más difícil que cambien las razones por las que se necesita un sistema, que la forma en que se realizan sus funciones. Puesto que las formas pueden regirse al contexto u envergadura en la que se desarrollará un software. El uso de modelos gráficos en la fase de Elicitación de Requisitos, se convierten en una potente herramienta para representar las expresiones de necesidad del usuario que permite identificar primigeniamente una visión más real del sistema a construir.

2.3.3 Historia de Usuario

En marzo del 2001, se firmó un documento llamado Manifiesto Ágil en consenso a una publicación anterior de Kent Beck “Extreme Programming Explained: Embrace Change” y como alternativa a las metodologías tradicionales cuyo enfoque se basa en una estructurada cantidad de documentos para la gestión del producto de software. En ese sentido, el manifiesto ágil, establece un conjunto de 12 principios los cuales aportan la base para responder de manera flexible y colaborativa a la identificación adecuada de necesidades del usuario.

“Nuestra mayor prioridad es satisfacer al cliente mediante la entrega temprana y continua de software con valor”, “Los proyectos se desarrollan entorno a individuos motivados”;

son algunos de los principios propuestos que destacan la importancia de la participación del usuario en la construcción de un producto de software y sus requisitos; este análisis funcional describe sus requisitos y que por lo general utilizamos el UML, al respecto del enfoque ágil, la historia de usuario se convierte en la herramienta que permite sintetizar de manera inteligible los requisitos.

El origen de la historia de usuario está ligado a la metodología ágil, y se encuentra en el sector del desarrollo de software, la acuciosidad de sus procesos resulta de provecho para el cliente en un mercado dinámico y creciente (Manifesto for Agile Software Development, 2001).

En la construcción de una Historia de Usuario, se establece intervenciones cortas que narran funcionalmente las acciones en primera persona del usuario que es el que ejecuta algún rol en particular.

Los procesos de desarrollo ágil tienen una perspectiva diferente en comparación con los procesos de desarrollo tradicionales que siguen un modelo más lineal o en cascada para realizar tareas. Una de las diferencias es que puede faltar una especificación de requisitos detallada durante una gran parte del proyecto o incluso durante toda la duración del proyecto. Algunas otras diferencias incluyen el uso de historias como fuente de requisitos (Qasaimeh, 2011), su función principal es identificar problemas percibidos durante las entrevistas, proponer soluciones y estimar el esfuerzo que requieren implementar las ideas propuestas por los usuarios e interpretados por el equipo de proyecto.

Las condiciones para validar una historia de usuario marcan la pauta y ventaja estricta del cumplimiento o no del requerimiento solicitado, por el contrario, para el entendimiento en el modelamiento UML resulta necesario consultar las matrices de seguimiento que muchas veces no se encuentran actualizadas y son engorrosas.

Las condiciones de validez que se incluyen en la Historia de Usuario se constituyen en las pruebas que debe superarse para ser clasificada como terminada. El nivel que determina su condición es dicotómico: “Hecha” o “No hecha”, en ese sentido el margen de clasificación es claro y se sabe cuándo están realmente terminadas.

Teeravarunyou, Sakol, Sato, Keiichi, 2001, destacan el entendimiento del usuario, suscitando el reconocimiento pertinente de las necesidades. Asimismo, el diseñador adquiere conocimiento que consolida la especificación adecuada de un requisito.

Para alcanzar el grado de cumplimiento satisfactorio del software desarrollado, este debe adaptarse a las necesidades propias de la organización, para ello se debe utilizar el conocimiento del usuario de forma natural porque permitirá abordar de manera acertada sus requisitos.

Para que las Historias de Usuario alcancen un eficiente nivel de productividad, se requiere de un equipo con capacidades técnicas, funcionales, interpersonales de alto nivel e implica a equipos multidisciplinarios trabajando siempre de forma colaborativa en el tiempo.

Es en este contexto donde la dinámica de la entrega continua en las metodologías ágiles podría intimidar a los involucrados y ver en esto una suerte de control, por el contrario, se convierte en beneficio porque el equipo aporta intensamente durante el proceso y este involucramiento hace que el producto sea más refinado.

2.3.4 Trazabilidad

La trazabilidad es la técnica que establece la ruta seguida por un requerimiento hasta ser materializado y, que producto de la necesidad de mantenibilidad o actualización se puede identificar el punto de partida, es decir gestionar el cambio de los requisitos (Fangfei, 2019). El éxito de este proceso constituye un factor de calidad (IEEE 830,1998).

En la tesis doctoral de (Jiménez, 2012), constituye la gestión de la trazabilidad como recurso idóneo para establecer el nivel de satisfacción del usuario, ya que la verificación de cada requisito evidenciará su cumplimiento.

En un enfoque ágil, la trazabilidad de requisitos no se puede realizar ya que se establece en la gestión de requisitos tradicional, como lo exigen ISO / IEC 12207: 2008 o SWEBOK, porque no se considera en el proceso un documento que soporte detalles tan granulares como la especificación de requisitos del sistema / usuario – SRS que la metodología tradicional obliga a definir.

La ausencia de esta especificación, o su inadecuada gestión puede crear serios problemas a otros procesos, como la gestión de cambios, el análisis de impacto y la estimación, debido a que la base inicial del proceso se alimenta de lo definido/establecido en un Documento de Especificación de Requisitos (SRS). Las matrices es uno de los activos que utilizan esta información, sin embargo, no en la profundidad que se espera, e inclusive su modificación/actualización de esta documentación resulta siendo engorrosa, ubicar la

línea de afectación y que es transversal no sólo en uno sino en varios documentos, termina por perderse el rastro, siendo ilegible para el equipo y para cuando se requiere un mantenimiento del sistema, simplemente nos topamos con las consecuencias de estas acciones.

Aunque los métodos ágiles no consideran desarrollar un documento de especificación de requisitos formales, es necesario definir en cierta medida un artefacto que nos provea el sustento en miras del desarrollo de un proyecto exitoso (Espinoza y Garbajosa, 2011). Toda información generada desde que se origina el requisito es valiosa y aporta en gran medida a definir su nivel de calidad, mientras más preciso el requisito, más certera la solución, en consecuencia, el usuario tendrá un sistema que le garantice su adecuado funcionamiento.

El registro manual de la información de rastreo causaría una carga de trabajo adicional para las personas que realizan el proceso y, por lo tanto, haría que la trazabilidad sea costosa y requiera mucho trabajo. Asimismo, las huellas grabadas manualmente a menudo son subjetivas e idealizadas, es decir, las huellas se adaptan a ciertas expectativas, para cumplir con los planes predefinidos, por lo tanto, las huellas grabadas manualmente a menudo no reflejan la realidad. (Jaqueira et al., 2013) indica que la captura de trazas totalmente automatizada es imposible debido a la gran cantidad de información informal utilizada durante la ingeniería de requisitos, actas de reuniones o notas de entrevistas, sin embargo, actualmente, trabajos de implementación a través del uso de herramientas (Enterprise Project Management, JIRA, Team Foundation Server, Quality Center, Enterprise Architect, SVN y Eclipse) han incorporado la trazabilidad en cada una de las fases de producción del software para Gestionar el cambio y mejorar la calidad del producto (Akman, Özman, Aydın y Göktürk, 2016).

2.4 Hipótesis general

La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad mejora la fase de captura de requisitos en el proceso de desarrollo de software.

2.5 Hipótesis específicas

HE 1: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad reduce el tiempo de transformación de requisitos a funcionalidades.

HE 2: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad incrementa la calidad del requisito.

2.6 Identificación de variables

Variable independiente:

La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad.

Variables dependientes:

1. *Eficiencia del proceso de transformación de requisitos en funcionalidades*, es decir realizar una tarea en un tiempo mínimo con un mínimo de esfuerzo.
2. *Calidad del requisito* está definida por un conjunto de características en (IEEE830, 1998) y que (Davis et al., 1993) las encapsulan en las llamadas métricas de la calidad de la especificación para materializarlas en valores porcentuales como son:
 - 2.1 *Especificidad del requerimiento*: Valor que representa una única interpretación para cada requerimiento por parte de los revisores.
 - 2.2 *Compleitud del requerimiento*: No hay omisiones que comprometan la integridad de los requisitos que representan las necesidades del usuario.
 - 2.3 *Validez del requerimiento*: Establece la viabilidad de lo propuesto por el cliente y lo que el equipo de desarrollo puede realizar. El grado de validación representa la corrección en la definición de los requisitos.

2.7 Operacionalización de variables

La especificación del término “operacional” nos señala los procesos o procedimientos que debemos realizar para entender el concepto teórico de nuestra hipótesis (Vara-Horna, 2012), para los indicadores: Reducción del tiempo, Especificidad, Compleitud y Validez en requisitos se definen las siguientes operaciones para medirlos.

La operacionalización de la métrica de la especificidad del requerimiento es

$$ER = \frac{Nui}{Nr}$$

Donde:

ER, es el grado de especificidad de los requisitos.

Nui, es el número de requisitos con una sola interpretación por parte de los revisores.

Nr, es el número total de requisitos funcionales y no funcionales.

La operacionalización de la métrica de la validez del requerimiento

$$RV = \frac{Nc}{Nc + Nnv}$$

RV, es el grado de validación de los requisitos.

Nc, es la representación numérica de requisitos validados como correctos.

Nnv, es la representación numérica de requisitos no validados aún.

Para la ejecución del experimento se calcularon las métricas 2.1 y 2.3, las cuales se aplicaron en sólo 3 o 4 requisitos por proyecto, con el fin de evaluarlas en diferentes de contextos. La *métrica de Completitud de los Requisitos* no se empleó debido a que su cálculo es sobre la totalidad de los requisitos necesarios para desarrollar un sistema, además se presume que es resultado de la *métrica de la especificidad de los requisitos* dado que al reducirse la ambigüedad en la especificación de los requisitos serán aceptados por el usuario.

2.8 Matriz de consistencia

Es un instrumento elaborado en una tabla cuyas cabeceras de columnas son el problema, los objetivos, las hipótesis, las variables y técnicas de recolección, donde su información permite evaluar el grado de coherencia y conexión lógica con el título de la investigación. En una visión vertical nos muestra las actividades involucradas para evidenciar los resultados de la investigación y en una visión horizontal la relación del efecto que causa la acción anterior. La Tabla 9, muestra la matriz de consistencia de esta investigación.

Tabla 10

Matriz de Consistencia

PROBLEMA	OBJETIVO	HIPÓTESIS	VARIABLES.	TÉCNICAS DE RECOLECCIÓN
¿Qué mejoras se obtienen en el proceso de elicitación de requisitos mediante la automatización de historias de usuarios con una herramienta gráfica desde un modelo orientado a objetivos con gestión de la trazabilidad?	Desarrollar la Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad para la fase de elicitación de requisitos en el proceso de desarrollo de software.	La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad mejora la fase de captura de requisitos en el proceso de desarrollo de software.	Variable independiente: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad. Variables dependientes: • Reducción del tiempo del proceso de transformación de requisitos en funcionalidades. • Mejora de la calidad del requisito: ✓ Especificidad de los requisitos. ✓ Validez de los requisitos.	• La ejecución de la herramienta para calcular el tiempo de generación de las Historias de Usuario. • La ejecución de la herramienta para obtener las métricas de calidad de requisitos.
PROBLEMAS ESPECÍFICOS	OBJETIVOS ESPECÍFICOS	HIPÓTESIS ESPECÍFICAS		
• ¿Cómo se reduce el tiempo de transformación de requisitos a funcionalidades mediante la automatización de	• Demostrar que la Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la	• La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la		

historias de usuarios con la herramienta gráfica desde un modelo orientado a objetivos con gestión de la trazabilidad? • ¿Cómo se incrementa la calidad del requisito con la aplicación de la automatización de historias de usuarios con la herramienta gráfica desde un modelo orientado a objetivos con gestión de la trazabilidad?	trazabilidad reduce el tiempo de transformación de requisitos en funcionalidades del sistema. • Determinar que la Herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad mejora la calidad de requisitos.	trazabilidad reduce el tiempo de transformación de requisitos a funcionalidades. • La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad mejora la calidad del requisito.		
---	---	---	--	--

2.9 Resumen del Capítulo

En este capítulo se ha descrito el marco Epistemológico relevante para este trabajo de investigación. Se ha iniciado con la definición de la Epistemología del Modelado y la Epistemología de la Trazabilidad. Por otro lado; en el contexto del desarrollo de una herramienta gráfica se realizó la revisión de trabajos relacionados, métodos, modelos o técnicas y software existente.

Luego se describieron los conceptos relacionados con la teoría de la Ingeniería de Requisitos, Framework i*, Historia de Usuario y Trazabilidad.

En atención a los conceptos y, avances existentes relacionados a la presente investigación se definió la Hipótesis General y 2 Hipótesis Específicas, luego se identificaron y operacionalizaron las variables. Finalmente se elaboró la matriz de consistencia para analizar la relación entre el problema, los objetivos de investigación, las hipótesis variables y, las técnicas de recolección.

En el siguiente capítulo se presentará la Metodología a usar para el desarrollo de la herramienta MOOTRACE.

3 CAPÍTULO: METODOLOGÍA

3.1 Tipo y Diseño de Investigación

Según (Hernández, 2016) “Diseño: es el plan o estrategia que se desarrolla para obtener la información que se requiere en una investigación” al respecto (Vara-Horna, 2012), afirma “Los diseños son planes y estrategias de investigación concebidos para obtener respuestas confiables a las preguntas de investigación. El diseño, entonces, plantea una serie de actividades sucesivas y organizadas, que deben adaptarse a las particularidades de cada tesis y que indican los pasos y pruebas a efectuar y las técnicas a utilizar para recolectar y analizar los datos”

A fin de adaptarnos a la cultura ágil y establecer mecanismos para una mejor interpretación de requisitos esta investigación es de tipo Cuantitativo y Diseño Experimental, porque se demostró empíricamente la reducción del tiempo del proceso de transformación de requisitos en funcionalidades e incrementó la calidad del requisito desde sus etapas iniciales generando oportunidades de mejora durante el proceso de construcción del Software. A través de la validez de un requerimiento y su posterior gestión.

3.2 Unidad de análisis

Carbajal (2007) y Hernández y Mendoza (2018) especifican “Para definir la población objeto de estudio, se establecerá la unidad de análisis (personas, organizaciones, instituciones, etc.), delimitándose así la población”.

En una investigación son aquellas que tienen características similares, y que se encuentran en un ámbito determinado.

Son propiedades, características, o cualidades de personas, o fenómenos o hechos a los cuales se aplican los instrumentos para medir las variables. Indica quienes o que van a ser medidos.

En el presente estudio la unidad de análisis fueron los proyectos de software que se desarrollaron en una institución del estado, específicamente en la fase de captura de requisitos.

3.3 Población de estudio

La selección se realizó por muestreo de tipo simple, debido a que cualquier proyecto desarrollado en esta entidad del estado es desarrollado y usado sí y sólo sí en la misma organización. Es no probabilístico por criterio, porque la muestra elegida cumple con 2 razones específicas, primero, los proyectos se desarrollan con el personal de la misma entidad y, por otro lado, los productos finales son de uso institucional, en ese sentido se realizó una selección de proyectos con nivel de complejidad similar.

3.4 Tamaño de muestra

Velázquez (1999), Hernández et al. (2018) consideran la muestra como “una fracción de la población, que cumpla la condición de que, con una probabilidad ‘P’, las conclusiones puedan tener validez para todo el universo”.

Las muestras no probabilísticas, también llamadas muestras dirigidas, suponen un procedimiento de selección orientado por las características de la investigación, más que por un criterio estadístico de generalización (Hernández, 2014).

Razón a ello, la muestra de esta investigación estuvo constituida por 8 proyectos de software desarrollados en el año 2019 dentro y para la misma entidad estatal donde se realizó la presente investigación.

3.5 Técnicas de recolección de Datos

El procedimiento para la recolección de los datos y procesamiento de la información para la presente investigación se realizó durante la ejecución de la herramienta gráfica toda vez que el modelo gráfico haya sido diseñado y se haya procesado esta información luego de pulsar el botón para la generación automática de historias de usuario.

El procedimiento para recolectar los datos primarios de investigación fue el siguiente:

1. Se elaboró un marco muestral conteniendo todas las unidades de análisis (8 proyectos).
2. Se seleccionó el Grupo de Control y el Grupo de Ejecución

3. Se capacitó a los Grupos de analistas de sistemas quienes participan en la captura de requisitos, en el uso de las Herramientas Rational Rose (UML) y MOOTRACE (Framework i*).
4. Se asignó un tiempo prudencial para la aplicación del experimento de acuerdo con la complejidad de cada proyecto de software.
5. Se realizó el modelado en Rational Rose (UML) y MOOTRACE (Framework i*) para diagramar las necesidades del usuario, en el caso de la segunda herramienta el usuario participó durante su elaboración.
6. Se ejecutó la generación automática de Historias de Usuario del modelado gráfico.
7. La información es analizada por el equipo de proyecto para evaluar el impacto de cada Historia de Usuario en el proyecto.

3.6 Técnica de análisis de los resultados

Fórmulas matemáticas para calcular los valores asociados a las unidades de medición de especificidad y validez de los requisitos.

3.7 Resumen del Capítulo

En este capítulo se ha determinado que el Tipo de Investigación es Cuantitativa y es un Diseño Experimental. Además, se definió que la Unidad de Análisis son los proyectos de software que se desarrollaron en una institución del estado, la selección de la población de estudio se realizó por muestreo de tipo simple, el tamaño de la muestra de esta investigación es no probabilística y estuvo constituida por 8 proyectos de software.

Luego, el procedimiento para la recolección de los datos y procesamiento de la información para la presente investigación se realizó durante la ejecución de la herramienta gráfica, después de haberse generado automáticamente desde el modelo gráfico.

Finalmente se implementaron fórmulas matemáticas para analizar los resultados.

4 CAPÍTULO: APOORTE TEÓRICO

En este capítulo se detalla el aporte teórico de la aplicación desarrollada para generar la automatización de historias de usuario desde un modelo gráfico orientado a objetivos bajo el enfoque del marco de trabajo i star con gestión de trazabilidad, siguiendo una arquitectura web a fin de ayudar a las partes interesadas de una manera más flexible, dinámica y con un lenguaje natural, a través de modelos i * que brindarán una visión gráfica e integral de las historias de usuario y sus relaciones.

4.1 Comparación de las tecnologías existentes

Para enfrentar este reto, dos mecanismos de gran ayuda son la adopción de metodologías de desarrollo ágiles y la utilización herramientas para automatizar los procesos y la comunicación usuarios del sistema - equipo de proyecto.

En el contexto de MDD, el modelado es una parte costosa debido al trabajo adicional que se requiere para la construcción de los metamodelos, por lo que se necesita optimizar esta etapa (Fiallos, Suarez, Villavicencio, 2016; Gonçalves et al., 2019). Obtener especificaciones y crear modelos abstractos, para establecer la correspondencia entre los modelos origen y su destino soportado con estereotipos que sean comprensibles en la cadena de comunicación de usuario e ingeniero de requisitos y posteriormente con los programadores sin desvirtuar su objetivo.

De la eficacia de contar con una lista de cotejo de las tareas o matriz de trazabilidad, se hace necesario contar con la eficiencia de una herramienta que nos permita visualizar: el impacto de la correlación de las tareas sobre otras es decir su trazabilidad, granular el esfuerzo del cumplimiento de cada una de ellas en horas de producción sea de un recurso interno o externo, en tal sentido, esta relación permitirá estimar el trabajo específico que conlleva resolver un requerimiento definido en una historia de usuario. En consecuencia, la estimación es más real y acertada en la planificación de la construcción en función al tiempo necesario para las actividades definidas con el fin de cumplir el objetivo.

En la tabla 11, se presenta un cuadro comparativo de las investigaciones de otros autores, el cual destaca aspectos como el modelado, tipo de requisitos que son

considerados en sus modelos, entorno donde funciona la aplicación, fase en la que se usa, ambiente o alcance de la herramienta y tecnología.

Incluso en un entorno ágil, es necesario desarrollar algunos modelos antes de cualquier implementación para garantizar una comprensión compartida por el equipo de desarrollo, de modo que esté sincronizado con los objetivos del valor comercial y el contexto del proyecto (Beatty y Chen, 2012), la mejora de la comunicación entre las partes interesadas es un factor fundamental.

Tabla 11

Cuadro Comparativo de Tecnologías existentes relacionadas a la investigación.

Investigación	Modelado	Tipo de Requisitos	Entorno	Fase	Ambiente	Tecnología
De modelos de negocio a requisitos del software (Hernández-González, Anaísa, Enríquez-Hernández, Dairis Milagros, y Ruano-Chichatskaia, Alexandra. (2021))	BPMN y Casos de Uso	Funcionales y No funcionales	Web	Requisitos y Análisis	A medida	Basado en UML
Aplicación asistida por computadora jMDA basada en modelos para sistemas de información (Moreno-Rodríguez, 2020)	Model Driven Architecture	Funcionales	Local	Requisitos y Análisis	A medida	MxGraph y swing y awt de Java
Planteamiento bajo el enfoque orientada a objetivos para la observación de requisitos en aplicaciones de Internet enriquecidas (Aguilar, Garrigós, Casteleyn y Mazón, 2011).	Orientado a Objetivos	No Funcionales	Web	Análisis	A medida	Algoritmo basado en el óptimo de Pareto
Modelado de Sistema de Recomendación de Objetos de Aprendizaje bajo el enfoque de Ingeniería Web UML (Nieves Guerrero, Ucán y Menéndez, 2014)	Casos de Uso	Requisitos Funcionales	Web	Requisitos y Análisis	A medida	Basado en UML.
Estrategias para gestionar la variabilidad en LPS en pequeñas y Medianas Empresas (Jarzabek, 2010).	Componentes	Requisitos Funcionales	Web	Requisitos	LPS	XCVL

Tabla 12

Cuadro Comparativo de Enfoques de Objetivos para modelado.

