Establecimiento del estado sobre el proceso de implantación de sistemas informáticos
Reporte Técnico RT-UNLP- III-LIDI-2019-01

Doctoranda: Marisa Daniela Panizzi
Directores: Rodolfo Bertone - Alejandro Hossian
Instituto de Investigación en Informática- III-LIDI. Facultad de Informática – Universidad Nacional de La Plata
marisapanizzi@outlook.com

Introducción.

Para la construcción del estado del arte de la tesis doctoral, se realiza un mapeo sistemático de la literatura (en Inglés Systematic Mapping Studies o SMS), con el objetivo de cotejar y sistematizar la evidencia empírica para identificar (determinar) cuáles de las metodologías de desarrollo de software, son las más estudiadas por la comunidad científica y las más utilizadas en la industria del software (Sección 1). Una vez identificadas las metodologías de desarrollo, se decide cuales se consideran para el análisis comparativo con el propósito de identificar si estas custodian el proceso de implantación de sistemas informáticos (Sección 2). Por último, se presentan las conclusiones sobre el estado de arte (Sección 3).

En el contexto de esta línea de investigación, se considera al proceso de implantación, como el conjunto de actividades y tareas necesarias que permiten la transferencia del producto software finalizado a su ambiente de utilización por parte de la comunidad usuaria (Panizzi et al. 2017).

1. Desarrollo del Mapeo Sistemático de la literatura (SMS).

El SMS permite clasificar y analizar la literatura sobre un tema específico de la Ingeniería de Software (Genero et al., 2014). Para su desarrollo, se utiliza el proceso propuesto por Kitchenham et al. (Kitchenham et al., 2007) y Genero et al. (Genero et al., 2014). En este, se proponen tres actividades: la planificación de la revisión (Sección 1.1.), ejecución de la revisión (Sección 1.2), y reporte de la revisión (Sección 1.3). Las actividades del proceso, se presentan en la Fig. 1.

Fig. 1. Actividades de la revisión sistemática de la literatura.

1.1.Planificación.

En esta actividad, se definen las tareas a realizar durante la revisión sistemática de literatura. Estas tareas comienzan por identificar la necesidad del estudio (Sección 1.1.1.), continua por definir las preguntas de investigación que se buscan contestar (Sección 1.1.2), definir el protocolo de la revisión (Sección 1.1.3) y por último, validar el protocolo de la revisión (Sección 1.1.4). En la Fig. 2., se presentan las tareas de la Actividad ¨Planificación¨.

Fig. 2. Tareas de la Actividad ¨ Planificación¨.

1.1.1. Identificar la necesidad del estudio.

Esta tarea, tiene por objetivo identificar (determinar) cuáles de las metodologías de desarrollo de software son las más estudiadas por la comunidad científica, ya sea porque se las compara, se analizan sus aportes, se evalúa su aplicación en la industria del software, se experimenta con ellas, etc.

En este estadío de la investigación, se presenta la necesidad de desarrollar la definición de algunos conceptos, que en adelante se utilizarán en la tesis doctoral. El propósito de este glosario, consiste en presentar una terminología unificada a fin de evitar ambigüedades en la acepción que se le dará en este trabajo. En la Tabla 1., se presenta el glosario de términos con sus definiciones.

Tabla 1. Glosario de términos

Término	Definición
Proceso de software	Guía la ejecución del proyecto software; por tanto, la clara definición de un proceso software representada mediante un modelo del proceso y su adecuado seguimiento redunda en la generación de productos de mayor calidad (Osterweil, 1987). Un conjunto de actividades para gestionar, desarrollar y mantener sistemas de software. En otras palabras, el proceso de software se centra en las tareas de construcción y no en los productos de salida. La definición de un proceso software debe especificar no sólo las actividades, sino también las técnicas para realizar las tareas, los actores que ejecutan las actividades, sus roles y los artefactos producidos (Acuña et al., 2005).
Metodología de desarrollo de software	En Piattini et al. (Piattini, 2004), se define como un conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar un nuevo software. Esta descomposición del proceso de desarrollo guía a los desarrolladores en la elección de las técnicas que debe elegir para cada estado del proyecto, así como facilita la planificación, gestión, control y evaluación de los proyectos. Representa el camino para desarrollar software de una manera sistemática.
Método	Método Standard. Describe las características del proceso o procedimiento ordenado utilizado en la ingeniería de un producto o en la prestación de un servicio. (IEEE Standard 610.12-1990).
Marco de trabajo (Framework)	En el marco de CCMI, se define como la estructura base que organiza los componentes CMMI, que incluye los elementos de los modelos actuales de CMMI, así como las reglas y los métodos para generar los modelos, los métodos de evaluación (incluyendo los artefactos asociados) y los materiales de formación. (CMMI – Dev., 2010). En este trabajo y en base a la definición anterior, se considera marco de trabajo para la construcción de un producto-software, a las actividades/tareas, artefactos, técnicas y prácticas, roles, métodos y herramientas.
Standard	La Real Academia de la Lengua Española, lo define como aquello que sirve como modelo, norma, patrón o referencia (RAE, 2005). Una segunda aproximación proviene del concepto utilizado por instituciones o agencias que promueven el uso de estándares en el sistema productivo. Una propuesta de definición es la entregada por la *Internacional Organization for Standardization* (ISO), definiendo un estándar como un documento establecido por consenso y aprobado por un cuerpo reconocido, que prevé el uso común y reiterado de reglas, pautas o características para las actividades o sus resultados esperados en el estándar (ISO/IEC-GUIDE, 2004).

1.1.2. Formular las preguntas de investigación.

En esta tarea, se plantea la Pregunta de Investigación (PI), y su descomposición en cuatro preguntas (PI1-4). La pregunta de investigación (PI): ¿Qué se ha publicado sobre metodologías de desarrollo de software a partir del año 2008?, y su descomposición se presenta en la Tabla 2., junto a la motivación de las mismas.

Tabla 2. Preguntas de investigación

Ref.	Preguntas	Motivación
PI1	¿Qué metodologías permiten llevar adelante la construcción de un producto software?	Descubrir que tipos de metodologías de desarrollo de software se investigaron.
PI2	¿En qué actividades de la construcción del producto software, gestión o procesos de soporte se centra la investigación?	Determinar en qué actividad de la construcción de la construcción del producto software se investiga (Requisitos, desarrollo, análisis y diseño, despliegue, pruebas, mantenimiento), gestión del proyecto o procesos de soporte (verificación y validación, gestión de la configuración, etc.)
PI3	¿Qué aportes realizan los trabajos?	Encontrar el tipo de resultados obtenidos en el contexto de las metodologías de desarrollo (proceso, modelo, herramienta, métrica, método, buena práctica, etc.)
PI4	¿Cuáles son los tipos de investigación sobre las metodologías de desarrollo?	Determinar donde se encuentra interés de investigación de las metodologías de desarrollo (se las evalúa, se las valida, proponen una solución, comunican experiencias, se les hace un juicio de expertos, etc.).

1.1.3. Definir el protocolo de revisión.

En esta tarea, se describen los elementos del protocolo para llevar a cabo el SMS, la definición de las estrategias de búsqueda (Sección 1.1.3.1.), y la definición de los criterios de selección de estudios (Sección 1.1.3.2.).

1.1.3.1. Estrategias de búsqueda.

Los términos candidatos (T) que se consideran más adecuados para definir la cadena de búsqueda, se presentan en la Tabla 3. Los términos han sido definidos en los idiomas español e inglés.

Tabla 3. Términos de investigación (T).

Referencia	Términos	Referencia	Términos
T1	desarrollo de software	T2	software development
T3	metodologías	T4	methodologies
T5	método	T6	methods
T7	software	T8	framework
T9	proceso	T10	development
T11	process		

Las cadenas de búsqueda definidas, se presentan en la Tabla 4.

Tabla 4. Cadenas de búsqueda

Cadenas de búsqueda
((¨desarrollo de software)¨ AND (¨proceso¨ OR ¨metodologías¨ OR ¨métodos¨)) OR ((¨software development¨) AND (¨process¨ OR ¨methodologies¨ OR ¨methods¨)) OR ((¨software)¨ AND (¨proceso¨ OR ¨process¨ OR ¨framework¨ OR ¨development framework¨)) OR ((framework¨ AND ¨methodologies ¨AND ¨ ¨software¨)) OR ((¨development¨) AND (¨methodologies¨) AND (¨software¨)) OR (¨development¨ AND ¨methodology¨)

1.1.3.2. Criterios de selección de estudios.

En esta sub-tarea, se definen los criterios de inclusión y exclusión de los estudios primarios. El objetivo que se persigue es identificar cuales de ellos muestran evidencia con respecto a las preguntas de investigación propuestas en la Tabla 2. Se definen los criterios de inclusión y exclusión, que se presentan en la Tabla 5.

Tabla 5. Criterios de inclusión y exclusión

Criterios de inclusión:
○ Artículos que respondan a las preguntas de investigación: a.1.Metodologías que se utilizan para la construcción de producto de software a.2.En qué actividades de la construcción de software se centra la investigación a.3.Tipos de aporte propuestos a.4.Interés de la investigación.
○ Estudios duplicados: si hay varios artículos de un mismo autor que contemple la misma investigación, se considerara el más completo.
○ Artículos publicados a partir del año 2008
○ Artículos en Español e Inglés
Criterios de exclusión:
a. Estudios a los que no se tiene acceso
b. Artículos que no coincidan con los criterios de inclusión

Se decide la búsqueda en las bibliotecas digitales que se presentan en la Tabla 6. Dado que la tesis doctoral se desarrolla en la República Argentina, los directores de la tesis recomendaron la revisión de las publicaciones del Congreso Argentino de Ciencias de la Computación (CACIC), de las Jornadas Argentinas de Informática e Investigación Operativa (JAIIO) y de la Revista Latinoamericana de Ingeniería de Software, dado que cuentan con estudios del ámbito nacional. También se consideró la revisión de eventos científicos de España por la vinculación del grupo de investigación con ese país.

Tabla 6. Base de datos y criterios de búsqueda para la selección de resultados.

Bibliotecas digitales	Opciones
IEEE Explore	Publicaciones de congresos, revistas
ACM Digital Library	Publicaciones de congresos, revistas
Springer	Publicaciones de congresos, revistas
Elsevier Science (Science Direct)	Publicaciones de congresos, revistas
SEDICI [1]	Publicaciones del Congreso Argentino de Ciencias de la Computación (CACIC) Actas de las Jornadas Argentinas de Informática e Investigación Operativa (JAIIO)
RELAIS [2]	Revista Latinoamericana de Ingeniería de Software (RELAIS)
SISTEDES Biblioteca Digital [3]	Publicaciones de las Jornadas de Ingeniería de Software y Base de Datos (JISBD)

1.1.4. Validación del protocolo de revisión.

El protocolo de revisión para la construcción del SMS, ha sido validado con el grupo de investigación y los Directores de la tesis doctoral.

[1] SEDICI: Repositorio digital de la Universidad Nacional de La Plata, en el cual se publica las Actas del Congreso Argentino de Ciencias de la Computación y los artículos de las Jornadas Argentinas de Informática e Investigación Operativa organizadas por SADIO (Sociedad Argentina de Informática), sitio: http://sedici.unlp.edu.ar/

[2] RELAIS: Revista Latinoamericana de Ingeniería de Software, Publicación Técnica de acceso abierto de la Red de Ingeniería de Software de Latinoamérica coordinada por la Universidad Nacional de Lanús, Argentina. ISSN: 2314-2642; sitio: http://revistas.unla.edu.ar/software/index

[3] Biblioteca Digital Sistedes. La Fundación Sistedes organiza las Jornadas Sistedes, que alberga las Jornadas de Ingeniería de Software y Base de Datos (JISBD) , sus actas se pueden encontrar en el sitio: https://biblioteca.sistedes.es/biblioteca/conferencias/

1.2. Realización de la revisión.

En esta actividad, se definen las tareas para la ejecución de la revisión de la literatura. Se comienza por la selección de los estudios primarios (Sección 1.2.1.), continúa por la evaluación de la calidad de los artículos (Sección 1.2.2), la extracción de datos relevantes (Sección 1.2.3) y por último, la síntesis de los datos extraídos (Sección 1.2.4). En la Fig. 3., se presentan las tareas de la Actividad ¨Realización de la revisión¨.

Fig. 3. Tareas de la Actividad ¨Realización de la revisión¨.

1.2.1. Selección de estudios primarios.

En esta tarea, se realiza la selección de los estudios considerando algunos criterios utilizados en el estudio realizado por Sierra et al. (Sierra et al., 2017), estos consisten en:

- aplicar los procedimientos definidos para la selección(estrategias de búsqueda, criterios de inclusión/exclusión),
- los resultados obtenidos en las fuentes de búsqueda,
- extracción de datos,
- clasificación de los diferentes documentos.