Enfoques	GBRAM	KAOS	MYLOPOULOS	LOUCUPOULOS	FRAMEWORK i*
Formal		√			√
No Formal	√		√	√	
Genera Jerarquía AND-OR		√	√	√	
Usa Softgoal		√	√		√
Tiene un método definido	√	√	√	√	√
Identifica conflictos	√	√	√	√	√
Se complementa con otras técnicas de elicitación	√				√
Tiene esquema de representación	√	√			√
Distingue metas de objetivos				√	√

Enfoques	GBRAM	KAOS	MYLOPOULOS	LOUCUPOULOS	FRAMEWORK i*
Tiene heurísticas definidas	√				√
Operacionaliza objetivos en requisitos	√	√			√
Incluye objetivos de calidad	√	√	√	√	√
Taxonomía de objetivos	√	√	√	√	√

Nota. Este cuadro comparativo evalúa algunas características en enfoques de Modelado Orientado a Objetivos donde el Framework i* reúne las mejores condiciones respecto de las otras, Thomas (2005)

En la tabla 12, se visualiza un cuadro comparativo de enfoques asociados que abordan el enfoque de Objetivos como Gbram, Kaos Mylopoulos, Loucupoulos y Framwork i*, la valoración de este último determinó su implementación como parte de la propuesta de la presente investigación.

4.1.1 Comparación de Librerías gráficas.

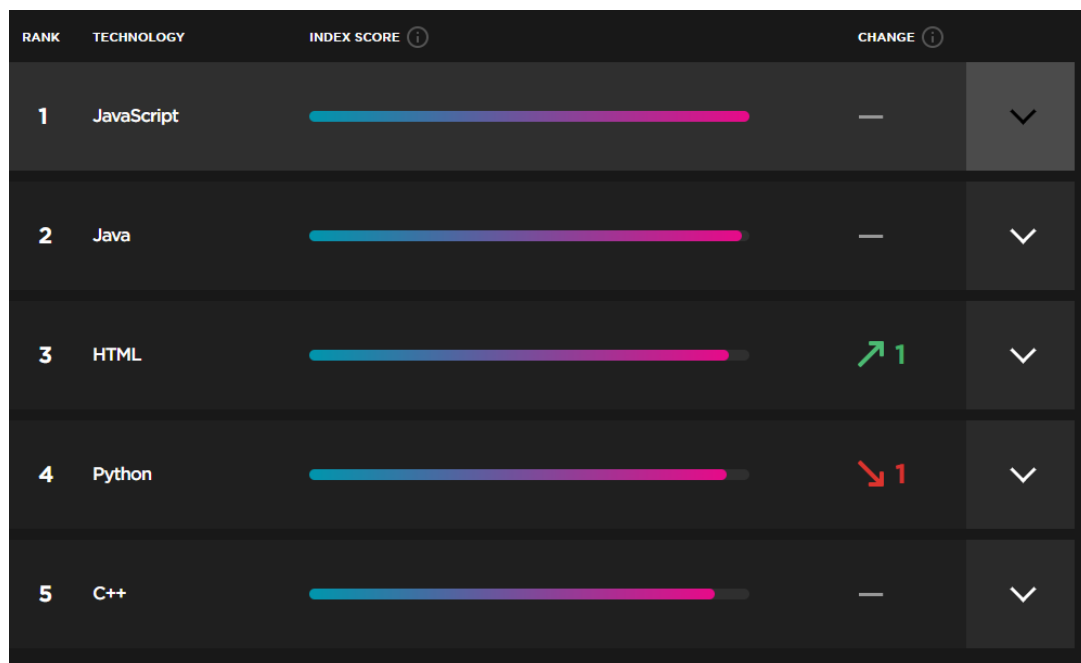
La evolución en el entorno web, establece sus inicios siendo documentos con lenguaje Hypertext Mark Up Language (HTML), hasta las más sofisticadas aplicaciones que son complementadas con un lenguaje de secuencia de comandos como lo es Javascript de código abierto, que mejora la apariencia en las aplicaciones web e innova constantemente con el aporte de colaboradores.

Javascript, por lo general se implementa a través una interfaz en el navegador web, mediante la cual el usuario puede variar su contenido, esta característica es denominada versión cliente-servidor (Coimbra, Tavares, Resende y Terra, 2018).

(Pluralsight Technology, 2004-2020), elaboró un ranking de las mejores tecnologías para desarrollo de software teniendo en cuenta su tendencia de uso entre los desarrolladores a nivel mundial. Resultado de su estudio en el mercado se puede visualizar en la Figura 28.

Figura 28

Ranking de las mejores tecnologías para desarrollo de software.



Nota. Tomado de *Pluralsight Technology Index*, 2004-2020.

En la Tabla 13 se muestra un cuadro comparativo de las principales librerías desarrolladas con la tecnología Javascript donde fueron evaluadas las siguientes características: tipos de gráficos, semejanza con el enfoque de la Metodología Orientada a Objetivo (base de esta investigación) y el tipo de licencia con la que cuentan.

Tabla 13

Librerías para Editores Gráficos.

Librería	Tipo de gráficos	Semejanza con MOO	Licencia
RAPPID	✓ Dependency Management ✓ Workflow Systems ✓ BPMN Tools ✓ Project Management ✓ Diagramming Editors ✓ Floor Planners ✓ Dependency Management	No	Libre
MXGRAPH	✓ Graph Visualization ✓ Graph Interaction ✓ Graph Layouts ✓ Graph Analysis	No	Libre
GOJS	✓ Flowcharts ✓ State Charts ✓ Sankey diagram ✓ Genogram and Medical diagrams ✓ Org charts ✓ Dynamic Ports ✓ Industrial Processes ✓ Dynamic Grouping	Sí	Libre

Estas librerías para programar editores gráficos cuentan con versión de prueba, ampliando sus funcionalidades en versión de paga, se realizó la evaluación de las plantillas de Diagramming Editors (RAPPID), Graph Visualization (MXGRAPH) o State Charts (GOJS), para soportar un entorno visual basado en la metodología orientada a objetivos. En ese sentido, la librería GOJS, a través de su editor gráfico generado con State Charts, soportó los estereotipos de la Metodología Orientada a Objetivos; asimismo, dando una vista intuitiva y versátil.

Las funciones de la librería GOJS respecto al indicador de semejanza con MOO que se usaron para la implementación de la herramienta son:

Diagramas de paleta: Una paleta es una subclase de diagrama que se utiliza para mostrar una serie de partes que se pueden arrastrar al diagrama que el usuario está modificando. Para efectos de la implementación de la herramienta MOOTRACE los elementos que estarán en esta paleta son los estereotipos que establece la Metodología Orientada a Objetivos.

GraphLinksModels: admite enlaces entre nodos y nodos de agrupación y enlaces en subgráficos. GraphLinksModels contiene datos de nodo y datos de enlace en matrices separadas. Los datos de nodo normalmente se representan en un diagrama por instancias de nodo, pero podrían representarse por partes simples o por grupos. Los datos de enlace deben estar representados por instancias de enlace (GoJS, 1998-2020).

Cada enlace referencia dos valores tras su instanciamiento en el diagrama, uno que se refiere al nodo del que proviene y otro al que va.

La propiedad linkFromKeyProperty nombra la propiedad en los datos del enlace cuyo valor es la clave del nodo "de". La propiedad linkToKeyProperty nombra la propiedad en los datos del enlace cuyo valor es la clave del nodo "a". Los valores predeterminados para estas propiedades son "de" y "a" respectivamente (GoJS, 1998-2020).

En la Figura 29, se listan las propiedades y métodos desarrolladas en la librería GoJS.

Figura 29

Propiedades y métodos de la librería GoJS

Properties		
archetypeNodeData	linkFromPortIdProperty	makeUniqueLinkKeyFunction
copyLinkDataFunction	linkKeyProperty	nodeGroupKeyProperty
linkCategoryProperty	linkLabelKeysProperty	nodeIsGroupProperty
linkDataArray	linkToKeyProperty	
linkFromKeyProperty	linkToPortIdProperty	
Methods		
addLabelKeyForLinkData	getGroupKeyForNodeData	setCategoryForLinkData
addLinkData	getKeyForLinkData	setDataProperty
addLinkDataCollection	getLabelKeysForLinkData	setFromKeyForLinkData
containsLinkData	getToKeyForLinkData	setFromPortIdForLinkData
copyLinkData	getToPortIdForLinkData	setGroupKeyForNodeData
copyNodeData	isGroupForNodeData	setKeyForLinkData
findLinkDataForKey	makeLinkDataKeyUnique	setLabelKeysForLinkData
getCategoryForLinkData	removeLabelKeyForLinkData	setToKeyForLinkData
getFromKeyForLinkData	removeLinkData	setToPortIdForLinkData
getFromPortIdForLinkData	removeLinkDataCollection	

Nota. Tomado de *GoJS*, 2020.

4.2 Definición de la solución

Respecto a la metodología Orientada a Objetivos, si bien fue propuesta por Ingenieros de requisitos, el modelado gráfico de las necesidades del negocio utilizando los estereotipos del lenguaje natural framework i*, sirve para interoperar entre el usuario y el analista por medio de la herramienta. A través de los estereotipos propuestos en el marco conceptual del framework i*, permite crear modelos gráficos que representen procesos de negocio a nivel holístico y, por otro lado, detallar las actividades, tareas, recursos, etc. a un nivel granular a fin advertir algún retraso y/o plantear la optimización de alguno de los objetos graficados que pueden entorpecer el funcionamiento del sistema, además, representar de manera explícita la contribución u aporte de los subobjetivos.

La representación gráfica propuesta en el modelamiento framework i* (Yu, 1995), presenta estereotipos de fácil entendimiento y con un lenguaje natural que puede ser utilizado e interpretado por los usuarios sin conocimiento técnico acerca de la definición de requisitos, y de nuestro lado nos da una visión integral de los actores y sus tareas las cuales podrían contribuir en otras y con otros actores, además la prioridad expresada en

los requisitos evidencian al usuario comprometido con su trabajo y conocedor de su negocio lo cual ayuda a refinar el proceso de elicitación.

Agra et al. (2015), proponen como alternativa a la historia de usuario los modelos BPMN, sin embargo, debe quedar claro que los modelos en cuestión no ofrecen una visión detallada de cómo lograr los objetivos, y tampoco tienen alcance sobre los Requisitos No Funcionales. Por otro lado, los artículos descritos por (Rubin, 2012; Jiménez, 2012) sí los considera basando su importancia en factores que den valor a la calidad del software. Conviene destacar que la presente propuesta evita el trabajo manual e interpretación entre los involucrados (Serna, 2013) técnicas que son históricamente usadas en el análisis, desarrollo y gestión de los requisitos, el uso del lenguaje verbal ocasiona en el desarrollo de software la pérdida de información, definiciones ambiguas o sobre valoradas que en consecuencia han traído impactos negativos en costo, tiempo y calidad del producto.

Estudios como (Hoda, Kruchten, Noble y Marshall, 2010; Wang, Zhao, Wang y Sun, 2014), determinan el impacto de la implementación de prácticas de ingeniería de requisitos en entornos ágiles. Su mejora en la calidad de producción del software y como fortalece este enfoque como buena práctica. Lucassen, Dalpiaz, Van der Werf y Brinkkemper (2016), detallan específicamente un estudio acerca del uso y la efectividad de historias de usuario en la práctica. En la sección de Discusión y conclusión los Hallazgos (H) encontrados y definidos son:

Hallazgo 1: La mayoría de los usuarios de historias de usuario (94%) los usan en combinación con Scrum.

Hallazgo 4: La estructura simple de las historias de usuario permite desarrollar el software adecuado porque facilita la creación de un entendimiento común sobre el requisito.

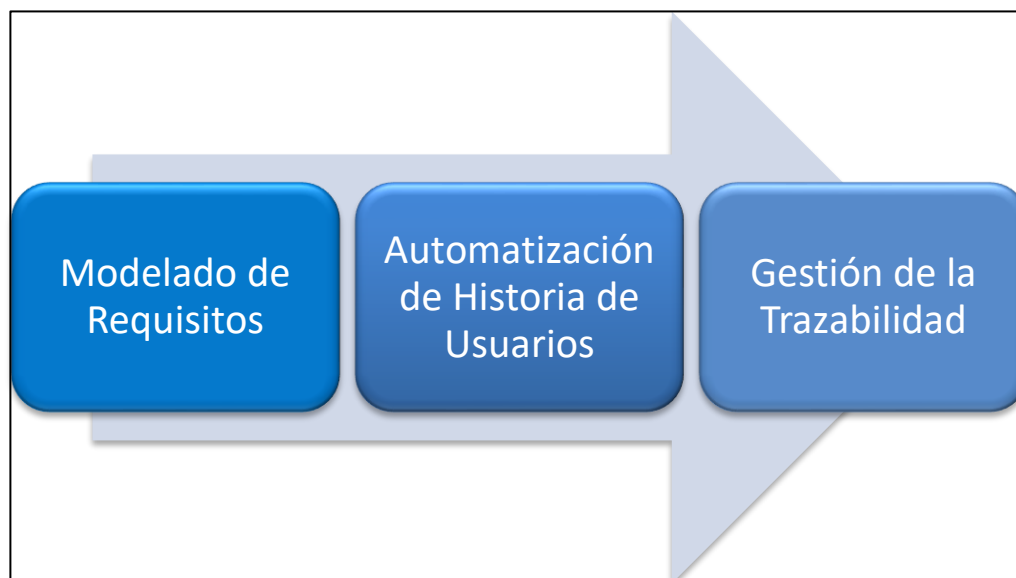
Por otro lado, los Hallazgo 2, Hallazgo 3, Hallazgo 6, Hallazgo 7, Hallazgo 8, Hallazgo 9 y el Hallazgo 10 establecen que este artefacto (HU) debe cumplir con estándares como INVEST o Connextra (formato de HU) a fin de mantener un estándar tanto para los que se inician en la implementación de alguna metodología ágil y para los experimentados quienes orientan sus esfuerzos a disminuir sus tiempos de producción.

La información técnica vertida en una Historia de Usuario es sencilla y se convierte en una potente herramienta para el equipo de desarrollo, en tanto se alimente con datos reales ofrecidos por el usuario y el equipo ágil.

Los procesos de desarrollo ágil tienen una perspectiva diferente en comparación con los procesos de desarrollo tradicionales que siguen una línea más secuencial o en cascada, lo que implica dependencia de la fase anterior y la espera obligatoria para completar la última. Así también, la generación de artefactos para documentar el análisis detallado que exigen las metodologías tradicionales evidencia una posible carencia de las metodologías ágiles (Figuerola-Díaz, Sólis y Cabrera-Silva, 2007). Pero que han sido cubiertas con metodológicas, frameworks o con herramientas como la propuesta en el presente estudio de modo que las nuevas tendencias de desarrollo de software promovidas en entornos ágiles tengan los activos necesarios para fortalecer y hacer sostenible los procedimientos y métodos de trabajo donde se adopten estos principios. La Figura 30 muestra los componentes conceptuales para el desarrollo de la Herramienta Gráfica.

Figura 30

Esquema conceptual de la Herramienta Gráfica



4.2.1 Modelado de Requisitos bajo el enfoque Orientado a Objetivos

Este enfoque destaca la importancia de los objetivos como motivación para el éxito de un sistema, porque los stakeholders y analistas alcanzan a comunicarse con un lenguaje sencillo y fácil de interpretar a través de este modelado. En ese sentido, de acuerdo con las características mostradas en la Tabla 11: Cuadro Comparativo de Tecnologías existentes, se eligió el Framework i* para ser implementados en MOOTRACE, debido a que sus estereotipos establecen un esquema de representación fácil de usar y además considera los Requisitos Funcionales y los No funcionales los cuales determinan la calidad de la solución desarrollada. La notación del Framework i* representada por sus estereotipos que se visualizan en la Figura 33 columna “símbolo” y, asimismo, define el concepto que representa cada uno dentro de un diagrama con modelamiento orientado a objetivos. Respecto al símbolo Contribution link, cabe enfatizar que su importancia radica en la información ingresada en estos “links” o enlaces en la fase de creación del modelo gráfico enriqueciéndolos para que finalmente sirva de insumo en la generación automática de las Historias de Usuario.

Representado en dos tipos de modelos:

1. Modelo de Dependencia Estratégica (SD por sus siglas en inglés), en el cual se representan las relaciones y dependencias entre actores en un contexto organizacional, para mostrar los efectos positivos y negativos de cada actor dando una visión holística del sistema. Los modelos gráficos de este diagrama permiten visualizar de manera integral e intuitiva los objetivos estratégicos del negocio como se visualiza en la Figura 31.
2. Modelo de Justificación de Dependencia (SR por sus siglas en inglés), representa el comportamiento del actor en el negocio mediante objetivos y tareas, es decir muestra cómo abordar sus necesidades. En la Figura 33 se visualiza el modelo SR, que se empleará como parte de la solución porque granula el requerimiento, permitiendo que sea más refinado y preciso para la representación de las necesidades del cliente. Además, tanto él como el equipo de proyecto utiliza el mismo lenguaje, ellos para comunicar y los analistas para interpretar sus necesidades.

La participación del usuario en la fase de elicitación de requisitos es más activa cuando su aporte lo ve materializado en el proceso de desarrollo del producto, con esto reforzamos la idea que el usuario es el recurso más importante para la definición de un requisito. El diagrama (SR) permite detallar sus tareas, en las especificaciones en los modelos de Casos de Uso no se consideran los RNF y los RF (funcionalidad) corresponden a un gráfico individual lo que complica su trazabilidad, puesto que al querer identificar sus relaciones tendríamos la información en 2 documentos distintos.

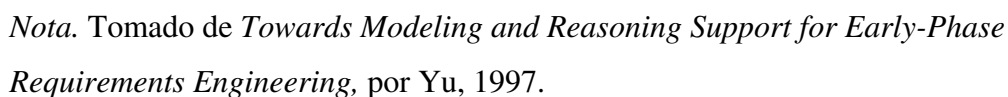
Figura 31

*Notación de Framework i**

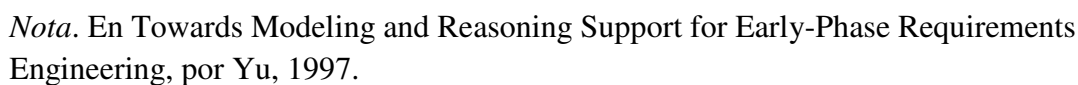
Símbolo	Nombre	Símbolo	Nombre
	SD Model		SR Model
	Actor		Goal
	Task		Resource
	Softgoal		Means-end link
	Task-decomposition link		Contribution link

Nota. Tomado de *Una propuesta orientada a objetivos para el análisis de requisitos en RIAs*, por Aguilar et al. 2011

Strategic Dependency.



Strategic Relational



A través de los estereotipos para los gráficos de BPMN, se representa la serie de procedimientos y las comunicaciones generadas entre los roles que cumplen funciones dentro del negocio, pero no dice cómo las realizan. Tampoco consideran los RNF y para las especificaciones a mayor detalle debe de utilizar un modelador para ese propósito.

La existencia de sistemas que funcionalmente cumplen su propósito, pero que el usuario espera aún más. En consecuencia, se tiene insatisfacción del usuario debido a que la lentitud de respuesta del sistema o la confiabilidad de sus datos no cumple sus expectativas, y un software que no se puede usar.

Estas razones o requisitos no funcionales deben ser considerados con la misma importancia que las capacidades funcionales del producto.

En el modelado con el Framework i^* (Yu, 1995), se visualiza la interrelación entre los requisitos Funcionales (goals) y los NFR (softgoals) con una visión global y determinar la interdependencia de manera clara, ayudando a los diseñadores a entender las condiciones de validez para garantizar la capacidad del software para responder a las necesidades del usuario y proporcionar información para una adecuada toma de decisiones.

Sommerville (2011) establece la existencia de una relación de causa efecto entre los requisitos No Funcionales (RNF) y la calidad del software, características como: fiabilidad, tiempos de respuesta o la capacidad de almacenamiento correctamente especificadas garantizan que la aplicación no sea sólo útil sino también usable. Según Buitrón, et al. (2017), consideran que los Requisitos no funcionales se “refiere a cómo se comporta el sistema frente a atributos observables como desempeño, reusabilidad, confiabilidad, entre otros”.

Las soluciones sistemáticas en la actualidad deben de enfocarse a cubrir necesidades para lograr la mejora continua del negocio, por ejemplo, entregando eficientemente información de calidad para la toma de decisiones. Los diagramas (SR), facilitan la transparencia de los requisitos del software a todos los interesados. Los modelos como técnica no sólo ofrecen beneficios para que podamos comprender los requisitos de sistemas complejos, sino, además son aplicables a entornos más pequeños. Debemos de

tener en cuenta que los negocios son incrementales en el tiempo y se debe generar mecanismos para responder oportunamente a estos cambios, y desarrollar la capacidad para adaptarnos.