La cantidad de artículos seleccionados en cada una de las bases de datos se presentan en la Tabla 7. Los artículos son clasificados en: *artículos relevantes, artículos no considerados y artículos primarios*. El significado de esta clasificación, se explica a continuación:

- *Artículos relevantes*: son los encontrados y seleccionados de los resultados de búsqueda iniciales que contienen los términos de búsqueda en el resumen, la introducción, las palabras clave o título.
- *Artículos no considerados:* son los artículos resultantes de la búsqueda que respetan los criterios de inclusión/exclusión y luego de su lectura no se consideran adecuados para la investigación.
- *Artículos primarios*: son los artículos que han sido leídos de manera completa y se consideran adecuados para la investigación porque cumplen con los criterios de inclusión.

Tabla 7. Base de datos y trabajos seleccionados para cada etapa.

Fuentes de búsqueda	Artículos relevantes	Artículos no considerados	Artículos primarios
IEEE Explore	57	13	44
ACM Digital Library	15	6	9
Springer	4	3	1
Elsevier Science (Science Direct)	16	4	12
SEDICI	11	1	10
SISTEDES Biblioteca Digital	3	1	2
RELAIS	2	2	0

En el Apéndice A, se presenta la lista de estudios primarios utilizados para el SMS.

1.2.2. Evaluación de la calidad de los artículos.

Para determinar la calidad de los artículos se adhiere al criterio que los estudios primarios se encuentran publicados en revistas indexadas y en eventos científicos con revisión por pares según la propuesta de Genero et al. (Genero et al., 2014).

1.2.3. Extracción de los datos relevantes.

Los datos extraídos de los estudios primarios se registraron en una plantilla. El esquema de extracción se compone de dos partes:

- La primera con los metadatos de cada estudio primario, denominada *Dimensión General.*
- La segunda parte, compuesta por dimensiones y categorías específicas del esquema definido para clasificar los estudios primarios seleccionados. Las dimensiones consideradas son: *Contexto*, *Contribución*, *Tipo de investigación* y *Resultados preliminares.* La definición de estas dimensiones y sus categorías se define en función de las preguntas de investigación planteadas en la sección 1.1.2., se muestran en la Tabla 8.

Tabla 8. Extracción de datos de los estudios primarios.

Datos extraídos	
Dimensión	Categorías
General:	ID (Identificador de registración del artículo), cadena de búsqueda, Año, título, autores, fuente de búsqueda, tipo de publicación, país, palabras clave, citas, problema y resultados.
Contexto*:	*Procesos principales*: Requisitos, Desarrollo, Despliegue o Implantación, Arquitectura y Diseño, Mantenimiento, Pruebas. *Procesos de soporte y de la organización*: Verificación y validación, gestión de la configuración, gestión del proyecto, gestión del riesgo, gestión de la calidad.
Contribución**:	Métrica, modelo, proceso, herramienta, artefacto, método/metodología, práctica/s, no contribuye.
Tipo de investigación***:	Validación, evaluación, propuesta de solución, filosóficos, comunicar una experiencia, opinión.
Resultados preliminares****	

*Para definir las categorías de la dimensión-contexto, se toma como base a los procesos propuestos en el estándar ISO/IEC 12207 (ISO, 1995), que agrupa los procesos en principales y procesos de soporte y procesos de la organización.

**Para clasificar los aportes que realizan los estudios, se considera si el estudio realizó la construcción de una métrica, modelo, herramienta, práctica, artefactos, método/metodología, proceso. Además se considera una categoría ¨No contribuye¨, cuyo significado se debe a que en el estudio no se logró la construcción de ninguno de los elementos contemplados en esta Dimensión.

***Para clasificar el tipo de investigación de los artículos primarios del SMS se considera el sistema de clasificación propuesto por Wieringa et al. (Wieringa, 2006), el cual emplea seis categorías.

****En esta dimensión se consideran los modelos de procesos de desarrollo (RUP, XP, Scrum, DSDM, etc.), los modelos de calidad (CMMI, Competisoft, ISO y sus variantes, etc.), los modelos de ciclos de vida (cascada, espiral, prototipo, etc.). Es importante aclarar que la inclusión de una categoría nueva en la dimensión, se ha realizado en función de su aparición en los estudios relevantes.

1.2.4. Síntesis de los datos extraídos.

Para resumir los datos extraídos de los artículos primarios del SMS, se realiza una síntesis cuantitativa, mediante el uso de gráficos (sección 1.2.4.1.) y luego se realiza un análisis de los artículos primarios obtenidos para dar respuesta a cada una de las preguntas de investigación (sección 1.2.4.2.).

1.2.4.1. Síntesis cuantitativa del SMS.

En el gráfico 1, se presenta la cantidad de artículos primarios por el año de publicación.

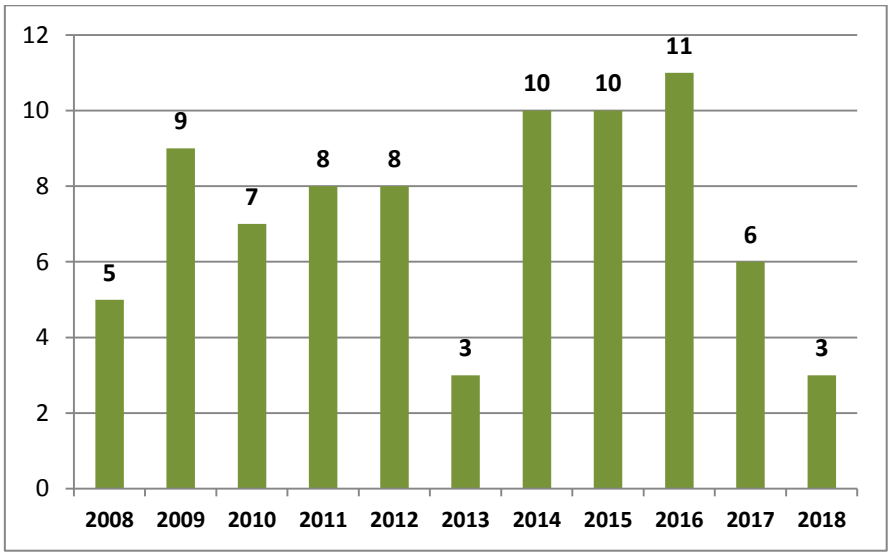

Gráfico 1. Cantidad de artículos primarios por año de publicación.

En el gráfico 2, se presenta que la mayoría de artículos encontrados han sido publicados en congresos, que representa un 71 % del total y los artículos publicados en revistas representan un 29 %.

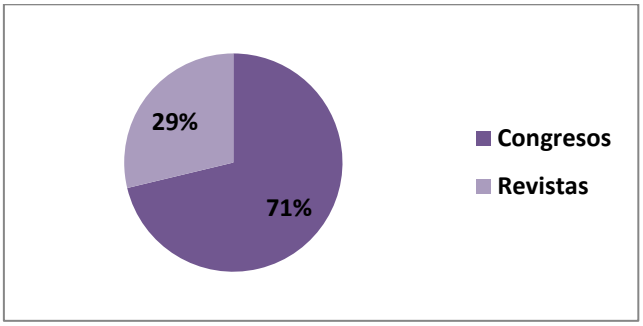

Gráfico 2. Artículos por tipo de publicación.

En el gráfico 3, se presenta una distribución de los congresos respecto a los continentes en los que han sido realizados. El mayor porcentaje corresponde a Europa con un 26 %, seguido del continente Asiático con un porcentaje del 23 %, América del Norte como América Latina y Del Caribe presentan un porcentaje del 21 % y por último el continente Africano con un 7 % y Oceanía con un 2 %.

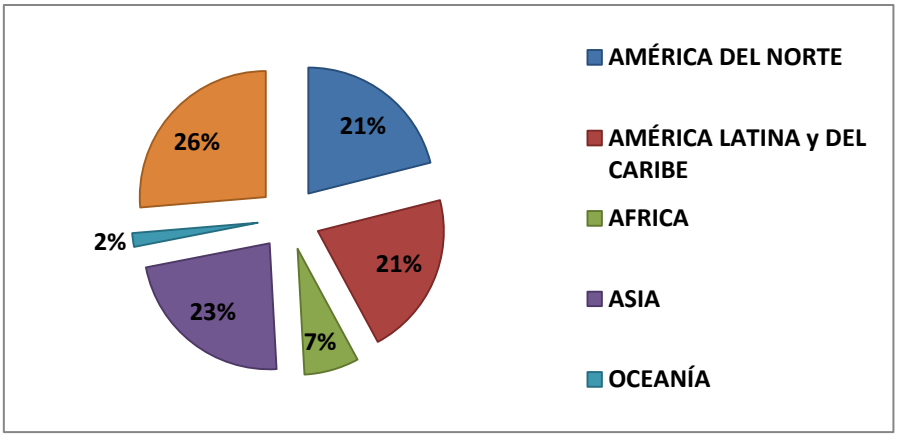

Gráfico 3. Distribución porcentual de los congresos respecto al continente de realización.

A continuación se presenta el Gráfico 4, con la cantidad de artículos primarios clasificados por las revistas en las que han sido publicados.

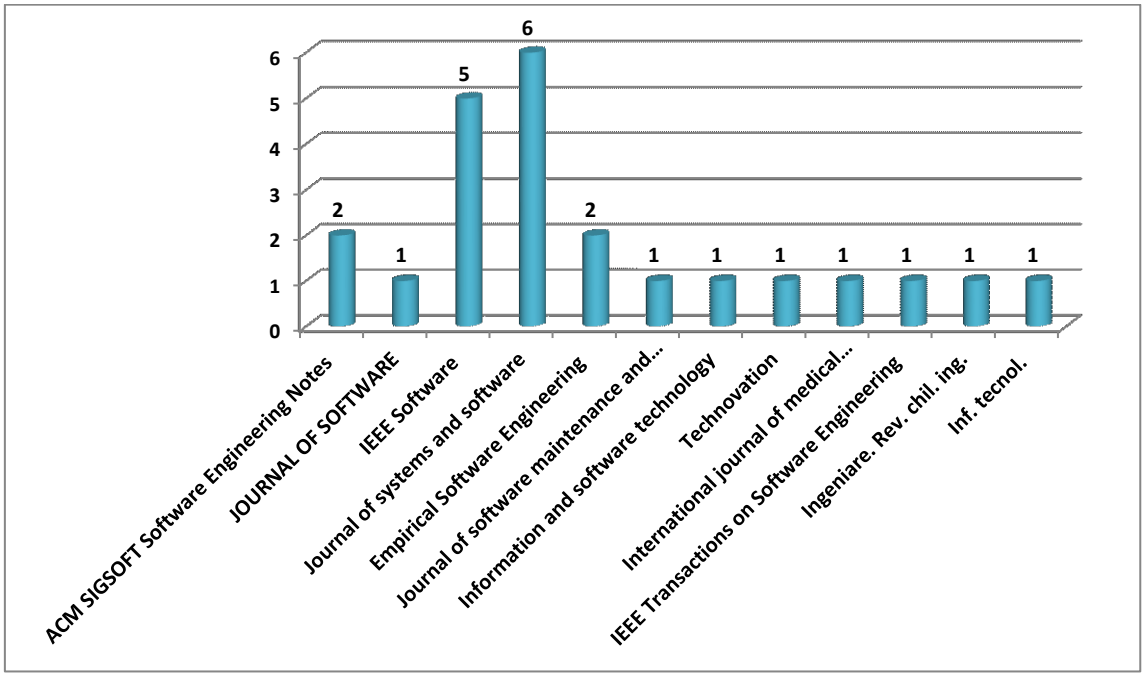

Gráfico 4. Revistas con los estudios primarios considerados.

En el gráfico 5, se presenta la cantidad de artículos primarios según la cadena de búsqueda definida en la sección 1.1.3.1. Se visualiza que la cadena más representativa para este estudio es ¨software development¨ dado que se han encontrado 28 estudios primarios.

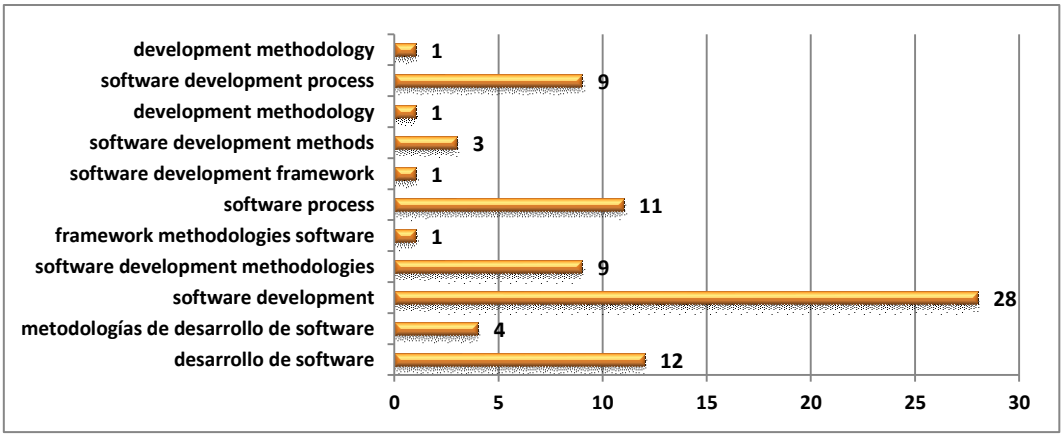

Gráfico 5. Cantidad de artículos según la cadena de búsqueda empleada.