4.2.2 Generación de Historias de Usuario

Desarrollar una historia de usuario es un proceso complejo, y normalmente requerirá la intervención de varios miembros del proyecto desde el Dueño de Producto, capaz de “traducir” una solución a un lenguaje en el que el resto del equipo pueda trabajar; un programador, quien puede estimar cuánto tardará en implantarse, o qué dependencias tiene una historia de usuario respecto a otras, alguien que la pruebe e incluso puede que un diseñador o un analista.

La información que se extrae del modelo gráfico con enfoque de la metodología orientada a objetivos es vertida de manera automática e intacta reduciendo el margen de subjetividad que se genera cuando el proceso es manual o se usa diferentes formatos para refinar los requisitos.

Sin embargo, hay datos técnicos para su viabilidad que deben ser incorporados por el grupo de desarrollo; por ejemplo: Asignar un valor de prioridad a la HU, duración de la tarea y estado de la HU y que corresponde a otra fase del ciclo de vida de la ingeniería del software.

El listado de tareas representa las acciones que viabilizan el cumplimiento del objetivo o el “Para” de la Historia de usuario; el 100% representa una capacidad del sistema en desarrollo y que es tangible para los usuarios porque pueden determinar si es útil o no para su trabajo.

Los Softgoals o SubObjetivos de la MOO, representarán el ítem dentro del formato de la historia de usuario que abordará el “Criterio de Aceptación” para realizar el seguimiento correspondiente de la validez o no, en el cumplimiento de su desarrollo en consecuencia definir el nivel de satisfacción del usuario.

En consecuencia, la Historia de usuario es una propuesta simple y concreta, cuyo 100% de cumplimiento es evaluado dicotómicamente.

4.2.3 Gestión de la Trazabilidad

La trazabilidad facilita la mantenibilidad de los sistemas, reduce drásticamente los costos de retrabajo y mejora la productividad. Sin embargo, sin la metodología y tecnología adecuada, puede ser altamente ineficiente y, por el contrario, causar situaciones engorrosas o descripciones poco entendibles para el equipo de trabajo. En ese sentido las relaciones dibujadas en el diagrama del modelo gráfico fueron convertidas en trazas explícitas, la herramienta desarrollada MOOTRACE las incorpora con validación semántica según la matriz de correspondencia de la figura 34, para producir de forma automática la historia de usuario digitalizada.

Cualquier alteración en los elementos/artefactos que componen el modelo gráfico es replicado en la generación de historia de usuario haciéndolo trazable. Asimismo, las iteraciones en el equipo de trabajo son más productivas debido a que las vistas a la que accede cada uno en función a su rol, se actualizan en tiempo real y puede advertir el impacto del cambio. Como consecuencia se identifica tempranamente las dependencias, conflictos entre requisitos y, oportunidades de mejora en la comprensión del sistema en su totalidad.

El enfoque de trazabilidad de la herramienta gráfica desarrollada aplica para metodologías ágiles que han tomado fuerza estas últimas décadas. Asimismo, la automatización de la trazabilidad en MOOTRACE para la generación de las Historias de Usuario, está basada en el enfoque de la MDA con el uso de librerías disponibles para presentar entornos más dinámicos e intuitivos al usuario, en los contextos ágiles.

La trazabilidad no sólo debe regirse sobre el desarrollo del requerimiento, sino también saber a qué actividades están asignados los recursos con la finalidad de estimar y planificarlos adecuadamente, por ejemplo:

Si un desarrollador terminó una HU asignada, de inmediato debe liberarse este recurso. La HU debe mostrarse como terminada y el recurso debe ser asignado a otra HU u otro sprint.

Otro ejemplo, es conocer sobre cuántas historias de usuario tiene implicancia un sólo actor y analizar la optimización como una posibilidad si fuera el caso. A través de este mecanismo es posible seguir el impacto de dicho requisito a lo largo de las actividades para el desarrollo del producto y para su mantenibilidad.

En consecuencia, el valor de la información que se desprende de una adecuada trazabilidad genera un incremento de valor en un producto o proyecto sin perder de vista el objetivo teniendo una vista integral.

4.2.4 Correspondencia conceptual para solución de la Herramienta Gráfica MOOTRACE.

La definición de las notaciones en los 2 contextos: modelo gráfico e Historia de Usuario es base en la solución propuesta (véase Figura 30), porque deben coexistir para fortalecer sus atributos y mitigar sus deficiencias (referencia de los que destacan atributos deficiencias). Cuando los usuarios visualizan la información que están entregando mediante gráficos se promueve una comunicación más asertiva y, ayuda a reducir tiempo y esfuerzo al equipo de proyecto en interpretar oportunamente los datos para generar conocimiento, que posteriormente ayudará al equipo de desarrollo definir la arquitectura de la aplicación.

Es cierto, que el valor es asignado por el Product Owner, y será más congruente si puede tener un dato base como el aporte que entrega el usuario en la fase inicial a través modelado de SR, esta relación de trazabilidad produce información al proceso de producción de software.

Figura 34

Correspondencia entre el Modelo Strategic Rationale (SR) y la Historia de Usuario

Modelo de Justificación Estratégica (SR)		Historia de Usuario	
Símbolo	Notación	Símbolo / Característica	Notación
	Actor: Rol que desea alcanzar su objetivo y que además está preocupado por las implicaciones a largo plazo de sus relaciones estructurales con otros actores	Como:	Es quien hará uso de la funcionalidad
	Objetivo: Condición o estado que al actor le gustaría alcanzar	Necesito:	Qué espera el usuario final
	Tarea: Una forma particular de hacer algo para cumplir un objetivo	Para:	Acciones u otras alternativas para llegar al mismo fin
	Objetivo suave: Es una condición que al actor le gustaría lograr, pero a diferencia del objetivo, el criterio para la condición de su cumplimiento que no se define a priori y está sujeto a interpretación.	Criterios de aceptación:	Condiciones que sirven para verificar que se ha concluido satisfactoriamente el desarrollo de la funcionalidad
	Contribución: Nivel de importancia negativa o positiva	Valor:	Escala cualitativa que representa la importancia de cada requisito
	Relaciones: Tipos de enlaces entre objetivos-tareas, tareas-recursos, objetivos-objetivos suaves	Dependencias:	Relaciones con otras Historias de Usuario
	Recurso: Es un entidad física o informativa	Encargado de la historia de usuario:	Desarrollador/Analista encargado de la historia de usuario.

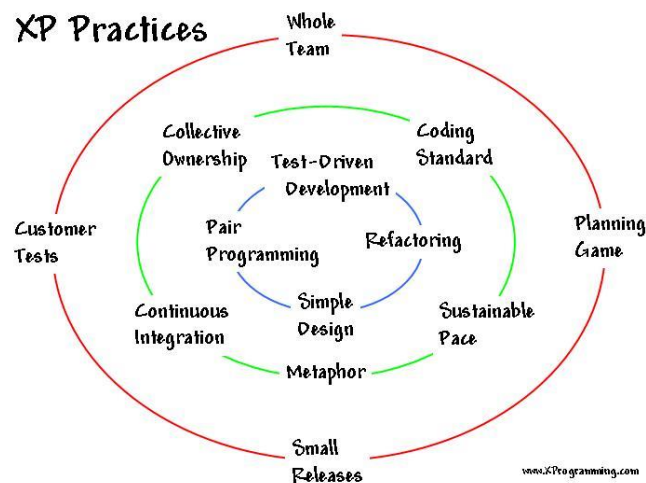
La Figura 34, establece la correspondencia entre la notación del Strategic Rationale (SR) del Modelo Orientado a Objetivos propuesto por Erick Yu y la Historia de Usuario como artefacto usado para especificar requisitos en las Metodologías ágiles. Se hace necesario resaltar que un artefacto no sustituye al otro, por el contrario, se complementan para nutrirse recíprocamente sin perder su esencia, así también aclarar que el uso de estos artefactos no será intrínseco a las funciones técnicas del equipo de desarrollo porque promueve los valores establecidos en el manifiesto ágil acerca de la participación de las personas y sus relaciones, cooperación con el cliente y replica ante la variabilidad. En ese mismo contexto, las carencias de uno serán reforzadas por las cualidades del otro (Figuroa-Díaz, Sólis, Cabrera-Silva, 2007) a fin de optimizar el tan delicado proceso ingenieril como lo es la elicitación de requisitos para construir software de calidad.

4.3 Metodología para el desarrollo de la herramienta MOOTRACE

Para el proceso de desarrollo de la Herramienta Gráfica se siguieron las fases de la Metodología Ágil XP (Xtreme Programming), formulada por (Beck y Andres, 2004), autores del primer libro sobre la materia en el cual se da mayor énfasis a la adaptación de los requisitos del sistema. Además, proponen valores, principios y prácticas a tener en cuenta para el desarrollo de software, dentro de las cuales está la simplicidad con la que se debe desarrollar, con diseños sencillos y donde la funcionalidad debe representar las necesidades del cliente. Todo proyecto de software en XP inicia con una o varias reuniones con el cliente, este rol fue desempeñado por cada analista del departamento de Desarrollo de Aplicaciones de una entidad del estado y el rol de programador por las autoras, las reuniones ayudaron a dar claridad de la necesidad puntual del mismo a través de las historias de usuario para finalmente crear las pruebas de aceptación antes de iniciar la codificación.

Figura 35

Metodología Xtreme Programming.



Nota. Tomado de *Metodología Xtreme Programming*.

<https://ronjeffries.com/xprog/what-is-extreme-programming/> Ron Jeffries, 2020.

Concluida esta etapa y con un plan de entregas se pactó 3 iteraciones al principio y se extendió una más por común acuerdo para levantar las observaciones y cerrar la iteración pendiente. La figura 35, muestra el ciclo de la propuesta de la metodología XP.

4.3.1 Fase de Planeación

Durante la Elicitación de Requisitos, fase inicial, la comunicación con el usuario fue en un lenguaje coloquial y sencillo, las alternativas se evalúan con respecto a las metas que se desean alcanzar. La información generada en esta etapa debe ser fehaciente para nutrir una buena especificación. Con la participación del usuario se logra que el margen de contribución sea mayor en el ciclo de desarrollo del software.

En este proceso, se estableció la repercusión de la herramienta en el marco de una estructura conceptual y el diseño técnico. Los requisitos, se elicitaban en el formato de historias de usuario, las cuales representaron las características que debía tener la herramienta gráfica:

1. Permitir el modelado gráfico de requisitos con base en la Metodología Orientada a Objetivos.
2. Generación automática de Historias de Usuario tomando como entrada el modelo gráfico generado en el punto 1.
3. Refinamiento de la Historia de Usuario:
Priorización de la HU, duración de la tarea y estado de la Historia de Usuario.

Las iteraciones están ligadas a la entrega de avances en un lapso de 1 a 3 semanas.

Bajo el enfoque de las metodologías ágiles se debe concebir que el proceso de desarrollo de la aplicación es dinámico y, el equipo a cargo tener la capacidad de identificar tempranamente los ajustes necesarios para alcanzar el esperado nivel de satisfacción del usuario.

4.3.2 Historias de Usuario.

Las Historias de Usuario se digitalizaron, pero siempre bajo la orientación del usuario, a diferencia del documento de Requisitos de usuario de la metodología tradicional, donde el analista “interpreta” lo que el cliente describió.

Con la digitalización de las HU, se logró el alcance de esta investigación cuyo objetivo general es la obtención automática de Historias de Usuario, desde un modelo gráfico previamente dibujado a través de una interfaz útil y usable.

Inicialmente se define la Historia de Usuario 1: Crear Editor Gráfico en la cual se especifica la necesidad de graficar la comunicación sostenida con los stakeholders (del Ingeniero de requisitos, el usuario final, etc; la Historia de Usuario 2: Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico, los stakeholders para relevar sus metas (objetivos), establecer los aportes positivos y negativos de los requerimientos, para definir esta historia de usuario se usó como base el modelo SR (Strategic Relationale) de la Metodología Orientada a Objetivos, que hace uso del lenguaje de framework i* con estereotipos que representan actores, tareas, recursos, sub objetivos y relaciones entre ellos en un lenguaje natural sin perder el enfoque del objetivo principal, luego la Historia de Usuario 3: Generar automáticamente la Historia de Usuario con la trazabilidad del modelo gráfico, el usuario debe visualizar los requerimientos especificados en el modelo gráfico en un formato de Historia de Usuario. Finalmente, Historia de usuario 4: Personalización de Historia de Usuario y cierre de trazabilidad, el usuario debe complementar la información técnica de la Historia de Usuario como asignar la prioridad al requerimiento, duración de la tarea y estado de la Historia de Usuario que permita posteriormente determinar si el requisito ha sido implementado.

A continuación, se presenta la Tabla 14, que contiene la Historia de Usuario 1: Crear Editor Gráfico, adicionalmente en el Anexo 2 se encuentran las Historias de Usuario que permitieron completar el desarrollo de la herramienta gráfica MOOTRACE: Tabla 28: Historia de Usuario 2: Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico, Tabla 29: Historia de Usuario 3: Generar automáticamente la Historia de

Usuario con la trazabilidad del modelo gráfico y, Tabla 30: Historia de usuario 4: Personalización de Historia de Usuario y cierre de trazabilidad.

Tabla 14

Historia de Usuario 1.

Historia de Usuario	
Número: 1	Usuario: Analista
Nombre historia: Crear Editor Gráfico	
Prioridad en negocio: Alta	Riesgo en desarrollo: (Alta / Media / Baja)
Puntos estimados:	Iteración asignada: 1
Programador responsable: Delicia Lucero	
Descripción: Genera el editor gráfico con el modelo conceptual de la metodología orientada a objetivos.	
Observaciones: Ninguna.	

4.3.3 Plan de Iteración

Tabla 15

Plan de Iteración

	Iteración 1	Iteración 2	Iteración 3	Iteración 4
Horas	50	40	36	15
Semanas	2	2	2	1
Horas Semanales	25	20	18	15
Historia de usuario	1	2	3	4

La Tabla 15, establece la distribución de las 4 iteraciones, en la primera se importó con Entity Framework que es el ORM (Object-Relational Mapper) de Microsoft, desde una

base de datos existente se generaron las clases del modelo en el código fuente. En la segunda se referenció la librería GOJS, para las personalizaciones en la paleta de herramientas en la cual se encuentran los estereotipos propios del modelado de requisitos orientado a objetivos, posteriormente se realizó la validación del propósito de cada uno. En la tercera iteración, se ha especificó la automatización de las relaciones dentro del modelo para la generación de Historias de usuario y finalmente en consenso se estableció la importancia y mejora del formato de la historia de usuario alimentándola con datos como su prioridad, duración de la tarea y su estado a fin de tener una trazabilidad concreta del requisito inicial.

4.3.4 Fase de Diseño

En esta fase a diferencia de la metodología tradicional, donde se diseñan vistas generales de lo que será el sistema; este modelo propone revisar y mejorar continuamente; cada iteración con el usuario debe tener como consecuencia aprobaciones y propuestas para plasmar las funcionalidades necesarias:

1. Creación del proyecto.
2. Generación automática de las Historias de Usuario.
3. Asignar la prioridad de la HU, duración de la tarea y estado de la HU.

Hacer lo necesario en XP, irrestrictamente representa el ahorro de tiempo es tan esencial así que no se debe invertir esfuerzo en lo que no se necesita o no se contempla algo porque existe el riesgo que no se use.

En este caso se ha utilizado el modelo propuesto por Erick Yu, para desarrollar el editor gráfico y la interfaz del formato correspondiente a las HU.

4.3.5 Fase de Codificación

El código fuente se desarrollará bajo el enfoque del patrón MVC en donde los controladores y el modelo son totalmente transparente para el usuario, es decir, si se aplican cambios a alguna de las dos partes, no afectará a la Vista. Las clases para el modelo fueron generadas con el ORM de Microsoft.

En la vista se utilizará la hoja de estilo de bootstrap, una librería gratuita y de código abierto para la personalización del DOM (Document Object Model) cuya plataforma

establece un conjunto estándar de objetos para representar documentos en HTML5 propias de las páginas web dinámicas, con la cual las vistas adquieren la capacidad de hacerse responsivas y asignar características para mejorar su presentación a partir de clases predefinidas en sus plantillas, evitando el trabajo sobre cada uno de los objetos.

4.3.6 Fase de Pruebas

Se requirió la realización de 2 tipos de pruebas:

4.3.6.1 Pruebas unitarias

Estas pruebas fueron diseñadas antes de la codificación de los métodos, el valor agregado resulta porque una prueba unitaria satisfactoria implica en mayor porcentaje lo más cercano a la construcción final del método con la certeza que ya funciona y fueron realizadas por el mismo programador.

Luego de esto, debe considerarse la integración al resto de funcionalidades, esto corresponde al enfoque incremental propio de la metodología XP.

Es importante mencionar que, por el contexto de la herramienta gráfica, uno de los usuarios es el programador y aunque no puede ser juez y parte, es cierto que en primer término se puede dar un gran aporte en el proceso de desarrollo.

4.3.6.2 Pruebas de aceptación

Estas son pruebas se realizaron por el usuario para verificar que el sistema cumple con los requisitos solicitados es decir se contrastaron con lo especificado en las Historias de usuario 1, 2, 3 y 4.

Las entregas parciales se hicieron para asegurar el avance en la misma línea de la necesidad del usuario y sin perder de vista el objetivo final.

Es importante resaltar la diferencia entre las pruebas unitarias y las de aceptación respecto a quién las realiza. Para el primer caso es de índole técnico es decir solo le compete al programador, en las pruebas de aceptación es el usuario quien la realiza y ejecuta según su historia de usuario al final de cada iteración según las historias implementadas en los planes de entrega.

En consecuencia, se describirán las pruebas que contienen los criterios de validación para la aceptación por cada historia de usuario que fueron realizadas para el desarrollo de la herramienta llamada MOOTRACE, como se visualiza la Tabla 16 referida a la Especificación de Prueba: Crear Editor Gráfico (Historia 1), adicionalmente en el Anexo 3 se encuentran las Especificaciones de Prueba que permitieron comprobar el correcto funcionamiento y uso de la herramienta gráfica MOOTRACE, Tabla 31: Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico (Historia 2), Tabla 32: Especificación de Prueba: Automatización de las relaciones de los objetos del modelo SR para generar automáticamente las Historias de Usuario establecidas en la correspondencia de la Figura 31 (Historia 3) y Tabla 33: Personalización de Historia de Usuario (Historia 4).

Tabla 16

Especificación de Prueba: Crear Editor Gráfico (Historia 1)

<i>Especificación de Prueba: Crear Editor Gráfico (Historia 1)</i>	
Descripción	
En esta historia se debe comprobar que la interfaz cuenta con una barra de herramientas, botones para abrir, crear y guardar proyectos. Los nodos deben de permitir editar su descripción, eliminarlos y relacionarse para ello se debe de importar la librería go.js	
Creación correcta de un modelo	
Descripción	
El usuario debe tener a disposición un botón para abrir los modelos de los proyectos disponibles que previamente han sido creados.	
El usuario debe tener a disposición un botón para crear un nuevo modelo, luego debe ingresar el nombre a asignar al modelo que se diseñará.	

El usuario podrá hacer clic sobre el elemento que está en la barra de herramientas y luego hacer clic en el panel para que se visualice en el modelo gráfico.

Condiciones de ejecución

Ninguna.

Entrada

- Del menú seleccionará “Nuevo”.
- Se habilitará la edición del nombre que le asignará a su modelo.
- El cliente seleccionará de la barra de herramientas los elementos que desee para su modelo.
- Luego presionar el botón “Guardar”.

El sistema mostrará un mensaje informando el registro exitoso.

- Si selecciona añadir otro modelo, realizará la misma secuencia de lo antes descrito.

Resultado esperado

Tras el diseño del modelo este se debe de listar en la lista de proyectos disponible para editar.

Evaluación de la prueba

Prueba Exitosa.

Creación incorrecta de un modelo

El usuario crea el modelo y al guardarlo sin haber ingresado el nombre, el sistema debe validar la que se haya ingresado una descripción para el modelo que se ha diseñado.

Condiciones de ejecución

Ninguna.

Entrada

- Del menú seleccionará “Nuevo”.
- El cliente seleccionará de la barra de herramientas los elementos que desee para su modelo.
- Luego presionar el botón “Guardar”.

Se mostrará un mensaje denegando el registro del modelo “Ingrese nombre para el modelo”.

Resultado esperado

Tras el diseño del modelo, este no se guardará en la base de datos.

Evaluación de la prueba

Prueba exitosa.

4.3.7 Resultados de las Pruebas

Caso de Estudio: Usabilidad de la solución

Se realizó un análisis cuantitativo para evaluar las métricas aplicando de casos de prueba en el uso de la herramienta MOOTRACE en base a los factores propuestos en el estándar de evaluación de la calidad del software ISO/IEC 25010 (D. Lucero, 2018) con el fin de identificar el nivel de usabilidad de la interfaz para la creación de un modelo gráfico y la Historia de Usuario generada de manera automática.