1.2.4.2. Respuestas a las preguntas de investigación.

En esta sub-tarea, se da respuesta a cada una de las preguntas de investigación que motivaron la realización del SMS, definidas en la sección 1.1.2.

PI1: ¿Qué metodologías permiten llevar adelante la construcción de un producto software?

La respuesta a esta pregunta, se resume en el Gráfico 6, en el cual se visualizan las metodologías de desarrollo de software más estudiadas. Es importante aclarar que no todas son metodologías, que algunas de ellas son frameworks o métodos; pero en esta investigación se les dará el mismo tratamiento y se las consideran como guías para la construcción de un producto software.

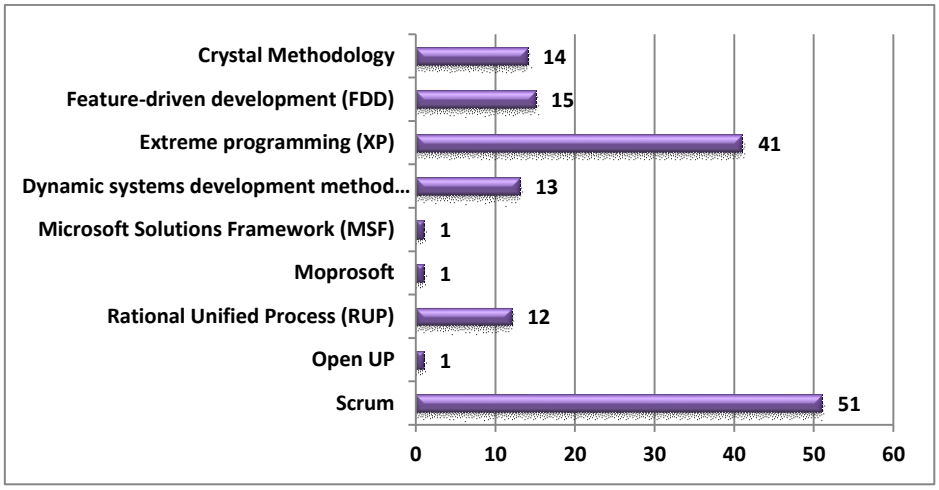

Gráfico 6. Cantidad de artículos por metodologías de desarrollo.

Con el desarrollo de este SMS, además de evidenciar las metodologías de desarrollo más estudiadas, se logró conocer los modelos de ciclo de vida tratados en los estudios primarios (ver Gráfico 7). También, se logró conocer los modelos y estándares de calidad tratados en los estudios, siendo estos para los procesos, los productos y las personas que conforman el equipo (ver Gráfico 8).

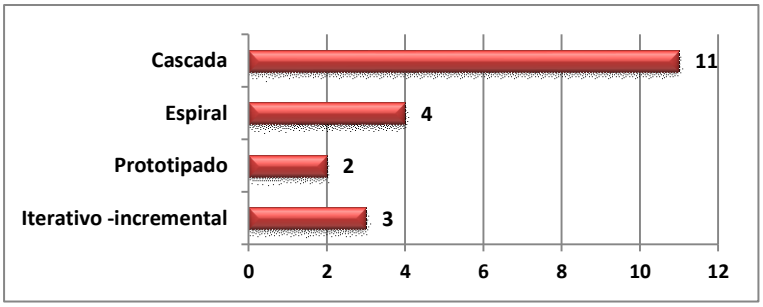

Gráfico 7. Cantidad de artículos por modelos de ciclo de vida.

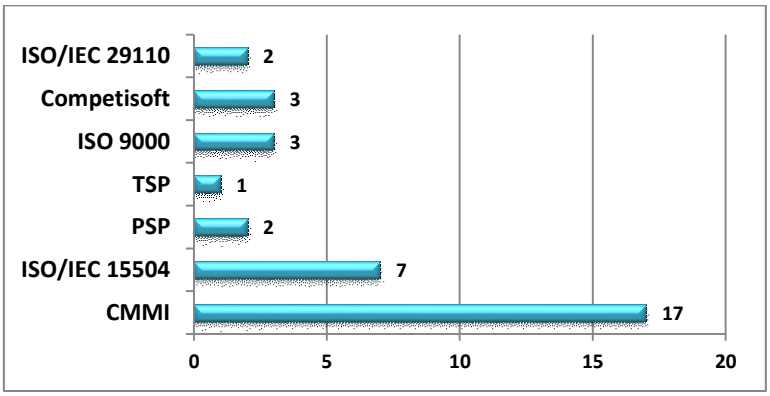

Gráfico 8. Cantidad de artículos por modelos y estándares de calidad.

PI2: ¿En qué actividades de la construcción del producto software, procesos de gestión o procesos de soporte se centra la investigación?.

En la sección 1.2.3, se definieron las categorías de la dimensión-contexto, la cual toma como base los procesos propuestos en el estándar ISO/IEC 12207 (ISO/IEC 12207, 1995). Los procesos, se agrupan en procesos principales, procesos de soporte y procesos de la organización. La respuesta a esta pregunta se resume en un primer gráfico, en el cual se visualiza la cantidad de los artículos según los tres grupos de procesos (Gráfico 9). Se ha considerado una categoría denominada ¨Abarcativo¨ para agrupar a los estudios que han considerado más de un tipo de proceso principal en su análisis.

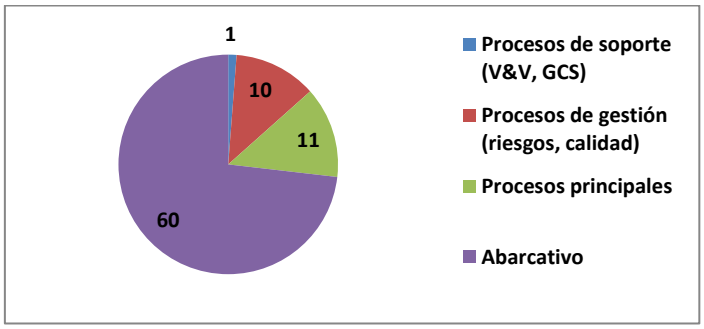

Gráfico 9. Estudios por grupos de procesos (principales, de soporte y de la organización).

Para el grupo de procesos principales, se presenta el Gráfico 10 con la cantidad de estudios primarios según los procesos principales (requisitos, arquitecturas y diseño, desarrollo, pruebas, despliegue o implantación, mantenimiento).

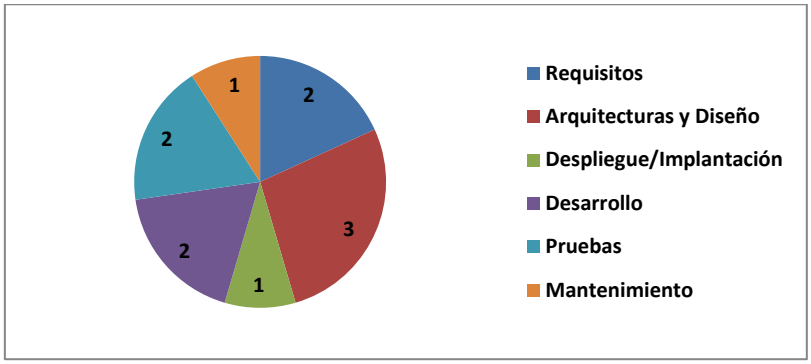

Gráfico 10. Estudios por grupos de procesos principales.

PI3:¿Qué aportes realizan los trabajos?

En base a la clasificación planteada para la dimensión-contribución (en la sección 1.2.3.), se presentan en el gráfico11, los tipos de aportes propuestos en los estudios primarios del SMS. La mayor cantidad de artículos primarios corresponden a la categoría ¨No contribuye¨ con un total de 45 artículos. Las propuestas de construcción más significativas que surgen en los estudios analizados, se presentan en los modelos con un total de 14 artículos y métodos /metodologías con un total de 11 artículos.

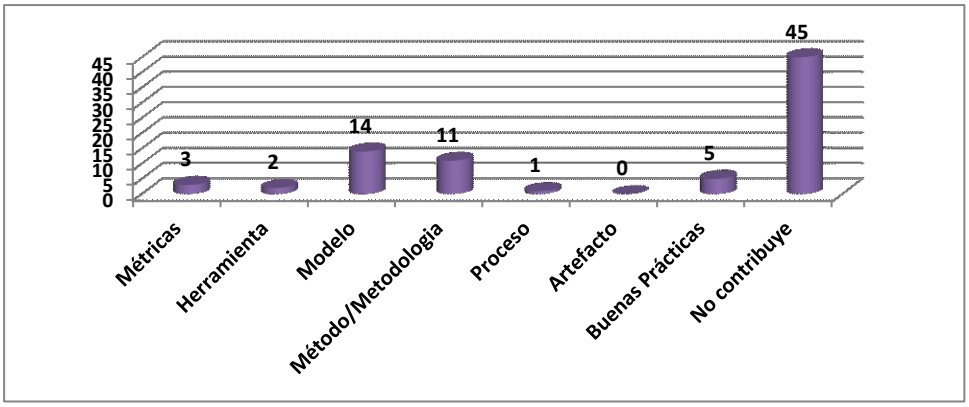

Gráfico 11. Cantidad de artículos por Tipos de aportes.

PI4: ¿Cuáles son los objetivos perseguidos en la investigación sobre las metodologías de desarrollo?

En el gráfico 12, se presenta una distribución de los tipos de investigación de los estudios primarios según la propuesta de Wieringa et al. (Wieringa et al., 2006). Se visualiza que los tipos de investigación de propuestas de solución (28 artículos) y comunicar una experiencia (27 artículos) son los más representativos respecto al total de estudios.

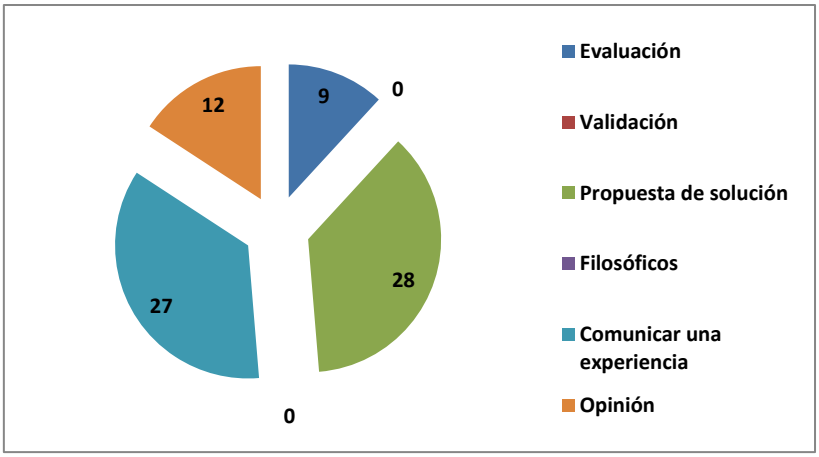

Gráfico 12. Artículos por tipos de investigación.

1.3. Reporte de los resultados del SMS.

El desarrollo del mapeo sistemático de la literatura permitió identificar cuáles de las metodologías de desarrollo de software son las más estudiadas por la comunidad científica y las más utilizadas en la industria del software. Al analizar los datos extraídos de los artículos primarios, se concluye:

- Las metodologías más consideradas en la comunidad científica como así también en la industria del software, son Scrum con un 63% y XP con un 51% respecto al total de estudios primarios considerados.
- En un segundo lugar, se presentan FDD (19%), Cristal (17%), DSDM (16%) y RUP (15%), estas metodologías cubren un porcentaje de representatividad entre el 15% y 20% respecto al total de estudios.
- En un tercer lugar con una representatividad inferior al 15%, se encuentran OpenUP (1%), Moprosoft (1%) y MSF (1%).
- El desarrollo de software presenta una tendencia hacia las metodologías ágiles en el período considerado para este estudio. No obstante, se visualiza que la única metodología robusta que logra un porcentaje de representatividad equivalente al de algunas metodologías ágiles, es RUP.

2. Comparativa de metodologías de desarrollo.

En esta sección, se definen las metodologías a considerar en el análisis comparativo y se plantea la justificación de su elección (sección 2.1.), se usan los lineamientos del método DESMET (Kitchenham B., 1996) adaptados a esta investigación, para la evaluación de las metodologías consideradas (sección 2.2.) y por último se presentan los resultados obtenidos y su interpretación (sección 2.3.).