El perfil de los participantes tiene una mediana de 8 años de expertiz en análisis y diseño de sistemas, de los participantes (40% tenían menos de 5 años de experiencia).

Cuando los usuarios visualizan la información que están entregando mediante gráficos permite al equipo de proyecto, comprender y comunicar la información de manera más asertiva, en ese sentido el equipo técnico del proyecto puede evidenciar con mayor precisión las necesidades de los usuarios y asistirlos pertinentemente.

A continuación, se mencionan algunas de las preguntas que fueron incluidas en la encuesta.

¿Los estereotipos usados en el modelo gráfico son entendibles?

Un noventa por ciento de las personas consultadas precisa que los estereotipos propuestos en el modelo gráficos son intuitivos.

¿Al crear el modelo gráfico este representa de manera sencilla los requerimientos esperados por el cliente?

Un ochenta y cinco por ciento de los encuestados refiere que el modelo gráfico representa lo requerido, aunque esperan un mejor detalle.

¿Al automatizarse la creación de las Historias del Usuario, considera que el tiempo ahorrado es relevante?

El 100% de los encuestados afirma que es importante el ahorro de tiempo en la automatización de este formato.

4.4 Resumen del capítulo

Para el desarrollo de los componentes la herramienta propuesta se utilizó la metodología de desarrollo ágil XP, la cual plantea un conjunto de tareas iterativas que involucra de manera cooperativa y crucial la participación del usuario en la misma, a fin de asegurar un mayor nivel de calidad del software.

Al respecto se elaboraron las Historias de Usuario, que permitieron desarrollar la herramienta MOOTRACE: H1: Modelado de requisitos, H2: Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico, H3: Generar automáticamente la Historia de Usuario con la trazabilidad del modelo gráfico y H4: Personalización de Historia de Usuario y cierre de trazabilidad.

Por otro lado, se elaboró el Plan de Iteración a través de 4 ciclos de desarrollo cortos, luego se diseñó las interfaces para la (i) Creación del proyecto, (ii) Generación automática de las Historias de Usuario y (iii) Asignar la prioridad de la HU, duración de la tarea y estado de la HU.

En la Fase de Codificación establecida en la metodología XP, se definió la tecnología para la arquitectura, estilos y código fuente a utilizar para el desarrollo de la herramienta MOOTRACE.

Finalmente, en la Fase de Pruebas se definieron 2 tipos de pruebas: Unitarias para determinar el correcto funcionamiento de porciones preliminares de código y las Pruebas de Aceptación para determinar el correcto funcionamiento el sistema de acuerdo a los requisitos solicitados por los usuarios finales, lo cual fue comprobado con el desarrollo de un Caso de Estudio para determinar la usabilidad de la solución

5 CAPÍTULO: DESARROLLO DE LA HERRAMIENTA

Lucero D., propone un esquema arquitectural para su herramienta gráfica denominada MOOTRACE, el cual se visualiza en la Figura 36, compuesto por 3 partes arquitecturales: el Modelo, la Vista y el Controlador conocido como el patrón MVC, que se implementa para desarrollar aplicaciones de manera ordenada y que se caracteriza por asignar responsabilidades operativas a los componentes que a continuación se detallan:

Modelo: Los objetos o entidades del modelo son porciones de código que establecen la lógica de las clases conceptuales de la aplicación por lo general para recuperar y almacenar su condición se realiza a través de mecanismos de persistencia.

Vista: Es el componente de cara al usuario y con el que interactúa de manera directa. Por lo general, la interfaz o vista es congruente con el modelo de datos y representa su estado en un momento concreto y en el contexto de una acción determinada.

Controlador: Es el componente que orquesta la comunicación entre las acciones de usuario, las respuestas del modelo y la presentación de la vista o interfaz (Valdecantos, 2010).

Figura 36

Arquitectura Conceptual MOOTRACE.



5.1 Tecnología del software

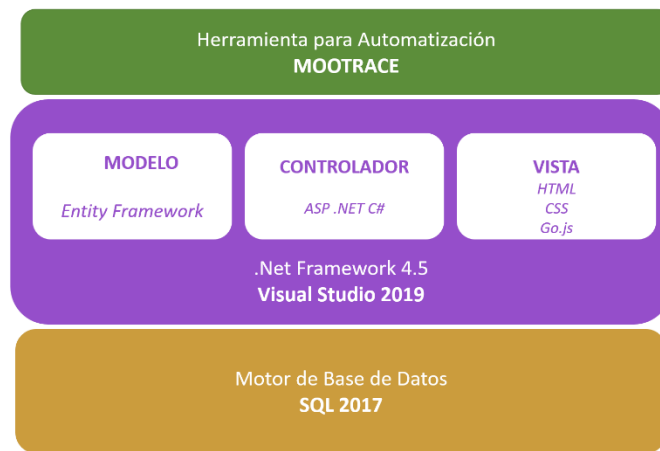
Al respecto del punto anterior, queda proponer una tecnología pertinente a la arquitectura propuesta, en ese sentido, la Figura 37 muestra los componentes seleccionados para su implementación.

El IDE de Visual Studio en su versión 2019 integra 4 modelos de programación para entornos web: **ASP.NET MVC**, Web Forms ASP.NET, ASP.NET Web Pages y ASP.NET en ese sentido el primer modelo mencionado y el lenguaje de programación C# serán utilizados para el desarrollo de la herramienta. Por otro lado, para la persistencia de la información se empleará el motor de base de datos SQL server 2019.

A través del elemento HTML Canvas, se genera un entorno para dibujar gráficos dinámicos apoyado del lenguaje de programación java script de la librería go.js, logrando una plataforma enriquecida con gráficos más complejos los cuales pueden editarse, crearse, eliminarse y, configurando los nodos de manera muy sencilla y rápida ya que no estamos manejando imágenes y definir el diagrama que soporte los requisitos del usuario.

Figura 37

Arquitectura Tecnológica de la herramienta gráfica.



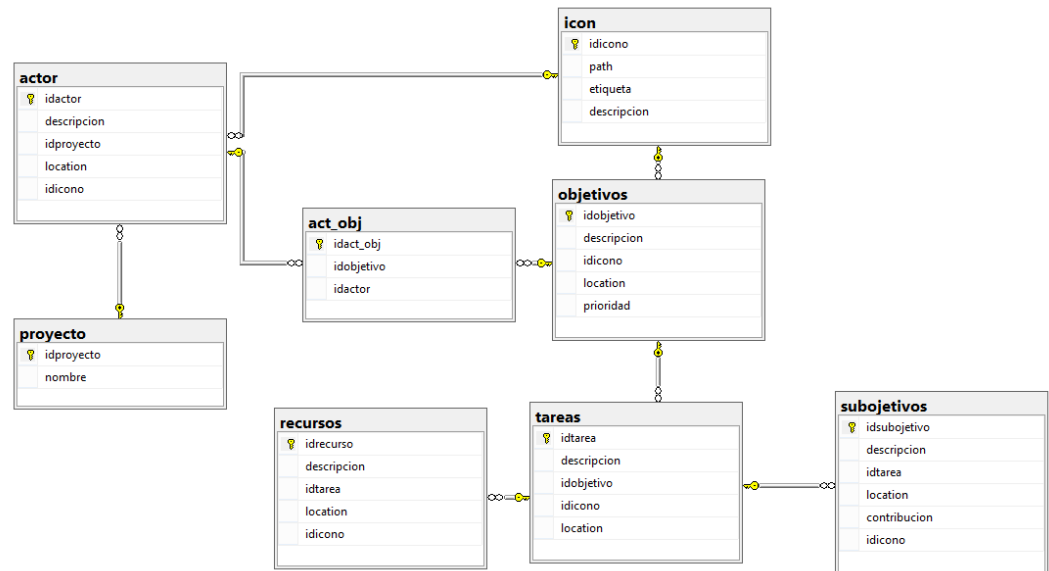
5.2 Codificación de las Historias de Usuario

Luego de definir los elementos que constituyen la herramienta MOOTRACE y las soluciones tecnológicas que serán necesarias para implementarla. A continuación, se precisa el desarrollo de cada uno de sus elementos.

El modelamiento de la base de datos se basó en el marco de trabajo del Modelado Orientado a Objetivos el cual se visualiza en Figura 38.

Figura 38

Modelado de la Base de Datos MOOTRACE.



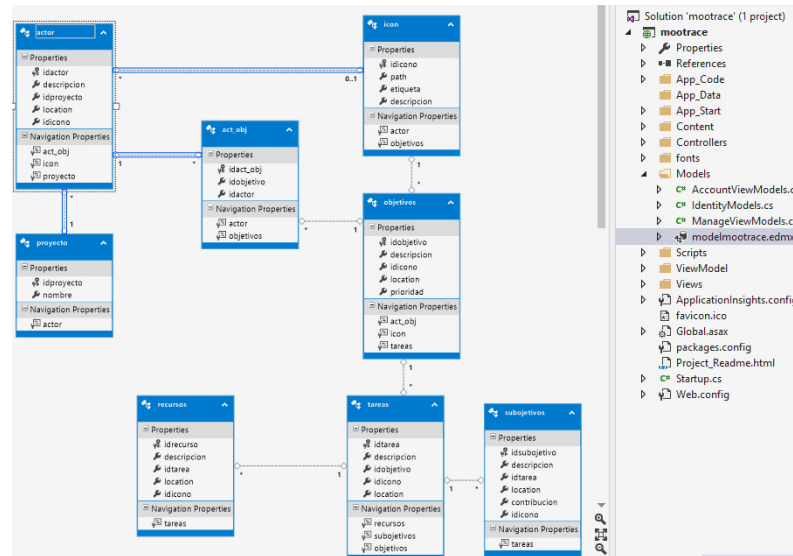
Respecto, al código fuente se realizó bajo el enfoque del patrón de arquitectura de software MVC de acuerdo con el siguiente detalle:

5.2.1 El Modelo

D. Lucero, trabajó con el sistema de Mapeo Objeto-Relacional denominado Entity Framework, e implementó el esquema Database First, haciendo uso del IDE de Visual Studio, mediante el cual se conectó a la base de datos generada en el punto anterior para luego transformarla en el modelo conceptual de la aplicación, la extensión del fichero generado después de este proceso es .edmx el cual contiene todos los objetos de la base de datos que se requieran mapear. Luego, se pulsa en la carpeta Models con clic derecho para agregar un nuevo elemento en la ventana, posteriormente seleccionamos la pestaña Datos para poder visualizar el elemento ADO.Net Entity Data Model, nombrando al fichero como modelmootrace y finalmente, se completa las configuraciones solicitadas teniendo como resultado el modelo que se visualiza en la figura 39.

Figura 39

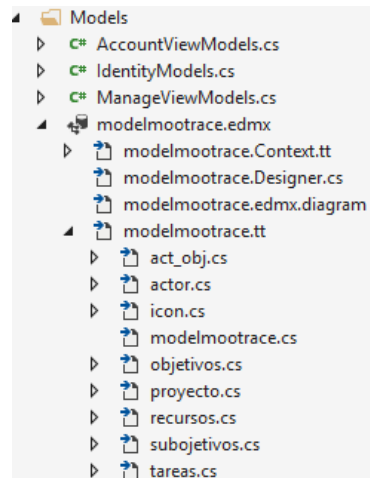
Arquitectura Conceptual



En el fichero del modelo generado y denominado modelmootrace.edmx, se visualizan dos ficheros con extensión *.tt como se observa en la Figura 40, la primera Context.tt a través de la cual establecemos la conexión a la base de datos; en la segunda clase con extensión *.tt se despliegan las clases tablas que han sido abstraídas desde la base de datos.

Figura 40

Modelo Conceptual

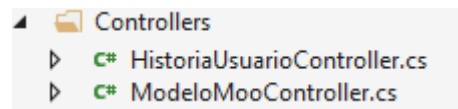


5.2.2 El Controlador

Se creó 2 archivos tipo Controller, los cuales solicitan los datos al modelo para ser mostrados en la vista; Get y Post son métodos del protocolo HTTP que permiten la comunicación entre el cliente y el servidor en el entorno web.

Figura 41

Controladores MOOTRACE



El controlador denominado **ModeloMooController.cs** es el encargado de atender de acuerdo las solicitudes realizadas en el modelo gráfico y cualquiera de las acciones que se puedan realizar en su interfaz para canalizarlas entre el modelo y la vista (D. Lucero, 2018).

El controlador denominado **HistoriaUsuarioController.cs** es el encargado de atender de acuerdo las solicitudes realizadas en la Historia de Usuario y cualquiera de las

acciones que se puedan realizar en su interfaz para canalizarlas entre el modelo y la vista (D. Lucero, 2018).

5.2.3 La Vista

Para el Editor Gráfico, se usó la librería go.js la cual finalmente representa una extensión de estas tecnologías, y que cubre con las expectativas para la creación de un entorno de modelado de requisitos resultado del cuadro análisis de librerías gráficas disponibles en el mercado para nuestro propósito.

Es preciso señalar que el uso del IDE ASP.NET MVC, genera en automático la carpeta de las vistas a partir de la creación del controlador las cuales mostrarán datos al cliente web, cabe destacar que en las Vistas del ModeloMoo se hace referencia a las librerías gráficas go.js y go-debug.js, para luego personalizarlas con los estereotipos del MOO.

GoJS es una biblioteca en lenguaje de programación JavaScript que permite crear diagramas interactivos en navegadores web modernos.

GoJS admite plantillas gráficas y enlace de datos de propiedades de objetos gráficos para la creación y edición de modelos.

Estos diagramas permitirán representar de forma gráfica las funcionalidades del sistema requeridas por el usuario establecidas en la etapa de elicitación con un lenguaje natural que forma parte del framework i star a través de sus estereotipos los cuales permiten un entendimiento sencillo, en el ámbito colaborativo del enfoque ágil resultando esto resulta muy productivo.

Estas representaciones gráficas (modelo), permiten caracterizar la solución a implementar, por lo tanto, se pueden analizar y evaluar con el fin de cumplir las expectativas del cliente.

Los diagramas corresponden a un modelo que contiene e interpreta los datos desde la interfaz de la aplicación para determinar las relaciones de enlace de nodo a nodo y las relaciones de grupo-miembro para ser almacenadas, actualizadas o eliminadas en la base de datos. El diagrama crea automáticamente un nodo o un grupo para cada elemento de datos en el *Model.nodeDataArray* del modelo y un enlace para cada elemento de datos en el *GraphLinksModel.linkDataArray* del modelo.

Tanto los objetos de tipo *nodo* o *enlace* son personalizables a través de las propiedades disponibles para cada objeto.

La visión global de los actores y sus tareas pueden también ser nutridas por otros que son parte del proceso o aportan durante su desarrollo, ya que tendrán en una sola vista los actores y tareas con las relaciones propias del proceso de negocio.

1. Cargando GoJS

El uso de librerías java script es fácil de referenciar en una página html, sólo basta con utilizar el tag `<script> </script>` para insertar la referencia y empezar a usarla, así como se muestra en la Figura 42.

Figura 42

Referencia a librería gráfica Go.js

```
<script src="~/Content/js/go.js"></script>
<script src="~/Content/js/go-debug.js"></script>
```

2. Alojar GoJS en un elemento `<div></div>`

GoJS administrará el contenido de ese elemento div, pero puede ubicar, clasificar y diseñar la **div** como lo haría con cualquier elemento HTML.

El diagrama agregará un elemento Canvas a ese elemento div en el que se dibujará el diagrama: esto es lo que los usuarios realmente ven.

El elemento Canvas se dimensiona automáticamente para que tenga el mismo tamaño que el elemento Div.

3. Personalización de la Caja de Herramientas de MOOTRACE

GoJS a través de su propiedad Palette, graficamos los iconos que representan los estereotipos de la metodología Orientada a Objetivos.

Figura 43

Creación de la Caja de Herramientas

```
myPalette = $(go.Palette, "myPaletteDiv",
{
    maxSelectionCount: 1,
    model: new go.GraphLinksModel(listiconos)
});
```

4. Configuración de los nodos

Figura 44 *Referencia a librería gráfica Go.js*

```
myPalette.nodeTemplate = $(go.Node, "Spot",
{
    locationSpot: go.Spot.Center,
    new go.Binding("location", "loc", go.Point.parse),
    { selectable: true, selectionAdornmentTemplate: nodeSelectionAdornmentTemplate },
    $(go.Panel, "Auto",
    {
        name: "PANEL",
        new go.Binding("desiredSize", "size", go.Size.parse).makeTwoWay(go.Size.stringify),
        $(go.Shape, "Ellipse",
        {
            width: 90, height: 90, stroke: "white", fill: "white",
            new go.Binding("fill", "color"),
        },
        $(go.TextBlock, {
            alignment: new go.Spot(0.5, 0),
            {
                text: "etiqueta", editable: true, stroke: "red", width: 78, wrap: go.TextBlock.WrapFit, height: 70, font: "bold 8pt serif",
                new go.Binding("text", "etiqueta").makeTwoWay(),
            },
            $(go.Picture, {
                alignment: new go.Spot(0.5, 1),
                {
                    width: 45, height: 45, new go.Binding("source", "path"),
                    {
                        tooltip:
                        $(go.Adornment, "Auto",
                        $(go.Shape, { fill: "#FFFFCC" }),
                        $(go.TextBlock, {
                            margin: 4, stroke: "gray", font: "bold 10pt serif", new go.Binding("text", "descripcion")
                        })
                    )
                }
            })
        })
    })
});
```

Figura 45 *Resultado en la vista final*



Cada diagrama GoJS está contenido en un elemento HTML <div> en su página HTML que le asigna un tamaño explícito:

Figura 46

Vistas MOOTRACE

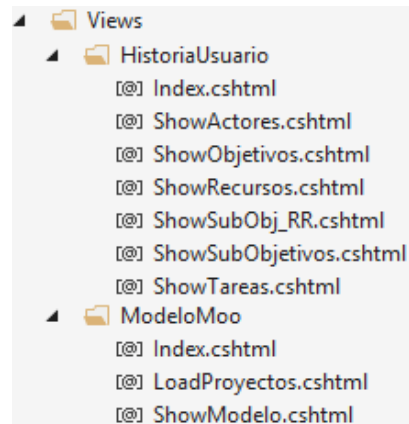


Figura 47

Creación del Modelo Gráfico

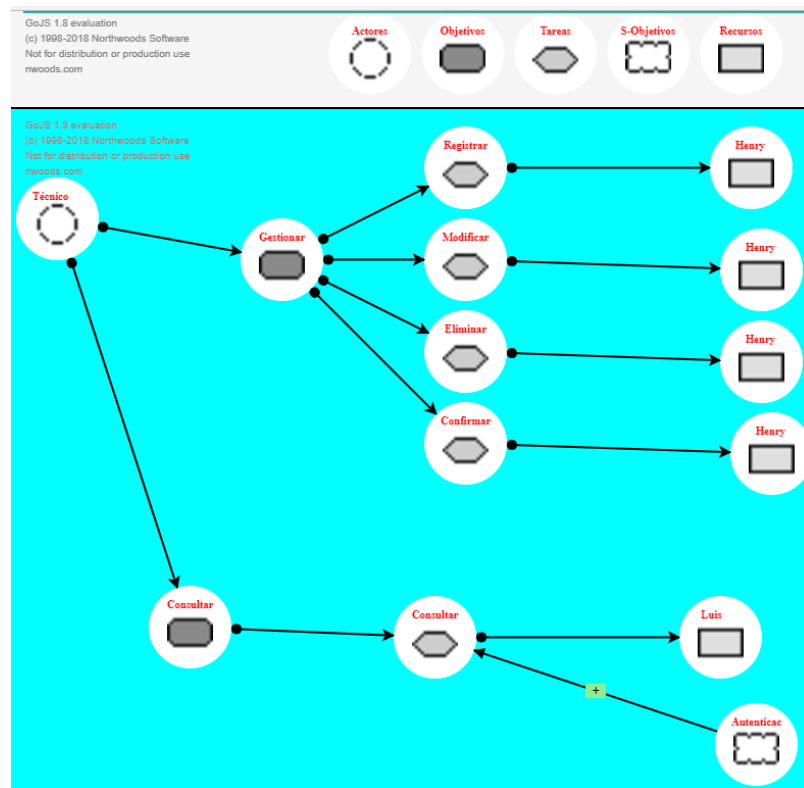


Figura 48

Automatización de las Historias de Usuario

NECESITO	
Objetivos	Prioridad
Consultar los pedidos	5
<button>update</button>	

PARA	
1	Consultar

TRAZABILIDAD	
CRITERIOS DE ACEPTACIÓN	
1	Autenticación
RECURSOS	
1	Luis Vázquez

5.3 Resumen del capítulo

En este capítulo se ha detallado el proceso de desarrollo de la herramienta propuesta siguiendo las fases de la metodología XP y con el framework de desarrollo MVC. Se estableció los requisitos tecnológicos necesarios para el desarrollo del Editor Gráfico. Luego se empezaron a implementar/codificar las historias de usuario. En la primera iteración se realizó el desarrollo del Diagrama del modelo de especificación de requisitos para representar el modelado orientado a objetivos de los requisitos. Se desarrollaron las personalizaciones de los nodos para el editor gráfico, así como sus relaciones y contribuciones, en ese sentido, se codificaron los 3 componentes parte de la solución y de acuerdo con las Historias de Usuario elaboradas en el capítulo anterior, según el siguiente detalle:

El primer componente: Modelado de requisitos, que define una interfaz mediante la cual el usuario final y el equipo de proyecto pueden graficar a través de los estereotipos establecido en la metodología orientada a objetivos los actores, tareas, recursos, subjetivos y relaciones entre ellos en un lenguaje natural sin perder el enfoque del objetivo principal.