2.1. Metodologías de desarrollo consideradas.

Con base en el reporte del SMS realizado en la sección 1, se consideran para el análisis comparativo, las metodologías de desarrollo cuya cota de representatividad en los estudios primarios es igual o superior al 15 %. Estas son Scrum, XP, FDD, Cristal, DSDM y RUP. No obstante, se decide considerar Métrica V3 y MoProsoft, dado que el Grupo de Investigación de Ingeniería de Software del III-LIDI realiza investigación conjunta con Universidades de España y de México.

2.2. Análisis comparativo.

El análisis comparativo de las metodologías de desarrollo de software, se descompone en tres partes, en la primera se definen las características a evaluar en cada una de las metodologías consideradas como así también su justificación (sección 2.2.1.). En una segunda parte, se describen las particularidades de las metodologías (sección 2.2.2.) En la tercera y última parte, se presenta la evaluación de las metodologías y la interpretación de los resultados obtenidos (sección 2.2.3.)

2.2.1. Definición y justificación de las características a evaluar.

Para realizar la comparativa de las metodologías se utiliza el método DESMET, del cual se selecciona el método la evaluación cualitativa basada en el análisis de características (Kitchenham B., 1995). Este análisis comparativo, tiene el propósito de dar respuesta a la siguiente pregunta de investigación: ¿Todos los elementos del proceso de implantación de un sistema informático se encuentran cubiertos en las metodologías de desarrollo?.

En la introducción de este reporte, se definió a que se considera *proceso de implantación* en esta línea de investigación. La implantación de sistemas informáticos es uno de los procesos genéricos de cualquier modelo de proceso de software. En base a la definición de proceso de software propuesta por Acuña et. al., …¨como un conjunto de actividades para gestionar, desarrollar y mantener sistemas de software¨. En otras palabras, el proceso de software se centra en las tareas de construcción y no en los productos de salida. La definición de un proceso software debe especificar no sólo las actividades, sino también las técnicas para realizar las tareas, los actores que ejecutan las actividades, sus roles y los artefactos producidos (Acuña et al., 2005)…¨. Al proceso de implantación, se lo estudiará de manera integral considerando los elementos que componen un proceso de software. Las características a utilizar en el análisis comparativo son los elementos de un proceso de software, los cuales se listan a continuación:

- Fases/Actividades/Tareas
- Herramientas
- Técnicas/Prácticas
- Artefactos (Insumos/Productos)
- Rol
- Competencias

La elección de este elemento para el proceso en estudio, se debe a que a este proceso se le resta importancia por tratarse del último eslabón de la cadena de procesos de software. Esta desvalorización del proceso se refleja en que el profesional al que se le asigna el rol de implantador o responsable de la entrega del producto software al cliente, no posee las competencias socio-técnicas necesarias para desempeñarse en dicho proceso.

A continuación se presenta un glosario de términos de las características (elementos del proceso de software) a considerar en la evaluación (ver Tabla 9):

Tabla 9. Glosario de términos de las características del análisis comparativo.

Término	Definición
Actividad	¨Conjunto de tareas específicas asignadas para su realización a un rol o más roles¨. (MoProSoft, 2005). Sommerville (Sommerville I., 2011) define proceso de software como una serie de actividades relacionadas que conduce a la elaboración de un producto de software. En el análisis comparativo, para unificar la terminología a la característica se la denominará: ¨Fases/Actividades/Tareas¨.
Herramienta	El Diccionario de la Lengua Española define el término "herramienta" como ¨…Instrumento, por lo común de hierro o acero, con que trabajan los artesanos…" . Si bien esta definición no refleja totalmente a que denominamos herramienta en el contexto de la Ingeniería de Software, podemos considerar de ella dos componentes, instrumento y que se utiliza para trabajar. En la construcción del producto software se requiere de herramientas, se pueden mencionar las herramientas de modelado, las cuales permiten la creación de los modelos necesarios del sistema que se está estudiando. Con la evolución de la Ingeniería de Software, se ha introducido un amplio abanico de diferentes tipos de programas que se utilizan para ayudar a las actividades del proceso de software, como el análisis de requerimientos, el modelado de sistemas, la depuración y las pruebas, a este grupo de herramientas se las denomina herramientas CASE (Ingeniería de Software Asistida por Computadora) (Panizzi et al., 2016).
Técnica/Prácticas	Técnica: conjunto de heurísticas y procedimientos que se apoyan en estándares, es decir, que utilizan una o varias notaciones específicas en términos de sintaxis y semántica y cumplen unos criterios de calidad en cuanto a la forma de obtención del producto asociado. (Pae , 2001). Práctica: representa un medio para la consecución de unos objetivos específicos de manera

	rápida, segura y precisa, sin necesidad de cumplir unos criterios o reglas preestablecidas. (Pae, 2001).
Artefactos (Insumos7Productos)	En Moprosoft (Secretaría de Economía México, 2005) se define producto como cualquier elemento que se genera en un proceso. Esta dimensión de análisis considera los artefactos productos y los artefactos insumos ya que son necesarios como inputs para la realización de las actividades del proceso (Panizzi et al., 2016).
Rol	MoProSoft (Secretaría de Economía México, 2005) define el concepto de rol como responsable por un conjunto de actividades de uno o más procesos. Un rol puede ser asumido por una o más personas de tiempo parcial o completo. Panizzi et al, consideran de importancia la definición de roles para la realización de actividades dentro del proceso estudiado así se conoce de antemano las responsabilidades, derechos, aptitudes y habilidades necesarias para el rol requerido (Panizzi et al., 2017).
Competencias	En Acuña (Acuña S., et al., 2005) plantean una clasificación de competencias en función de cómo se movilizan los recursos. Las competencias duras se centran en los aspectos técnicos que se requieren para llevar a cabo una actividad, generalmente se expresan en términos de conocimientos, destrezas y habilidades. Las competencias blandas se centran en los comportamientos, en los rasgos personales entre los cuales se pueden mencionar el liderazgo, la colaboración con los demás, la integridad y la persuasión. Esta característica considera las competencias duras y las competencias blandas.

2.2.2. Descripción de las metodologías.

En esta sección, se presenta una descripción de las particularidades de las metodologías consideradas, se focaliza en el proceso de implantación como proceso dentro del proceso de desarrollo de software. Se consideraran los aspectos de gestión del proyecto software por las interacciones que se presentan con el proceso de implantación.