En el segundo componente, visualizamos las tarjetas de historia de usuario generadas automáticamente del componente anterior, respetando las relaciones establecidas por el equipo de análisis de requisitos y el usuario. Además, se puede incorporar datos adicionales de la historia de usuario que deberán ser completados en base a su experiencia y consenso con el equipo de desarrollo por ejemplo asignar la prioridad en el desarrollo de las historias de usuario, adicionar prototipos, ampliar la definición de las tareas y restricciones.

El tercer componente referido a la gestión de la trazabilidad del requisito, que implícitamente ha sido alimentado desde la etapa inicial de la especificación del requisito en el modelo gráfico y que finalmente se consolida con los criterios técnicos que ingresa el programador en la Historia de Usuario. Este ciclo interactivo que incorpora trazabilidad se da por terminado con la asignación del estado final de la HU: “Terminada” o “No terminada”

Finalmente, se obtiene el modelo de requisitos orientado a objetos bajo el enfoque de los estereotipos definidos en el framework i*, a través del cual permite representar gráficamente sus actores, objetivos, tareas, recursos, subobjetivos y relaciones entre ellos. Este entorno proporciona al usuario un lenguaje natural para reproducir sus necesidades y el valor que poseen para ellos.

6 CAPÍTULO: RESULTADOS Y DISCUSIÓN

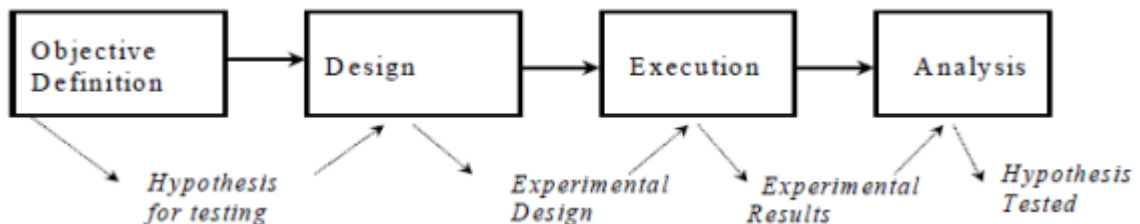
En el presente capítulo se explicará los resultados de los ensayos ejecutados con el fin de comprobar si la hipótesis causal “La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad mejora la fase de captura de requisitos en el proceso de desarrollo de software” es válida. Un experimento es un método por el cual una variable independiente es manipulada para conocer sus efectos y con ellos sustentar una hipótesis causal (Vara-Horna y Arístides, 2012), porque el margen de certeza y autenticidad debido a su enfoque empírico. Se puede concluir que los experimentos demostraron la mejora en la calidad del requisito y la reducción del tiempo en la transformación a funcionalidades.

6.1 Análisis, interpretación y discusión de resultados

El experimento se diseñó de acuerdo con el proceso definido por (Juristo y Moreno, 2010), en la Figura 49 se visualiza las 4 fases para del Diseño Experimental en software que establecieron estos autores: Definición de objetivos, Diseño, Ejecución y Análisis del resultado.

Figura 49

Proceso del Diseño Experimental en Software.



Nota. Tomado de *Basics of Software Engineering Experimentation*, por Juristo, N. y Moreno, A., 2010.

6.1.1 Definición del Objetivo del Experimento

El objetivo de este experimento consistió en comparar la especificación de requisitos realizada con una herramienta gráfica y la herramienta gráfica MOOTRACE bajo enfoque tradicional y ágil respectivamente. Su propósito estaba en demostrar las relaciones de efectividad entre las técnicas de elicitación utilizadas y su efecto en la calidad del requisito completado.

Las pruebas cuantificaron la duración y dificultad en la usabilidad de 2 herramientas (MOOTRACE y la herramienta CASE para modelado con UML Rational Rose) por parte de los usuarios para definir exactamente los mismos requisitos.

Los proyectos que fueron evaluados como parte del experimento son de diversa semejanza y serán de uso en las áreas de Dirección del Material, Gestión de Inventarios, Dirección de Recursos Humanos, etc.

Para el desarrollo del experimento participaron profesionales con el rol de analistas, desarrolladores y usuarios de las áreas solicitantes.

6.1.2 Diseño del Experimento

Los experimentos tienen elementos indispensables sobre los que se ejecutan acciones para un tener diseño adecuado. Vara (2010) define 4: Manipulación de la variable independiente, Medición de la variable dependiente, Asignación aleatorizada para controlar las variables extrañas y, Grupo de Control y Experimental para determinar empíricamente la validez de la Hipótesis en cuestión.

La medición de las Variables Dependientes (VD) en el proceso de especificación de requisitos se desarrolló en 2 etapas: En el Grupo de Control (GC) con una herramienta gráfica y en el Grupo Experimental (GE) que manipuló la Variable Independiente (VI) MOOTRACE (herramienta gráfica creada). La Asignación de los integrantes de los 2 grupos se realizó de manera aleatoria.

En la Figura 50, se asigna nombre a cada uno de los elementos entorno al contexto del tema investigado.

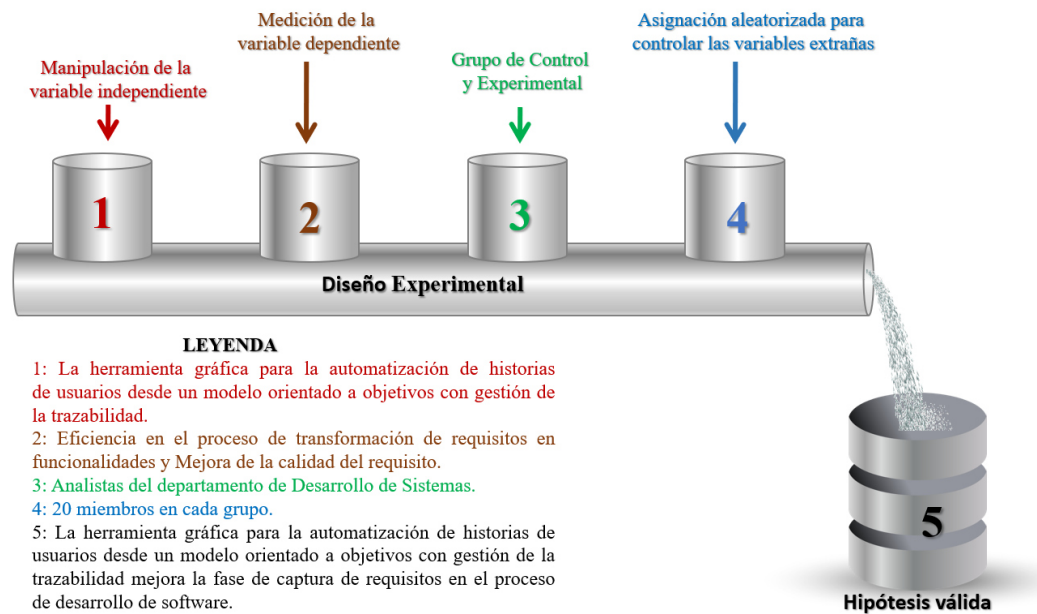
VI: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad.

VD: Eficiencia en el proceso de transformación de requisitos en funcionalidades y Calidad del requisito.

GC / GE: Analistas del departamento de Desarrollo de Sistemas.

Figura 50

Elementos de un experimento



.Nota. Adaptado Desde la idea hasta la sustentación p.211 - 212, por Vara, 2010.

Asimismo, Valverde (2014) define los Parámetros como las condiciones inalterables en las que se debe de realizar un experimento con la finalidad de no afectar los resultados, para este experimento fueron:

1. Cada participante recibió capacitación en el uso de la herramienta gráfica MOOTRACE.
2. Cada participante recibió capacitación en el uso de la herramienta gráfica UML Rational Rose.

3. Cada participante cuenta con experiencia en análisis entre 5 y 8 años.
4. Las computadoras deben tener acceso a internet.

Asimismo, se consideró a las Variables de respuesta del experimento son las siguientes:

1. Eficiencia en el proceso de transformación de un requisito a funcionalidad:
Esfuerzo que toma al sistema desde que el analista empieza a elaborar el modelo gráfico hasta que se convierte en un requisito de sistema.
2. Calidad del requisito, cuyo nivel se establece a través de las siguientes métricas:
 - a. Especificidad de los requisitos.
 - b. Completitud de los requisitos.
 - c. Validez de los requisitos.

Finalmente, la Unidad Experimental de esta investigación se seleccionaron 8 proyectos desarrollados en el año 2019 en el área de Desarrollo de Sistemas:

1. Proyecto de Trámites del Ciudadano – TUPA:

El Sistema de Trámites del Ciudadano – TUPA, permitirá al ciudadano a través de una plataforma accederá a las tasas estipuladas para cada trámite, luego de ello iniciará el trámite mediante la digitalización de su formato el cual será remitido a la mesa de partes, y en el módulo de seguimiento podrá visualizar el estado de su documento y descargar la respuesta a su trámite por la misma plataforma.

2. Proyecto de Bienes Patrimoniales

El Sistema de Bienes Patrimoniales, permitirá registrar los bienes, y clasificarlos de acuerdo con lo establecido en la directiva interna de la institución, para que los reportes anuales permitan mostrar los datos actualizados y se tomen las decisiones y acciones correspondientes a la vigencia de un bien.

3. Proyecto Planificador de Sala de Reuniones

El Sistema Planificador de Sala de Reuniones, permitirá a los usuarios registrar, notificar y calendarizar los eventos en los que se haga uso del ambiente físico y los implementos audiovisuales que se necesiten para su realización.

4. Proyecto de Calificación de Desempeño Profesional

El Sistema para la Calificación de Desempeño Profesional, permitirá a los usuarios jefes registrar la valoración en escala vigesimal, de acuerdo con las áreas técnicas en las que se

desempeña un empleado a su cargo. Asimismo, se contempla la creación de un web service que entrega la información generada al Sistema de Foja de Notas.

5. Proyecto Gestión Archivística

El Sistema para la Gestión Archivística, permitirá a los usuarios integrar la documentación digitalizada proveída por las diferentes áreas, a fin de determinar dentro de sus competencias su almacenamiento perpetuo u eliminación.

6. Proyecto Emisión Tarjeta Entidad Laboral

El Sistema para la Emisión Tarjeta Entidad Laboral, permitirá integrar tecnología en las tarjetas, a fin de incorporar información en el chip acerca de las sanciones, accesos restringidos, caducidad de la misma y control del personal en servicio.

7. Proyecto de Control de Citas

El Sistema para el Control de Citas, permitirá a los usuarios asegurados titulares y familiares separar una cita desde la comodidad de su móvil (podrá seleccionar horario y médico tratante), el sistema notificará la cita, y emitirá la boleta de costo por la misma de ser el caso que sea un familiar.

8. Proyecto de Índice de Masa Corporal

El Sistema de Índice de Masa Corporal, permitirá al usuario encargado de la evaluación, registrar el peso para visualizar el rango en el que se encuentra en el paciente, en función a los resultados se determinará las acciones a seguir como: planes alimenticios, derivación al médico especialista si los índices presentan escala de morbilidad. Asimismo, el sistema generará reportes y establecerá la calendarización de controles de índice de masa corporal y el cual será notificado al celular y correo electrónico del paciente, según corresponda.

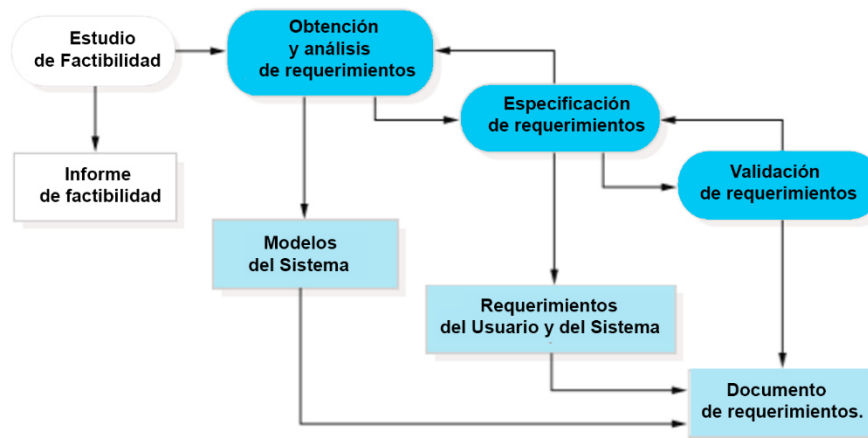
6.1.3 Ejecución

- a) El Experimento para medir la Eficiencia en el proceso de transformación de un requisito a funcionalidad se realizó en 3 etapas de las 4 definidas por (Bourque y Fairley, 2014; Sommerville, 2011) en el Desarrollo de un requisito las cuales se enfocan en elicitar requisitos (adquisición y análisis), transformarlos en alguna forma estándar (especificación) y verificar que cumpla con las expectativas del cliente (validación).

La Figura 51, muestra la secuencia para el desarrollo del requerimiento. Los experimentos se realizaron bajo la presunción que la Fase de Estudio de factibilidad ya ha sido completada satisfactoriamente.

Figura 51

Proceso de Desarrollo de Requisitos en la ingeniería de requisitos.



Nota. Tomado de *Ingeniería de Software*, por Sommerville, 2011.

El esfuerzo que demandó el desarrollo de los requisitos para el Sistema de Trámites al Ciudadano y posterior implementación se ejecutó de la siguiente manera:

En la fase de Adquisición y Análisis el Grupo de Control usó el artefacto Diagrama de Caso de Uso usando notación UML y el Grupo Experimental el Modelo gráfico en MOOTRACE. En la fase de Especificación el Grupo de Control y el Grupo Experimental, Casos de uso del Sistema y Modelo gráfico en MOOTRACE respectivamente. Finalmente, en la fase de Validación los Casos de Prueba y la Historia de Usuario en MOOTRACE sirvieron al Grupo de Control y al Grupo Experimental para determinar la materialización de los requisitos que el cliente espera del sistema. Si el diseño de un requisito es irrealizable, su implementación en la aplicación es poco probable. Al detectar un error, inconsistencia, ambigüedad retornamos a la primera fase a fin de corregir la observación para esa iteración.

Para las actividades en la fase de Adquisición y Análisis, el Grupo de Control diseñó los Casos de Uso con la lista de actores identificados y la información recopilada y priorizada de las entrevistas con ellos, del otro lado el Grupo Experimental se reunió con los usuarios con quienes en conjunto definieron la lista de actores, objetivos en un modelo gráfico de la herramienta gráfica en MOOTRACE.

En la fase de Especificación, el Grupo de Control definió en un documento las interacciones y eventos de los Diagramas de Casos de Uso generados en la fase anterior, además para definir los Requisitos No Funcionales (RNF) crearon un documento adicional llamado Especificaciones Suplementarias. Sin embargo, el Grupo Experimental optimizó el mismo Modelo Gráfico de la plataforma MOOTRACE creado en la fase anterior, agregando las relaciones y estereotipos como softgoals (que representan los RNF), recursos y la contribución de los softgoals para especificar su relevancia sobre los otros elementos del modelo (objetivos, tareas, recursos y contribuciones). Cabe destacar, que el usuario en esta fase continuó participando en el refinamiento del modelo gráfico.

Finalmente, en la fase de Validación el grupo de control debió elaborar un nuevo documento llamado Caso de Prueba para cada una de las especificaciones anteriores. En la misma circunstancia el Grupo de Ejecución, generó en simultaneo las Historias de Usuario de los requisitos definidos para cada proyecto.

Cabe considerar, que el Grupo de Control tuvo que utilizar 3 formatos distintos en cada fase ocasionando mayor esfuerzo no sólo en la elaboración de cada artefacto, sino, que además existiendo el riesgo que la transferencia de información entre uno y otra sufra alguna alteración.

En la tabla 17, se realizó un consolidado considerando los artefactos, tareas y tiempo transcurrido que utilizó el Grupo de Control y el Grupo de Ejecución durante el experimento en las 3 fases, la cual muestra una comparación respecto al tiempo de ejecución en el proceso de ingeniería de requisitos con la herramienta gráfica UML que es la más usada en entornos tradicionales versus la utilización de la herramienta MOOTRACE construida en la que se establece que durante la ejecución de la herramienta MOOTRACE reduce los tiempos y que además se trabajó con las misma herramienta durante todo el proceso. En la fase de verificación es más notoria debido a que se ejecuta

en simultáneo favoreciendo la utilización de herramienta MOOTRACE optimizando el costo de tiempo y esfuerzo.

Asimismo, atendiendo a los requisitos de los 8 proyectos obtuvimos la información de la ejecución del experimento en cada uno de ellos según se detalla a continuación:

Tabla 17: Tiempo de transformación de requisitos a funcionalidades del Proyecto de Trámites del Ciudadano.

Tabla 18: Tiempo de transformación de requisitos a funcionalidades del Proyecto de Bienes Patrimoniales.

Tabla 19: Tiempo de transformación de requisitos a funcionalidades del Proyecto Planificador de Sala de Reuniones.

Tabla 20: Tiempo de transformación de requisitos a funcionalidades del Proyecto de Calificación de Desempeño Profesional.

Tabla 21: Tiempo de transformación de requisitos a funcionalidades del Proyecto Gestión Archivística.

Tabla 22: Tiempo de transformación de requisitos a funcionalidades del Proyecto Emisión Tarjeta Entidad Laboral.

Tabla 23: Tiempo de transformación de requisitos a funcionalidades del Proyecto de Control de Citas.

Tabla 24: Tiempo de transformación de requisitos a funcionalidades del Proyecto de Índice de Masa Corporal.

Tabla 17

Tiempo de transformación de requisitos a funcionalidades del Proyecto de Trámites del Ciudadano.

TAREA \ FASE \ ARTEFACTO	Adquisición y Análisis		Especificación		Validación	
	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Crear un nuevo Tipo de Pago	10 min.	8 min.	12 min.	10 min.	15 min.	0.12 min.
Asociar FUT con Tipo de Pago	12 min.	12 min.	14 min.	8 min.	15 min.	
Generar Reporte Pago	12 min.	10 min.	14 min.	6 min.	15 min.	
TIEMPO PARCIAL POR FASE	34 min.	30 min.	40 min.	24 min.	45 min.	0.12 min.

Tabla 18

Tiempo de transformación de requisitos a funcionalidades del Proyecto de Bienes Patrimoniales.

FASE		Adquisición y Análisis		Especificación		Validación	
TAREA	ARTEFACTO	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
	Reasignación de bien	10	8 min.	18 min.	10 min.	20 min.	0.11 min.
	Consultar existencia en Unidades	8	5 min.	10 min.	5 min.	15 min.	
	Generar de Reporte de Reasignación	6	5 min.	10 min.	6 min.	8 min.	
	TIEMPO PARCIAL POR FASE	24 min.	18 min.	39 min.	21 min.	43 min.	0.11 min.

Tabla 19

Tiempo de transformación de requisitos a funcionalidades del Proyecto Planificador de Sala de Reuniones.

TAREA \ FASE \ ARTEFACTO	Adquisición y Análisis		Especificación		Validación	
	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Consultar Disponibilidad Sala	5 min.	4 min.	10 min.	5 min.	15 min.	0.08 min.
Crear Evento	8 min.	5 min.	12 min.	4 min.	9 min.	
Agregar asistentes	8 min.	5 min.	12 min.	4 min.	7 min.	
Notificar Asistentes	5 min.	2 min.	7 min.	4 min.	8 min.	
TIEMPO PARCIAL POR FASE	26 min.	16 min.	41 min.	17 min.	39 min.	0.08 min.

Tabla 20

Tiempo de transformación de requisitos a funcionalidades del Proyecto de Calificación de Desempeño Profesional

TAREA \ FASE \ ARTEFACTO	Adquisición y Análisis		Especificación		Validación	
	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Crear Tabla de Pesos ponderados para Evaluación	8 min.	6 min.	10 min.	6 min.	11 min.	0.05 min.
Registrar de Especialización	6 min.	4 min.	9 min.	5 min.	10 min.	
Realizar Evaluación	8 min.	6 min.	6 min.	4 min.	8 min.	
TIEMPO PARCIAL POR FASE	22 min.	16 min.	25 min.	15 min.	29 min.	0.05 min.

Tabla 21

Tiempo de transformación de requisitos a funcionalidades del Proyecto Gestión Archivística.

TAREA \ FASE	Adquisición y Análisis		Especificación		Validación	
	Artefacto	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Crear procedencia Tipo de Archivo	7 min.	5 min.	10 min.	4 min.	12 min.	0.05 min.
Registrar de Documento Histórico	6 min.	4 min.	9 min.	3 min.	10 min.	
Generar Reportes de Documentos	5 min.	4 min.	7 min.	3 min.	8 min.	
TIEMPO PARCIAL POR FASE	18 min.	13 min.	26 min.	10 min.	30 min.	0.05 min.

Tabla 22

Tiempo de transformación de requisitos a funcionalidades del Proyecto Emisión Tarjeta Entidad Laboral.

TAREA \ FASE \ ARTEFACTO	Adquisición y Análisis		Especificación		Validación	
	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Crear Tipo Tarjeta	6 min.	5 min.	9 min.	5 min.	12 min.	0.13 min.
Consultar estado del solicitante	8 min.	4 min.	12 min.	8 min.	14 min.	
Imprimir Tarjeta de Identidad	6 min.	4 min.	9 min.	4 min.	11 min.	
Generar Reporte de Renovación Tarjetas	6 min.	4 min.	9 min.	4 min.	11 min.	
TIEMPO PARCIAL POR FASE	26 min.	17 min.	39 min.	21 min.	48 min.	0.13 min.

Tabla 23

Tiempo de transformación de requisitos a funcionalidades del Proyecto de Control de Citas.

TAREA \ FASE \ ARTEFACTO	Adquisición y Análisis		Especificación		Validación	
	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Consultar Disponibilidad de horario	8 min.	5 min.	10 min.	3 min.	12 min.	0.05 min.
Registrar Cita	5 min.	2 min.	8 min.	1 min.	10 min.	
Emitir Comprobante	3 min.	2 min.	5 min.	1 min.	7 min.	
TIEMPO PARCIAL POR FASE	16 min.	9 min.	23 min.	5 min.	29 min.	0.05 min.

Tabla 24

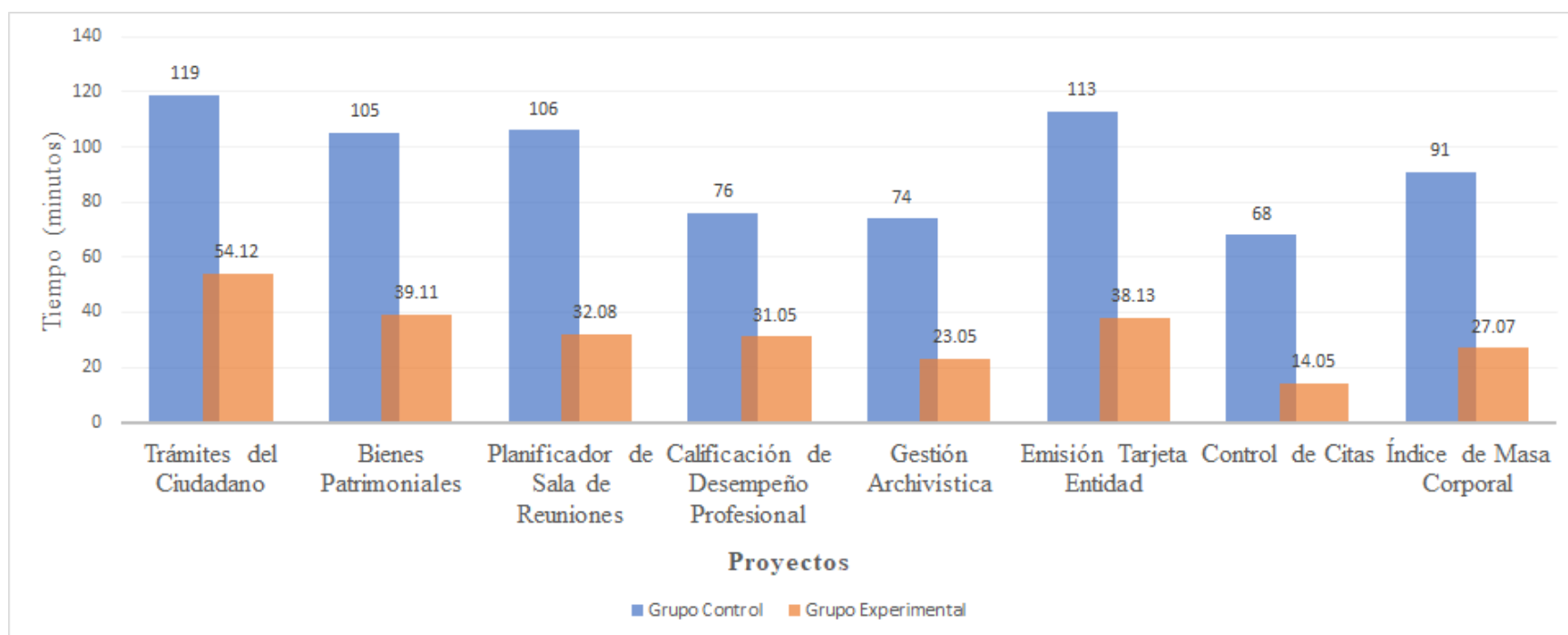
Tiempo de transformación de requisitos a funcionalidades del Proyecto de Índice de Masa Corporal.

TAREA \ FASE \ ARTEFACTO	Adquisición y Análisis		Especificación		Validación	
	Diagrama de Caso de Uso con notación UML	Modelo gráfico en MOOTRACE	Casos de uso del Sistema con notación	Modelo gráfico en MOOTRACE	Casos de Prueba	Historia de Usuario en MOOTRACE
Consultar estado de personal militar en la institución	8 min.	5 min.	10 min.	4 min.	12 min.	0.07 min.
Registrar Índice de Masa Corporal	9 min.	6 min.	12 min.	5 min.	14 min.	
Generar Reporte Personal por niveles de obesidad	6 min.	4 min.	9 min.	3 min.	11 min.	
TIEMPO PARCIAL POR FASE	23 min.	15 min.	31 min.	12 min.	37 min.	0.07 min.

Después de la ejecución del experimento, se realizó un consolidado con los tiempos promedio parciales por fase usando la información de la Tabla 16 a la Tabla 24 del para mostrar el esfuerzo total en cada proyecto por el Grupo Experimental y el Grupo de Control (véase Figura 52).

Figura 52

Tiempo del proceso de ingeniería de requisitos por fase en cada sistema.



- b) Para establecer la validez de la hipótesis específica 2 referida a la Calidad del Requisito se realizó el experimento con el Grupo de Control y el Grupo de Ejecución en paralelo aplicando las siguientes métricas (véase Tabla 25).

Tabla 25

Métricas de Calidad del requisito.

Métrica		Uso	Fórmula
M1	Especificidad de los requisitos.	Valor que representa una única interpretación para cada requerimiento por parte de los revisores.	$ER = \frac{Nui}{Nr}$ ER, es el nivel de especificidad de los requisitos; Nui, es el número de requisitos con una sola interpretación por parte de los revisores; Nr, es el número total de requisitos funcionales y no funcionales.
M2	Validez de los requisitos.	Valor que establece la validez de lo propuesto por el cliente y lo que el equipo de desarrollo puede realizar	$\frac{Nc}{Nc + Nnv}$ RV, es el grado de validación de los requisitos; Nc, es el número de requisitos validados como correctos; Nnv, es el número de requisitos no validados aún.

b.1) Pruebas de la M1: Especificidad de los requisitos.

Las conclusiones de las mediciones obtenidas de la ejecución del experimento se presentan en la Tabla 26, el rango para el valor de esta métrica se encuentra entre 0 y 1. Mientras más cerca del valor de 1 se encuentre el valor de la ER (Especificidad del Requerimiento) la interpretación de los revisores para cada requerimiento será más coherente y menor será la ambigüedad en la especificación de los requisitos de este modo los valores que corresponden a la especificación realizada en el modelo gráfico MOOTRACE así lo demuestran.

Tabla 26

Datos de cálculo de la M1 en los proyectos del experimento.

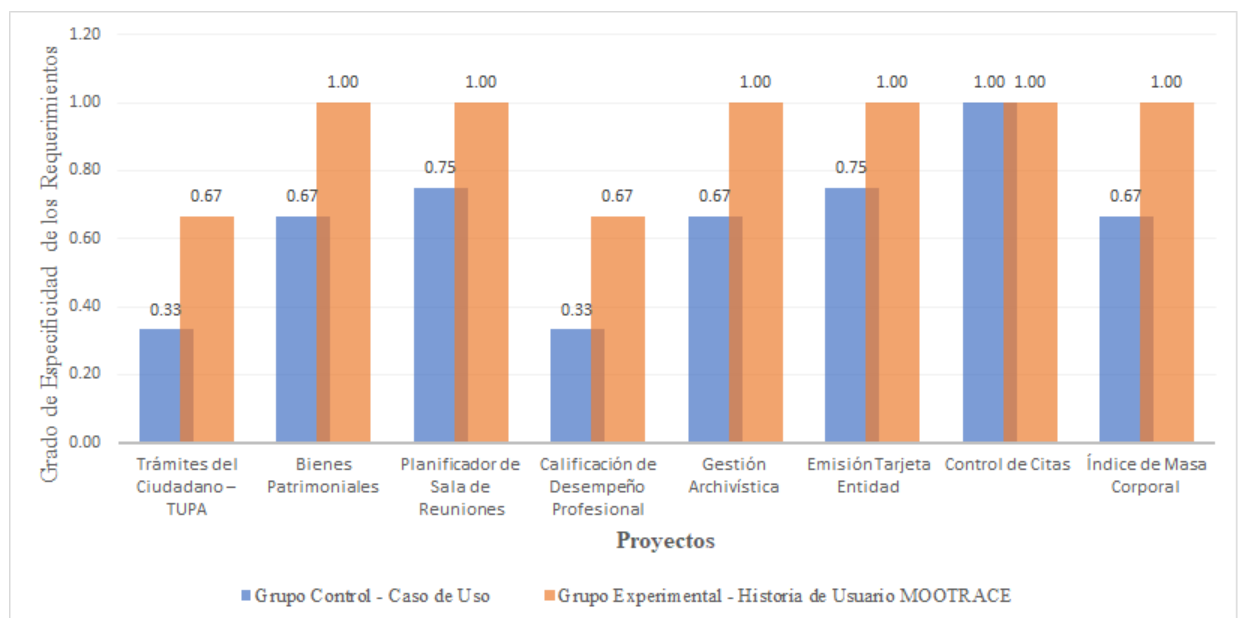
Proyecto	M1: Especificidad del requerimiento	
	GC: Especificación de Caso de Uso	GE: Modelo gráfico MOOTRACE
Trámites del Ciudadano – TUPA	0.33	0.67
Bienes Patrimoniales	0.67	1.00
Planificador de Sala de Reuniones	0.75	1.00
Calificación de Desempeño Profesional	0.33	0.67
Gestión Archivística	0.67	1.00
Emisión Tarjeta Entidad Laboral	0.75	1.00
Control de Citas	1.00	1.00
Índice de Masa Corporal	0.67	1.00

La Figura 53 muestra el comportamiento de la M1 en cada uno de los artefactos usados por cada grupo y permite calcular el grado de especificidad de requisitos (ER) de acuerdo con las conversiones semejantes que realizaron los analistas o ingenieros de requisitos.

En este caso, la métrica establece que la herramienta MOOTRACE permite a los analistas especificar con mayor precisión los requisitos expresados por los stakeholders, identificando las ambigüedades tempranamente para poder subsanarlas precisando lo que no haya quedado claro en esta etapa, evitando que se realicen tareas innecesarias o se desarrolle algo que el usuario no necesita. Si el significado de los requisitos es entendido por todos los usuarios e interesados, significa que las iteraciones del proceso son productivas y se está obteniendo adecuadamente los requisitos.

Figura 53

Métrica 1: Grado de Especificidad de los Requisitos.



b.2) Pruebas de la M2: Validez de los requisitos.

Los resultados obtenidos se presentan en la Tabla 27. En este caso, la métrica permite identificar que con el uso de la herramienta MOOTRACE el valor de la validez de los requisitos que se aprueban por el usuario es mayor, esto significa que los requisitos son claros y recogen todo aquello que el usuario espera del sistema que se desarrollará.

Tabla 27

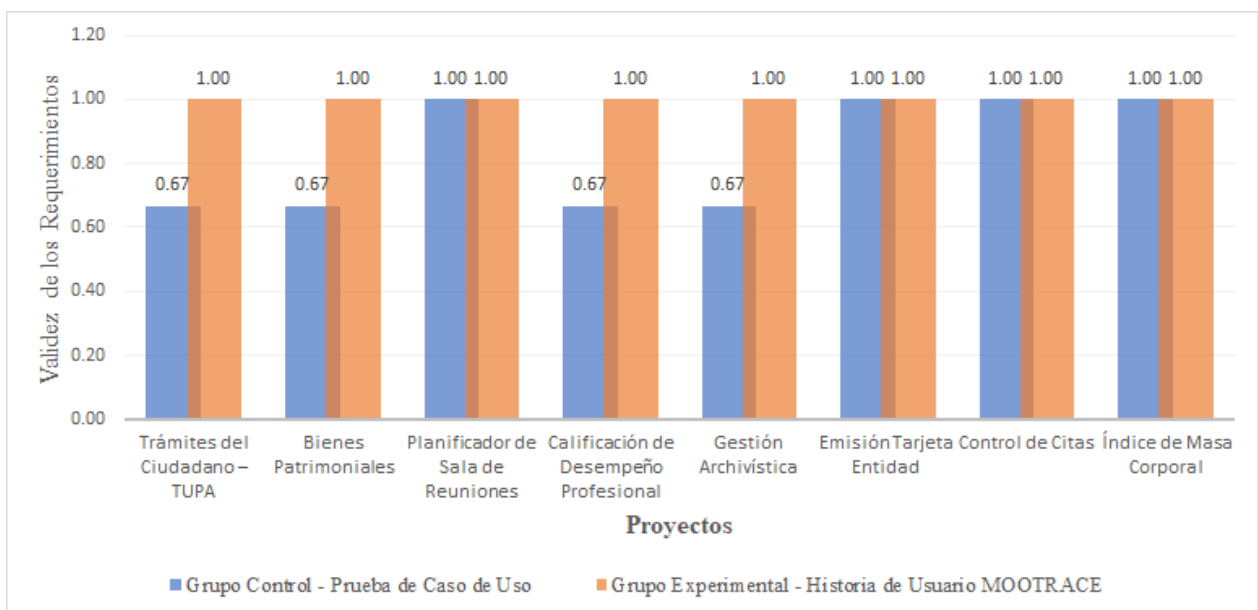
Datos de cálculo de la M2 en los proyectos del experimento.

Sistemas	M2: Validez del requerimiento	
	GC: Prueba de Caso de Uso	GE: Historia de Usuario MOOTRACE
Trámites del Ciudadano – TUPA	0.67	1.00
Bienes Patrimoniales	0.67	1.00
Planificador de Sala de Reuniones	1.00	1.00
Desempeño Profesional	0.67	1.00
Gestión Archivística	0.67	1.00
Emisión Tarjeta Entidad	1.00	1.00
Control de Citas	1.00	1.00
Índice de Masa Corporal	1.00	1.00

La Figura 54 muestra el comportamiento de la M2 en cada uno de los Grupos y permite calcular la validez de los Requisitos con el Grupo de Control en las Pruebas de Casos de Uso y por otro lado con el Grupo Experimental en las Historias de Usuario enriquecidas con la información que ha recibido del modelo gráfico en la fase anterior

Figura 54

Métrica 2: Grado de Validez de los Requisitos



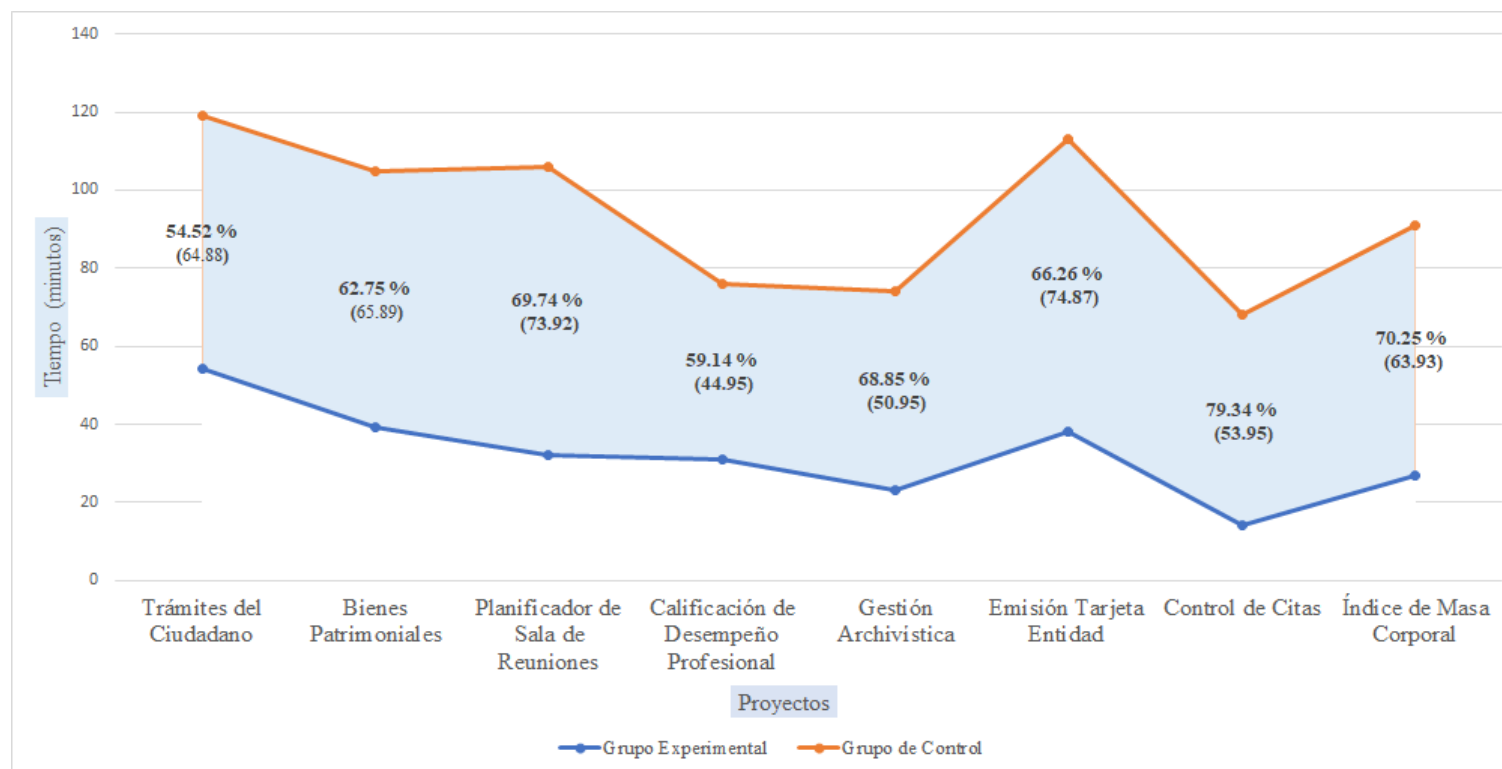
6.2 Pruebas de Hipótesis

6.2.1 Prueba de Hipótesis del Objetivo Específico 1

La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad reduce el tiempo de transformación de requisitos a funcionalidades. La medición del porcentaje de esfuerzo realizado (véase Figura 55) demuestra una reducción considerable 66.35 % en tiempo y eficacia en el proceso desarrollo del requisito en sus tres fases desde su captura hasta su validación.

Figura 55

Gráfico del Porcentaje de eficiencia en la ejecución del proceso de ingeniería de requisitos en el Grupo de Control y Experimental

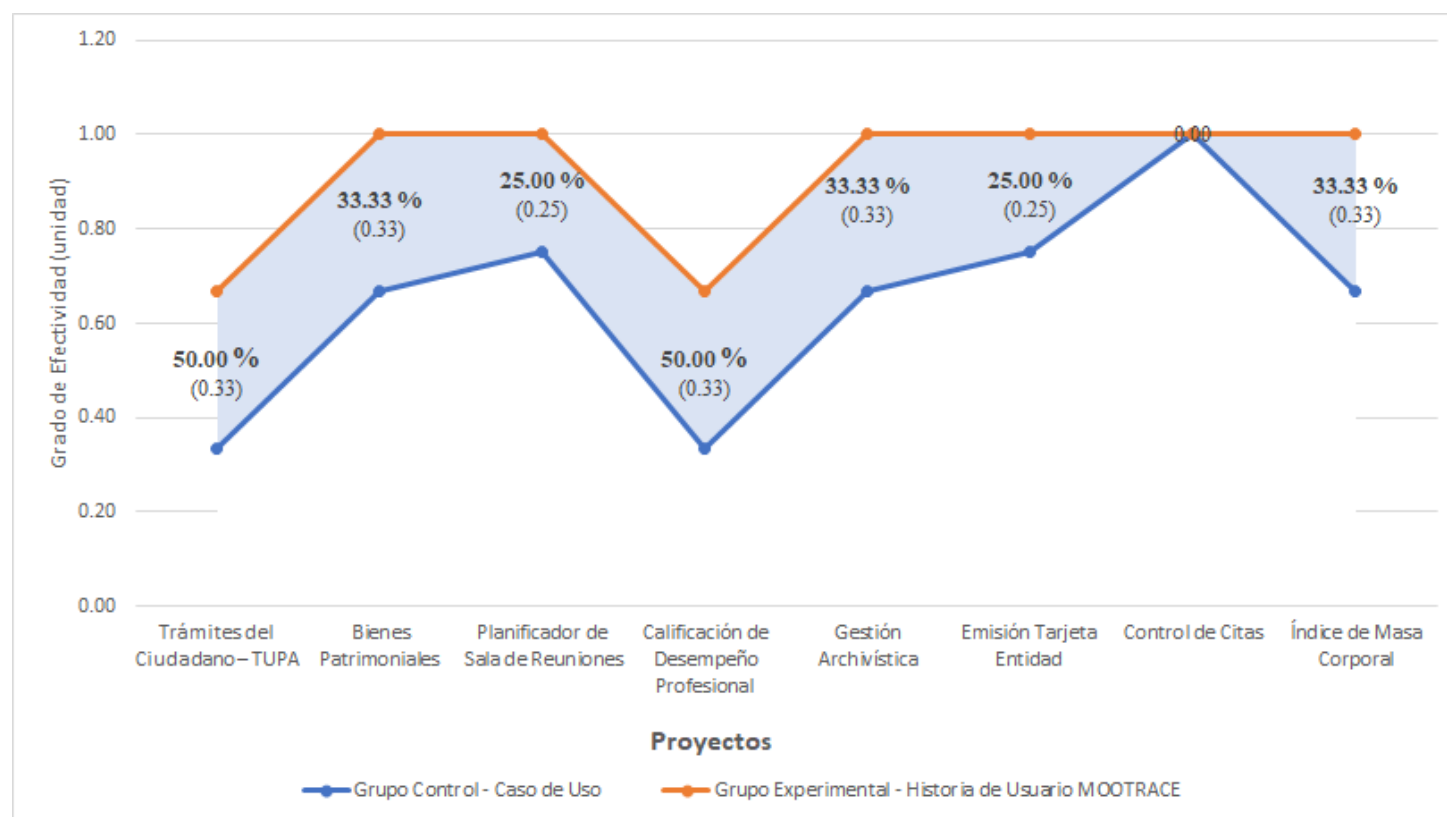


6.2.2 Prueba de Hipótesis del Objetivo Específico 2 Métrica 1

La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad incrementa la Calidad del Requisito. La Figura 56 muestra el comportamiento de la M1 que calcula el grado de **Especificidad del requisito**, el porcentaje de efectividad representa un 31,25% en promedio en los requisitos elicitados con Historias de Usuario es mayor frente a los especificados con Caso de Uso. En razón a ello, se infiere que los requisitos fueron entendidos adecuadamente.

Figura 56

Gráfico del Porcentaje de eficiencia en la ejecución de la Métrica 1 del Objetivo Específico 2.

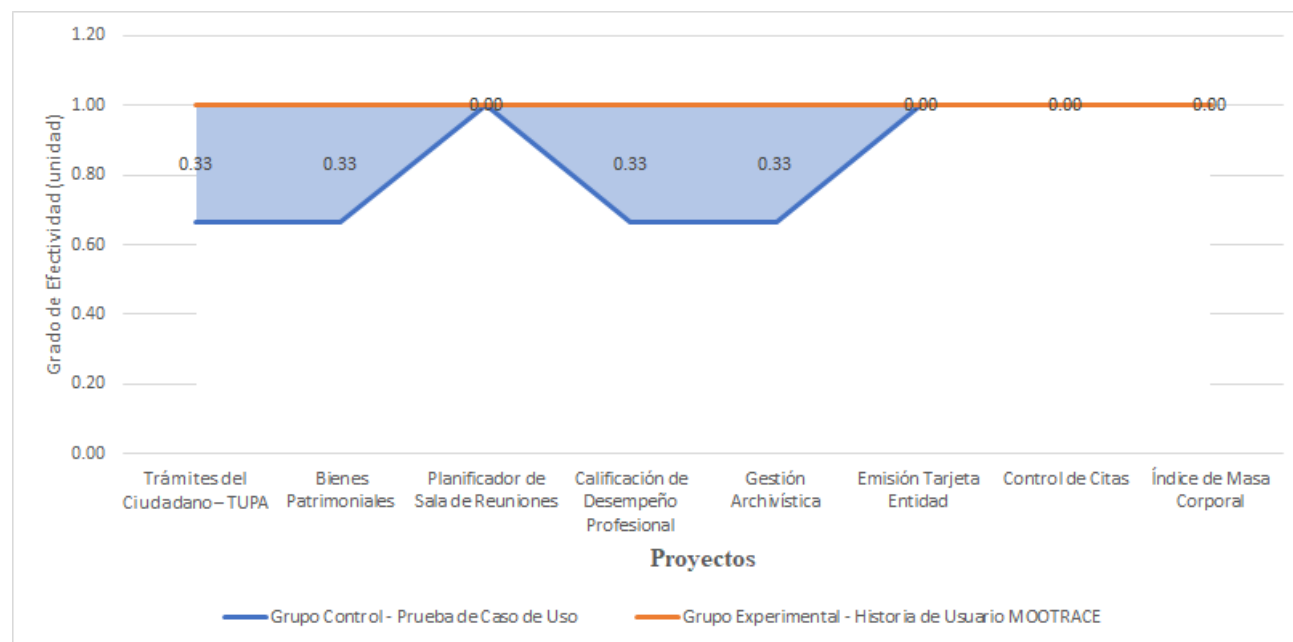


6.2.3 Prueba de Hipótesis del Objetivo Específico 2 Métrica 2

La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad incrementa la Calidad del Requisito. La Figura 57 muestra el comportamiento de la M2 que calcula el grado de **Validez del requisito**, el porcentaje de efectividad en los requisitos representa un 33% en promedio elicitados con Historias de Usuario es mayor frente a los especificados con Caso de Uso. En razón a ello, se infiere que los requisitos fueron validados por los usuarios y el equipo de proyectos determinó la viabilidad de las funcionalidades.

Figura 57

Gráfico del Porcentaje de eficiencia en la ejecución de la Métrica 2 del Objetivo Específico 2.



6.3 Presentación de resultados

HOE 1: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad reduce el tiempo de transformación de requisitos a funcionalidades. La medición del porcentaje de esfuerzo realizado (véase Figura 55) demuestra una reducción considerable en tiempo y eficacia en el proceso desarrollo del requisito en sus tres fases desde su captura hasta su validación.

HOE 2 M1: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad incrementa la Calidad del Requisito. El grado de **Especificidad del requisito** calculado (véase Figura 56) demuestra el porcentaje de efectividad en los requisitos elicitados con Historias de Usuario es mayor frente a los especificados con Caso de Uso, se infiere que los requisitos fueron entendidos adecuadamente.

HOE 2 M2: La herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad incrementa la Calidad del Requisito. El grado de **Validez del requisito** calculado (véase Figura 57) el porcentaje de efectividad en los requisitos elicitados con Historias de Usuario es mayor frente a los especificados con Caso de Uso. En razón a ello, se infiere que los requisitos fueron validados por los usuarios y el equipo de proyectos determinó la viabilidad de las funcionalidades.

7 CONCLUSIONES

En el marco de la investigación realizada y teniendo en cuenta la necesidad de la aplicación de procedimientos y herramientas que coadyuven a establecer con precisión los requisitos del usuario para generar conocimiento en el Campo de la Ingeniería de Requisitos y en la Ingeniería de Software, se concluye lo siguiente:

Se cumplió con el objetivo principal de la tesis que consistió en desarrollar una herramienta gráfica para la automatización de historias de usuario desde un modelo orientado a objetivos con gestión de la trazabilidad, la cual fue denominada MOOTRACE, en el capítulo 4 numeral 4.2 se definió la solución compuesta por sus 3 componentes: Modelado de Requisitos bajo el enfoque Orientado a Objetivos, Generación de Historias de Usuario y, Gestión de la Trazabilidad. Asimismo, en el Capítulo 5 se evidencia su implementación desde la elección de la tecnología con la que se desarrolló la herramienta hasta la fase de pruebas de su usabilidad que se evidencia en (D. Lucero, 2018).

Durante el uso de la Herramienta gráfica MOOTRACE en la fase de Elicitación de Requisitos, se comprobó que redujo el tiempo de transformación de requisitos en funcionalidades del sistema. Al respecto, en el capítulo 6 numeral 6.2.1, se evidencia que la reducción representa el 66.35 % en tiempo y eficacia en el proceso desarrollo del requisito en sus tres fases: Adquisición y Análisis, Especificación y Validación.

Durante el uso de la Herramienta gráfica MOOTRACE en la fase de Elicitación de Requisitos, se comprobó que incrementó la calidad de requisitos. Al respecto, para establecer los criterios de calidad se utilizaron 2 métricas: M1: Especificidad del requisito que determina mientras mayor la coherencia en la interpretación de los revisores para cada requerimiento menor será la ambigüedad y M2: Validez del requerimiento que determina si el requerimiento es claro y recogen todo aquello que el usuario espera del sistema, las cuales luego de la ejecución de las pruebas detalladas en el capítulo 6 numeral 6.2.1 numeral 6.2.2 y 6.2.3 obteniendo como resultado una mejora del 31,25% y 33% respectivamente.

8 RECOMENDACIONES

Conviene destacar que la presente investigación se desarrolla en el ámbito de la gestión de los requisitos, sin embargo, para ampliar el alcance se realizan las siguientes recomendaciones:

1. Para tener una visión holística de la fase inicial de Elicitación de requisitos se podría implementar el modelo SD de la metodología Orientada a Objetivos, que tiene como propósito una visión a nivel del proceso del negocio.
2. La información automatizada y vertida en la Historia de Usuario puede aportar a la gestión del proyecto exportándose en formatos xls, xml, etc. u otra tecnología que permita interoperar la información para ser incluida en los cronogramas para la fase de ejecución para mantenerlos actualizados de ser el caso, sin necesidad de duplicar esfuerzos.
3. Obtener indicadores que permitan gestionar de manera más eficiente la información de la trazabilidad de los requisitos e identificar y, analizar la criticidad de su culminación y/o cambios en las tareas o recursos dibujados en el modelo gráfico.

REFERENCIAS BIBLIOGRÁFICAS

- Agile Alliance. (2001). *Manifesto for Agile Software Development*. Obtenido de Manifesto for Agile Software Development: <https://agilemanifesto.org/>
- Akman, S., Özmut, M., Aydın, B., y Göktürk, S. (2016). *Experience report: implementing requirements traceability throughout the software development life cycle*. Journal of Software: Evolution and Process, 28(11), 950–954.
- Anderson, D. y Carmichael, Andy (2016). *Essential Kanban Condensed*. Lean-Kanban University. https://thinkingwithyou.com/wp-content/uploads/2018/01/EssentialKanbanSpanish_8-24.pdf
- Anaya R., Tabares M. S., Arango F. (2006). *Una revisión de modelos y semánticas para la trazabilidad de requisitos*. Revista EIA, ISSN 1794-1237 .6, pág. 33-42.
- Aguilar, J. A., Garrigós, I., Casteleyn, S. y Mazón, J. (2011). *Una propuesta orientada a objetivos para el análisis de requisitos en RIAs*, Universidad de Alicante, España.
- Agra, C. Sousa, A. Melo, J. Lucena, M. Alencar, F. (2015). *Especificando pautas para transformar el modelo i * en Historia de Usuario: una visión general*. Proceedings of the Eighth International i* Workshop. pág. 109–114, Canadá.
- Apshvalka, D., Donina, D., y Kirikova, M. (2009). *Understanding the Problems of Requirements Elicitation Process: A Human Perspective*. In W. Wojtkowski, G. Wojtkowski, M. Lang, K. Conboy, y C. Barry (Eds.), Information Systems Development, pág. 211–223. Boston, MA: Springer US. https://doi.org/10.1007/978-0-387-68772-8_17.
- Arias, M. Cechich, A. (2014). *Capacidades de Composición Dinámica de Componentes para una Línea de Productos de Software*, (Tesis de Licenciado en Ciencias de la Computación). Argentina.
- Bauer, Friedrich L. (1971). *Ingeniería de software*. IFIP Congress (1). pág. 530-538, Amsterdam
- Beatty, J. y Chen, A. (2012). *Visual Models for Software Requirements*. Washington, Microsoft Press.
- Beck, K. y Andres, C. (2004). *Extreme Programming Explained: Embrace Change (2nd Edition)*. Boston: Addison-Wesley Professional. ISBN: 0321278658
- Boehm, B. (1987). *Industrial software metrics top 10 list*. IEEE Software. 4(5), pág. 84-85.

Bourque, P. y Fairley, R. (2014). *Guide to the Software Engineering Body of Knowledge SWEBOK v3.0*. IEEE Computer Society.

Briones, G. (1996). *Epistemología de las ciencias sociales*. Bogotá: ICFES.

Buitrón Ruiz, S. L., Flores Ríos, B. L. y Pino Correa, F. J. (2017). *Elicitación de requisitos no funcionales basada en la gestión de conocimiento: el marco de trabajo Merlino*. Revista Ingenierías Universidad de Medellín, 17(32), 155-178. <https://doi.org/10.22395/rium.v17n32a8>.

Cao L. y Ramesh B. (2008). *Agile requirements engineering practices: An empirical study*. IEEE Software, 25(1), pág. 60–67.

Carrizo, D., Dieste, O., y Juristo, N. (2014). *Systematizing requirements elicitation technique selection*. Information and Software Technology, 56(6), 644–669. <https://doi.org/10.1016/j.infsof.2014.01.009>.

Carrizo, Dante, y Rojas, Jorge. (2018). *Metodologías, técnicas y herramientas en ingeniería de requisitos: un mapeo sistemático*. Ingeniare. Revista chilena de ingeniería, 26(3), 473-485. <https://dx.doi.org/10.4067/S0718-33052018000300473>

Christensen, C.M., Hall, T., Dillon, K., Duncan, D.S. (2016). *Know your customers Jobs to Be Done*. Harvard Business Review.

Ciancarini, P., Cimato, S., Mascolo, C. (1997). *Engineering Formal Requirements*. An Analysis and Testing Method for Z Documents. Annals of Software Engineering 3(1), 189–219.

Cleland-Huang, J., Gotel, O. C. Z., Huffman Hayes, J., Mader, P. y Zisman, A. (2014) *Software traceability: Trends and future directions*, in Proc. of FOSE. ACM, pág. 55–69.

Claus, Pablo. (2020). *Klein: Proceso de desarrollo de software con trazabilidad de requisitos, documentación mínima y aplicable con equipos pequeños y distribuidos*, (Tesis de maestría). Universidad Nacional de la Plata. Facultad de Informática.

Coimbra, Tavares R., Resende, A. y Terra R. (2018). *A Correlation Analysis between Halstead Complexity Measures and other Software Measures*. IEEE: XLIV Latin American Computer Conference (CLEI). pág. 31-39.

Cohn, M. (2004). *User Stories Applied: for Agile Software Development*. Addison Wesley.

Comai, S. y Carughi, G. T. (2007). *A behavioral model for rich internet applications*. Web Engineering: Springer, pág. 364-369.

Dalpiaz, F., Ferrari, A., Franch, X., Palomares, C. (2018). *Natural Language Processing for Requirements Engineering: The Best Is Yet to Come*. IEEE Software 35(5), pág. 115–119.

Dalpiaz, Fabiano y Sturm, Arnon. (2020). *Conceptualizing Requirements Using User Stories and Use Cases: A Controlled Experiment*. 10.1007/978-3-030-44429-7_16.

Donoso Barraza, B. y Vega Zepeda, V. (2017). *Factores sociales y humanos que afectan el proceso de educación de requisitos: una revisión sistemática*. Revista Ibérica de Sistemas y Tecnologías de Información, 24(10),69-83. <https://doi.org/10.22395/rium.v17n32a8>.

Durán, F., Troya, J. y Vallecillo, A. (2013). *Desarrollo de software dirigido por modelos*. FUOC.Fundación para la Universitat Oberta de Catalunya. pág. 25-26.

Escobar, A., Velandia, D., Ordoñez, H., Cobos, C. (2015). *A review of the impact on XP methodology of business model inclusion in requirements elicitation*. Sistemas & Telemática, 13(33). 45-61

Elicitar, <https://es.wikipedia.org/wiki/Elicitación> (Consultado desde enero 2020-actualidad).

Estrada, Hugo., Martínez Rebollar, Alicia., Pastor, Oscar., Mylopoulos, John. (2006). *An Empirical Evaluation of the i* Framework in a Model-Based Software Generation Environment*. Valencia University of Technology.

Espinoza, A. y Garbajosa, J. (2011). *Study to Support Agile Methods More Effectively through Traceability*. Computer Science Innovations in Systems and Software Engineering. 7 (1). 53-69.

Fangfei Lin, Hao Chen. (2019). *Comparative Study of Requirements Traceability in Facing Requirements Change: Systematic Literature Study and Survey*, (Tesis de maestría). Blekinge Institute of Technology. Faculty of Computing.Suecia.

Farid, W. M., & Mitropoulos, F. J. (2012). *Novel lightweightengineering artifacts for modeling non-functional requirements in agile processes*. IEEE Southeastcon, pág. 1–7.

Farfán, J. A., y Vega-Zepeda, V. (2020). *Mantenibilidad del software. consideraciones para su especificación y validación*. Ingeniare: Revista Chilena De Ingenieria, 28(4), 654-667.

Fiallos, A., Suarez P., Villavicencio, M. (2016). *Metodologías ágiles en el desarrollo de software: un enfoque basado en Scrum y MDD*. Escuela Superior Politécnica del Litoral. Guayaquil, Ecuador.

Figueroa-Díaz, Roberth., Sólis, Camilo., Cabrera-Silva, Armando. (2007). *Metodologías Tradicionales vs Metodologías Ágiles*. 10.13140/RG.2.1.2897.3206.

Finkelstein, A. C., Kappel, G. y Retschitzegger, W. (2002). *Ubiquitous web application development-a framework for understanding*.

Gonçalves, E., de Oliveira, M. A., Monteiro, I., Castro, J., y Araújo, J. (2019). *Understanding what is important in iStar extension proposals: The viewpoint of researchers*. Requirements Engineering (2019), 24(1), 55-84.

González-Baixauli, B., Sampaio, J., Mylopoulos, J., Laguna, M., Sampaio Do Prado Leite, (2004). *Visual Variability Analysis for Goal Models*. IEEE 12th International Requirements Engineering Conference. Kyoto, Japan

GoJS. (1998-2020). *Interactive JavaScript Diagrams in HTML*. Recuperado de: <https://gojs.net/latest/index.html>.

Ghanam, Y., Maurer F. (2008). *Un Modelo Iterativo Ágil para Ingeniería en Línea de Productos*. Accepted for special Issue of Journal of Systems and Software (JSS).

Gobova D. y Huchenko I. (2021). *Selección de técnicas de obtención de requisitos de software: Método para la gestión del alcance del proyecto*. Proceedings of the 2nd International Workshop IT Project Management, 2581. <http://ceur-ws.org/Vol-2851/>

Goguen, J.A. y Jirotko, M. (1994). *Requirements Engineering as the Reconciliation of Technical and Social Issues*. In Requirements Engineering: Social and Technical Issues. eds. London, UK: Academic Press.

Grigera, J., y Nava, A. (2021). *El futuro del trabajo en américa latina: Crisis, cambio tecnológico y control **. El Trimestre Económico, 88(4), 1011-1042.

Halmans, G., and Pohl, K. (2003). *Communicating the Variability of a Software-Product Family to Customers*. Journal of Software and Systems Modeling 2. pág.15-36.

Hanssen, G., and Faegri, T. (2008). *Process Fusion: An Industrial Case Study on Agile Software Product Line Engineering*. Journal of Systems and Software (JSS).

Hasnain, E. (2010). *An overview of published agile studies: A systematic literature review*. Proceedings of the national software engineering conference, pág. 1–6.

Hernández-González, Anaisa, Enríquez-Hernández, Dairis Milagros, y Ruano-Chichatskaia, Alexandra. (2021). *De modelos de negocio a requisitos del software*. Ingeniería Industrial, 42(2), 32-59.

Hernández Sampieri, R. (2014). *Capítulo 8 Selección de la muestra*. En Metodología de la Investigación (pág. 189). Mexico: Mc Graw Hill.

Hernández Sampieri, R. y Mendoza Torres, C. P. (2018). *Capítulo 11 El inicio del proceso cualitativo: planteamiento del problema, revisión de la literatura, surgimiento de las hipótesis e inmersión en el campo* (pág. 394). Mexico: Mc Graw Hill.

- Hsia, P., Davis, A., Kung, D. (1993). *Status Report: Requirements Engineering*. *IEEE Software*, 10(6):75-79.
- Hoda, R., Kruchten, P., Noble, J., Marshall, S. (2010). *Agility in Context*. Proc. of OOPSLA. pág. 74-88. ACM
- IEEE Standard Glossary of Software Engineering Terminology. (1990). IEEE Std 610.12-1990 1-84.
- IEEE830. (1998). *Recommended Practice for Software Requirements Specifications*. IEEE Std 830-1998, vol., no., pág.1-40.
- ISO/IEC 25030:2007. (2014). *Ingeniería del Software – Requisitos de Software de Calidad y Evaluación– Requisitos de Calidad*.
- Jacobson, I., Griss, M. y Jonsson, P. (1997). *Software Reuse: Architecture, Process and Organization for Business Success*. New York, NY: ACM Press; Harlow, England: Addison Wesley Longman.
- Jaqueira A., Lucena, M., Aranha, E. Alencar, F. Castro, J. (2013). *Uso de modelos i* para enriquecer historias de usuarios*. Proceedings of the 6th International i* Workshop, Valencia.
- Jarzabek, Stan. (2010). *XVCL (XML-based Variant Configuration Language)*. Department of Computer Science. School of Computing National University of Singapore.
- Jiménez, Álvaro. (2010). *M2DAT - DB Desarrollo de Editores Gráficos para el Modelado de Bases de Datos Objeto-Relacionales: SQL2003 y Oracle 10g*, (Tesis de Pregrado). Universidad Rey Juan Carlos, España.
- Jiménez, Álvaro. (2012). *Incorporando la Gestión de la Trazabilidad en un entorno de Desarrollo de Transformaciones de Modelos Dirigido por Modelos*, (Tesis de Doctorado). Universidad Rey Juan Carlos, España.
- Juristo, N. y Moreno, A. (2010). *Basics of Software Engineering Experimentation (1st. ed.)*. Springer Publishing Company, Incorporated.
- Lee, C., Guadagno, L. y Jia, X. (2003). *An agile approach to capturing requirements and traceability*. Proc. de TEFSE.
- Leite, J., Rossi, G., Balaguer, F., Majorana, V., Kaplan, G., Hadad, G., Oliveros, A. (1997). *Enhancing a Requirements Baseline with Scenarios*. IEEE Third International Requirements Engineering Symposium, IEEE Computer Society Press, pág. 44-53

Lindvall M. and Sandahl, K. (1996). Practical implications of Lindvall M. and Sandahl, K. Practical implications of traceability. *Journal of Software Practice and Experience*. 26(10), pág. 1161-1180.

Loucopoulos P. (1995). *System Requirements Engineering*. McGraw-Hill International series in Software Engineering, ISBN 0-07-707843-8.

Loucopoulos P. Karakostas V. (1995). *Modelado empresarial y enfoque teleológico de la ingeniería de requisitos*. *Revista internacional de sistemas de información cooperativos*, pp 45-79.

Lucassen, G., Dalpiaz, F., Van der Werf, J. y Brinkkemper, S. (2016). *The Use and Effectiveness of User Stories in Practice*. Daneva M., Pastor O. (eds) *Requirements Engineering: Foundation for Software Quality. REFSQ 2016. Lecture Notes in Computer Science*, 96(19). Springer, Cham.

Lucassen, G., Dalpiaz, F., van der Werf, JMEM., Brinkkemper, S. y Zowghi, D. (2017). *Trazabilidad de requisitos basados en el comportamiento a través de pruebas de aceptación automatizadas*. *IEEE 25th International Requirements Engineering Conference Workshops (REW)*, 431 -434. <http://dx.doi.org/10.1109/rew.2017.84>.

Lucassen, G., Keuken, M., Dalpiaz, F., Brinkkemper, S., Sloof, G. y Schlingmann, Johan. (2018). *Ingeniería de requisitos orientada a trabajos por hacer: un método para definir historias de trabajo*. DOI 10.1007/978-3-319-77243-1_14.

Lucero, D. (2018). *Automatización de requisitos: historias de usuario generadas a partir de un modelo orientado a objetivos basado en el framework i**. *Interfases*. Ed. n.º 11 // Enero-diciembre 2018 // ISSN 1993-4912. 57-72.

L. M. Cysneiros and E. Yu. (2004). *Non-functional requirements elicitation: in Perspectives on software requirements*. ed: Springer. pág. 115-138.

Mader, P. y Egyed, A. (2015). *Do developers benefit from requirements traceability when evolving and maintaining a software system? Empirical Software Engineering*, 20(2), pág. 413–441.

Manifiesto for Agile Software Development. (2001). *Manifiesto for Agile Software Development*. Recuperado de <https://agilemanifesto.org/>

Model Driven Architecture (MDA). Object Management Group. (2000). Recuperado de <http://www.omg.org/mda/>

Manrique, B., Quintero, J.B., Marin, D.A. y Morales, J.S. (2018). *Analítica de Texto para procesar documentos de requisitos*. *En Tecnologías del lenguaje humano: aplicaciones desde la Lingüística Computacional y de Corpus*. Sello Editorial Universidad de Medellín, Ed. Medellín.

Marín-Alvarez, D. A., Manrique-Losada, B., & Quintero, J. B. (2020). *Método para la representación semi-automática de modelos conceptuales desde documentos de negocio escritos en lenguaje natural en español*. *Ingeniare: Revista Chilena De Ingeniería*, 28(4), 565-584.

Mylopoulos, J., Chung, L., Yu, E. (1999). *From Object-oriented to Goal-oriented Requirements Analysis*. *Communications of the ACM* 42(1), pág. 31–37.

Moreno-Rodríguez, Rosendo. (2020). *CASE jMDA de Arquitectura Dirigida por Modelos para Sistemas de Información*. *Revista Cubana de Ciencias Informáticas*, 14(4), 210-223. Epub 01 de diciembre de 2020.

Nieves Guerrero, C., Ucán Pech, J. y Menéndez Domínguez, V. (2014). *UWE en Sistema de Recomendación de Objetos de Aprendizaje. Aplicando Ingeniería Web: Un Método en Caso de Estudio*. *Revista Latinoamericana de ingeniería de software*. 2. 137-143.

Ozkaya, M. (2016). *What is software architecture to practitioners: A survey*. 2016 4th Int.Conf. Model. Eng. Softw. Dev., pág. 677-686.

Paige, R., Xiaochen, W., Stephenson, Z., and Phillip J. (2006). *Hacia la Construcción de un proceso ágil para una Línea de Productos*. LNCS: XP, pág. 198 – 199.

Pluralsight Technology Index. (2004-2020). *Top technologies for Software Development*. Recuperado de: <https://www.pluralsight.com/tech-index/>.

Project Management Institute. (2017). *A Guide to the Project Management Body of Knowledge (PMBOK Guide)*, 6th. ed. Project Management Institute: Newtown Square, PA, USA.

Qasaimeh, M. (2011). *Extending extreme programming user stories to meet ISO 9001 formality requirements*. *Journal of Software Engineering and Applications*, 4 (11).626-638.

Ottensooser, A., Fekete, A., Reijers, H.A., Mendling, J., Menictas, C. (2012). *Making Sense of Business Process Descriptions: An Experimental Comparison of Graphical and Textual Notations*. *Journal of Systems and Software*.

Ozkaya, M. (2016). *What is software architecture to practitioners: A survey*. 2016 4th Int.Conf. Model. Eng. Softw. Dev., pág. 677-686.

Ramesh B. (1998). *Factors influencing requirements traceability practice*. *Communication of the ACM*. 41(12), pág. 37-44.

Ramírez Leal, J. C., Giraldo Orozco, W. J., y Anaya Hernández, R. (2017). *Una propuesta metodológica para mejorar la comunicación en ingeniería de requisitos*. *Revista EIA*. 13(26), 121-139. Recuperado de: <https://doi.org/10.24050/reia.v13i26.952>.

- Rendón Rojas, M. Á. (2021). Una epistemología dialéctica para el análisis de la ciencia. *Palabra Clave*, 10(2) doi:<http://dx.doi.org/10.24215/18539912e121>
- Ron Jeffries. (1998-2020). *Extreme Programming (Figura 35)*. Recuperado de <https://ronjeffries.com/xprog/what-is-extreme-programming/>
- Roos, F. (2012). *Análisis automatizado de líneas de productos de software con modelos de variabilidad ortogonal: ampliación del ecosistema FaMa*, (Tesis de doctorado) Universidad de Sevilla. España.
- Rubin, Kenneth. (2012). *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley Professional / Upper Saddle River, New Jersey.
- Sealights Technologies. (2020). *Agile Testing Metrics*. Recuperado de: <https://www.sealights.io/test-metrics/testing-metrics-in-agile-development/>
- Sehrish A., Shahid N. B., S. Asim A. S. y Amr M. J. (2017). *Impacto y desafíos de la ingeniería de requisitos en metodologías ágiles: una revisión sistemática*. Revista internacional de aplicaciones y ciencias informáticas avanzadas (IJACSA), 8 (4), pág. 411-420. <http://dx.doi.org/10.14569/IJACSA.2017.080455>
- Serna, Alexei. (2012). Formal Methods in industry. *Los Métodos Formales en la Industria*, Revista Antioqueña de las Ciencias Computacionales y la Ingeniería de Software (RACCIS). 2(2), pág. 44-51.
- Serna, Edgar. (2013). *La ruta de la profesionalización de la ingeniería de software*. En *Libro Blanco de la Ingeniería de Software en América Latina* urbanos (pág.113-135). Valencia: Universidad Politécnica de Valencia.
- Serna, Edgar, y Serna, Alexei. (2020). *Process and progress of requirement formalization in software engineering*. Ingeniare: Revista Chilena De Ingeniería, 28(3), 411-423.
- Sommerville, Ian y Sawyer, Pete. (1997). *Requeriments Engineering A good practice guide*, 10(4). Addison-Wesley, Boston.
- Sommerville, I. (2011). *Ingeniería de Software 9th edición*. Pearson Educación de México.
- StarUML. (2020). *Herramienta para crear diagramas UML (Unified Modeling Language)*. Enlace para descarga en Disponible en <https://staruml.io/download>.
- Sutherland, J., & Schwaber, K. (2017). *The Scrum Guide*. <http://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-US.pdf>

Robinson, P. T. (2019). *Communication Network in an Agile Distributed Software Development Team*. ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE)., 100-104.

Szyperski, Clemens. (2002). *Component Software: Beyond Object-Oriented Programming*. Addison-Wesley and ACM Press, Boston.

Teeravarunyou, Sakol, Sato, Keiichi. (2001). *Object-Mediated User Knowledge Elicitation Method. A Methodology in Understanding User Knowledge*. Institute of Design - Illinois Institute of Technology, Chicago, USA. The proceeding of the 5th Asian International Design Research Conference, Seoul, Korea.

The Nielsen Company. (2011). *Sales of vitamins/minerals & herbal supplements in New Zealand, 11 Septiembre 2010 – 11 Septiembre 2011*[Tabla]. Recuperado de: ACNielsen Market Information Digest New Zealand.

Thomas P. (2005). *Definición de un Proceso de Elicitación de Objetivos*, (Tesis de pregrado). Facultad de Ingeniería de la Universidad Nacional de la Plata. Argentina

Urroz Urzúa, G. (2012). *Adaptación de Software de Aplicación al Paradigma de la Ingeniería de Línea de Productos de Software*. Disponible en <http://repositorio.uchile.cl/handle/2250/104366>

Valdecantos, Héctor. (2010). *Principios y patrones de diseño de software en torno al patrón compuesto Modelo Vista Controlador para una arquitectura de aplicaciones interactivas*, (Tesis de pregrado). Facultad de Ciencias Exactas y Tecnología Departamento de Ciencias de Computación. Tucumán.

Valverde, Carolina. (2014). *Calidad de Datos en Experimentos de Ingeniería de Software*, (Tesis de maestría). Universidad de la República de Montevideo. Facultad de Ingeniería.Uruguay.

Van Lamsweerde, A. (2009). *Requirements Engineering: From System Goals to UML Models to Software Specifications*. Wiley.

Vara-Horna, Arístides (2012). *Desde la idea hasta la sustentación: Siete pasos para una tesis exitosa. Un método efectivo para las ciencias empresariales*. Instituto de Investigación de la Facultad de Ciencias Administrativas y Recursos Humanos. Universidad de San Martín de Porres. Lima. Manual electrónico disponible en internet: <http://www.doctorvara.com/421668511> 451.

Vargas, C. A., Basto Guevara, M., Jiménez Millán, N., Barón Velandia, J., y Daza, A. (2013). *Sistema integrado de información para el régimen subsidiado de salud en Colombia*. Revista vínculos. 10(2). 355-368. <https://doi.org/10.14483/2322939X.6522>

Vela, B., Cáceres, P., de Castro, V., Marcos, E. *MIDAS: Una Aproximación Dirigida por Modelos para el Desarrollo Ágil de Sistemas de Información Web*. España, 2005.

Velázquez Fernández, Ángel R. y Rey Córdova, Nériida (1999). *Metodología de la investigación científica*. Lima: San Marcos.

A. Verma, G.S. Kamboj, y I. Kaur. (2017). *Genetic optimized software component reuse model for reducing development and maintenance efforts*. Int. J. Adv. Appl. Sci. Vol. 4(10), pág. 144-149.

Villafañe, A., Ferraro, M., Medina, y., Greiner, Cristina., Dapozo, Gladys., Estayno, Marcelo. (2013). *Herramienta de gestión de trazabilidad de requisitos en proyectos de software*. X Workshop Ingeniería de Software (WIS). Universidad Nacional del Nordeste. Corrientes, Argentina.

Wang, X., Zhao, L., Wang, Y., Sun, J. (2014). *The Role of Requirements Engineering Practices in Agile Development: An Empirical Study*. Proc. of APRES, CCIS, 432, pág. 195–209.

Wakil, K. y Jawawi, D. N. (2018). *A New Adaptive Model for Web Engineering Methods to Develop Modern Web Applications*. ICSIM2018: *Proceedings of the 2018 International Conference on Software Engineering and Information Management*, pág. 32-39.

Yu, Eric. (1995). *Modelling Strategic Relationships for Process Reengineering*, (Tesis de Doctor). University of Toronto. Canadá.

Yu, Eric. (1997). *Towards Modeling and Reasoning Support for Early-Phase Requirements Engineering*. Third IEEE Symposium on Requirements Engineering. pág. 226-235.

ANEXOS

Anexo 1: Siglario

BPMN	Business Process Model and Notation
IDE	Entorno de Desarrollo Integrado
MDA	Arquitectura Dirigida por Modelos
MOO	Metodología Orientada a Objetivos
LPS	Línea de Producto de Software
RF	Requisitos Funcionales
RNF	Requisitos No Funcionales
SD	Strategic Dependency
SR	Strategic Rationale

Tabla 28

Historia de Usuario2.

Historia de Usuario	
Número: 2	Usuario: Analista y Desarrollador
Nombre historia: Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico.	
Prioridad en negocio: Alta	Riesgo en desarrollo: (Alta / Media / Baja)
Puntos estimados:	Iteración asignada: 2
Programador responsable: Delicia Lucero	
Descripción: Los iconos deben representar los estereotipos establecidos en el modelado orientado a objetivos.	
Observaciones: Uso de la librería GOJS.	

Tabla 29

Historia de Usuario 3.

Historia de Usuario	
Número: 3	Usuario: Desarrollador
Nombre historia: Generar automáticamente la Historia de Usuario con la trazabilidad del modelo gráfico.	
Prioridad en negocio: Alta	Riesgo en desarrollo: (Alta / Media / Baja)
Puntos estimados:	Iteración asignada: 3
Programador responsable: Delicia Lucero	
Descripción: Automatización de las relaciones de los objetos del modelo SR para generar automáticamente las Historias de Usuario establecidas en la correspondencia de la Figura 36.	
Observaciones: Ninguna	

Tabla 30*Historia de Usuario 4.*

Historia de Usuario	
Número: 4	Usuario: Desarrollador
Nombre historia: Personalización de Historia de Usuario y cierre de trazabilidad.	
Prioridad en negocio: Alta	Riesgo en desarrollo: (Alta / Media / Baja)
Puntos estimados:	Iteración asignada: 4
Programador responsable: Delicia Lucero	
Descripción: Asignar la prioridad, duración de la tarea y estado de la Historia de Usuario que permita dar por implementado el requisito.	
Observaciones: Ninguna	

Tabla 31

Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico (Historia 2).

Especificación de Prueba: Personalizar estereotipos del modelado orientado a objetivos en el editor gráfico (Historia 2).

Descripción

En esta historia se debe corroborar que los elementos que componen la barra de herramientas deben de corresponder a los estereotipos de la metodología orientada a objetivos.

Creación correcta de un nodo

Descripción

El usuario debe tener a disposición una barra de herramientas cuyos elementos representen los estereotipos establecidos por la metodología orientada a Objetivos.

Para representar un elemento de la barra de herramientas en el modelo debe dar clic en el elemento de la barra de herramientas y luego volver hacer clic en la posición que desee en el modelo gráfico.

Condiciones de ejecución

Ninguna.

Entrada

- De la barra de herramienta se hará clic en el nodo que desea dibujar y luego ubicar el cursor del mouse en la posición que desee en el modelo gráfico.
- Luego doble clic para editar el nombre del nodo.
- Si selecciona añadir otro nodo, realizará la misma secuencia de lo antes descrito.

Resultado esperado

El nodo que se seleccionó de la barra de herramientas debe ser el mismo que se ha dibujado en el modelo gráfico.

Creación incorrecta de un nodo

El usuario hará clic en la barra de herramientas y arrastrará el nodo al modelo.

Condiciones de ejecución

Ninguna.

Entrada

-De la barra de herramientas el usuario arrastra el hacia el modelo.

-El sistema no hará ninguna acción.

Resultado esperado

Tras el arrastre el elemento al que se hizo clic en la barra de herramientas no se mostrará en el modelo.

Evaluación de la prueba

Prueba exitosa.

Tabla 32

Especificación de Prueba: Automatización de las relaciones de los objetos del modelo SR para generar automáticamente las Historias de Usuario establecidas en la correspondencia de la Figura 31 (Historia 3)

Especificación de Prueba: Automatización de las relaciones de los objetos del modelo SR para generar automáticamente las Historias de Usuario establecidas en la correspondencia de la Figura 31 (Historia 3)

Descripción

En esta historia se debe comprobar que la interfaz permite crear relaciones entre los nodos (actores, objetivos, tareas, recursos y subobjetivo) cuya trazabilidad, se reutilizará en el posterior proceso de automatización de las Historias de Usuario.

Creación correcta de las relaciones entre los elementos del SR

Descripción

Todos los objetos del modelo deben tener al menos alguna relación para que las Historias de Usuario sean generadas con esta información.

Condiciones de ejecución

Deben existir al menos 2 nodos para la creación de una relación.

Entrada

-Del modelo gráfico para asignar una relación entre nodo actor y nodo objetivo, debe hacer clic en el nodo del primero y arrastrar el cursor del mouse hacia el segundo.

-Del modelo gráfico para asignar una relación entre nodo objetivo y nodo tarea, debe hacer clic en el primer nodo y arrastrar el cursor del mouse hacia el segundo.

-Del modelo gráfico para asignar una relación entre nodo tarea y nodo recurso, debe hacer clic en el primer nodo y arrastrar el cursor del mouse hacia el segundo.

-Del modelo gráfico para asignar una relación entre nodo subobjetivo y nodo tarea, debe hacer clic en el primer nodo y arrastrar el cursor del mouse hacia el segundo.

Resultado esperado

Visualización del modelo gráfico con las relaciones entre los nodos del modelo.

Evaluación de la prueba

Prueba exitosa.

Creación incorrecta de las relaciones entre los elementos del SR

No se considera la precedencia entre los nodos, ni la cantidad de ellos en el modelo.

Condiciones de ejecución

Deben existir al menos 2 nodos para la creación de una relación.

Entrada

-Del modelo gráfico para asignar una relación entre nodo objetivo y nodo actor, debe hacer clic en el nodo del primero y arrastrar el cursor del mouse hacia el segundo.

-Del modelo gráfico para asignar una relación entre nodo tarea y nodo objetivo, debe hacer clic en el primer nodo y arrastrar el cursor del mouse hacia el segundo.

-Del modelo gráfico para asignar una relación entre nodo recurso y nodo tarea, debe hacer clic en el primer nodo y arrastrar el cursor del mouse hacia el segundo.

-Del modelo gráfico para asignar una relación entre nodo tarea y nodo subobjetivo, debe hacer clic en el primer nodo y arrastrar el cursor del mouse hacia el segundo.

Resultado esperado

Tras el diseño del modelo este no permitirá generar las Historias de Usuario.

Evaluación de la prueba

Prueba exitosa.

Tabla 33

Personalización de Historia de Usuario (Historia 4)

Especificación de Prueba: Personalización de Historia de Usuario (Historia 4)

Descripción

En esta historia se debe permitir completar información de la HU con valores en base a criterios técnicos por parte del programador como: asignar la prioridad, duración de la tarea y estado de la Historia de Usuario.

Asignación correcta de criterios técnicos

Descripción

El formato automatizado de la Historia de Usuario debe permitir ingresar la prioridad, duración de la tarea y estado de esta.

Condiciones de ejecución

Existe un modelo gráfico previo con sus respectivas relaciones.

Entrada

- En la historia de usuario se ingresará el valor para la prioridad.
- En la historia de usuario se seleccionará la fecha de inicio y la fecha fin de la tarea.
- En la historia de usuario se ingresará el estado de esta “Terminada” o “No terminada”.

.

Resultado esperado

Se visualizará el formato digitalizado de la HU con los datos completos y se guardarán los datos en la base de datos.

Evaluación de la prueba

Prueba exitosa.

Creación incorrecta de las relaciones entre los elementos del SR

El usuario presiona el botón para la generación de Historias de Usuario, dejando al menos un elemento sin ser relacionado.

Condiciones de ejecución

Ninguna.

Entrada

- Del panel del modelo presiona el botón de generación de Historias de Usuario.
- Se mostrará un mensaje denegando la generación de HU porque no se han completado las relaciones en el modelo.

Resultado esperado

Tras el diseño del modelo este no se guardará en la base de datos.

Evaluación de la prueba

Prueba exitosa