2.2.2.1. SCRUM.

Schwaber Ken y Sutherland Jeff definen a SCRUM como un marco de trabajo para el desarrollo y el mantenimiento de productos complejos dentro del cual las personas pueden afrontar complejos problemas adaptativos, a la vez que entregan productos del máximo valor posible de forma productiva y creativa. Scrum es un modelo de desarrollo ágil caracterizado por: (i) adoptar una estrategia de desarrollo incremental, en lugar de la planificación y ejecución completa del producto, (ii) basar la calidad del resultado más en el conocimiento tácito de las personas en equipos auto-organizados, que en la calidad de los procesos empleados, (iii) solapamiento de las diferentes fases del desarrollo, en lugar de realizarlas una tras otra en un ciclo secuencial o de cascada (Schwaber K. y Sutherland J, 2016).

Este marco de trabajo consiste en los Equipos Scrum y en sus roles (Dueño del Producto, Equipo de Desarrollo, Scrum Master y los Interesados), eventos (el Sprint, la reunión de Planificación, Scrum Diario, la Revisión de Sprint y la Retrospectiva de Sprint), artefactos (Pila de producto, la Pila de Sprint y el Incremento) y reglas asociadas (Palacio J., 2015).

Esta metodología presenta aspectos vinculados a la gestión del proyecto, los cuales se los visualiza en la planificación, en las técnicas de estimación, en las revisiones de los sprints. Los aspectos vinculados al proceso de implantación no se explicitan en esta metodología.

2.2.2.2. Programación Extrema (XP).

XP es uno de los métodos ágiles más conocido y ampliamente utilizado, (en Inglés Extreme Programming). El nombre lo acuño Beck (2000) debido a que el enfoque se desarrolló llevando a niveles ¨extremos¨ las prácticas reconocidas, como el desarrollo ágil (Sommerville I., 2011).

Beck (Beck K., 2011).define un conjunto de cinco valores que establecen el fundamento para todo trabajo realizado como parte de XP: comunicación, simplicidad, retroalimentación, valentía y respeto. Todos estos valores se usan como un motor de actividades para actividades, acciones y tareas específicas de XP.

La metodología XP engloba una serie de reglas y prácticas que ocurren en el contexto de cuatro actividades estructurales: planeación, diseño, codificación y pruebas (Pressman R., 2010). XP implica varias prácticas que se ajustan a los principios de los métodos ágiles, entre ellas: la planificación

incremental, entregas pequeñas, refactorización, programación en parejas, diseño simple, integración continua, etc. (Beck K., 2011).

El proceso de implantación estudiado no se visualiza en este método, solamente se visualizan algunos aspectos de gestión del proyecto entre ellos la planificación y la estimación del esfuerzo a partir de las historias de usuario.

Se enfatiza el proceso como un conjunto de ciclos anidados. En la mayoría de los proyectos se perciben siete ciclos: 1) el proyecto, 2) el ciclo de entrega de una unidad, 3) la iteración, 4) la semana laboral, 5) el período de integración, de 30 minutos a tres días, 6) el día de trabajo, 7) el episodio de desarrollo de una sección de código, de pocos minutos a pocas horas.

2.2.2.3. Desarrollo basado en funcionalidades (FDD).

El desarrollo basado en funcionalidades (en inglés, feature-driven development, FDD) es un enfoque de desarrollo ágil de software desarrollado por Jeff De Luca y Peter Coad.

FDD consiste en cinco procesos secuenciales durante los cuales se diseña y construye el sistema (Palmer et al., 2002). La parte iterativa soporta desarrollo ágil con rápidas adaptaciones a cambios en requerimientos y necesidades del negocio. Cada fase del proceso tiene un criterio de entrada, tareas, pruebas y un criterio de salida. Generalmente, la iteración de un rasgo insume de una a tres semanas. Las fases son: desarrollo de un modelo general, construcción de la lista de rasgos, planeamiento por rasgo y diseño por rasgo y construcción por rasgo.

FDD consiste en un conjunto de "mejores prácticas", el modelado de objetos del dominio, el desarrollo por rasgo, la propiedad individual de clases (código), los equipos de rasgos, pequeños y dinámicamente formados, la inspección, los builds regulares, administración de configuración y el reporte de progreso.

Esta metodología presenta un conjunto de roles agrupados en tres categorías: roles claves, roles de soporte y roles adicionales. Los seis roles claves de un proyecto son: 1) administrador del proyecto; 2) arquitecto jefe (puede dividirse en arquitecto de dominio y arquitecto técnico); 3) manager de desarrollo, que puede combinarse con arquitecto jefe o manager de proyecto; 4) programador jefe; 5) propietarios de clases y 6) experto de dominio. Los cinco roles de soporte comprenden: 1) administrador de entrega; 2) abogado/guru de lenguaje, que conoce a la perfección el lenguaje y la tecnología; 3) ingeniero de construcción; 4) herramientista (toolsmith) y 5) administrador del sistema, que controla el ambiente de trabajo o productiza el sistema cuando se lo entrega. Los tres roles adicionales son los de verificadores, encargados del despliegue y escritores técnicos. Un miembro de un equipo puede tener otros roles a cargo, y un solo rol puede ser compartido por varias personas. (Ambler S., 2018).

FDD suministra un conjunto de artefactos para la planificación y control de los proyectos; se pueden mencionar; las vistas de desarrollo, vistas de planificación, reportes de progreso, reportes de tendencia, vista de plan entre otros.

Esta metodología propone fases, técnicas, artefactos y roles para la gestión del proyecto. Respecto al proceso de implantación, presenta roles que cubren de manera parcial alguna de las actividades del mismo.

2.2.2.4.Metodologias Crystal.

Las metodologías Crystal fueron creadas por Alistair Cockburn (Cockburn A., 1997). La familia de metodologías Crystal se centra en la eficiencia y habitabilidad (las personas pueden vivir con él e incluso usarlo) como componentes de la seguridad del proyecto.

La familia Crystal dispone un código de color para señalar la complejidad de una metodología: cuanto más oscuro un color, más "pesado" es el método. Cuanto más crítico es un sistema, más rigor se requiere. El código cromático se aplica a una forma tabular elaborada por Cockburn que se usa en muchos Métodos Ágiles para situar el rango de complejidad al cual se aplica una metodología.

Los métodos se llaman Crystal evocando las facetas de una gema: cada faceta es otra versión del proceso, y todas se sitúan en torno a un núcleo idéntico. Hay cuatro variantes de metodologías: Crystal Clear ("Claro como el cristal") para equipos de 8 o menos integrantes; Amarillo, para 8 a 20; Naranja, para 20 a 50; Rojo, para 50 a 100. (Cockburn A., 2001).

Los siete valores o propiedades de Crystal Clear son: entrega frecuente, comunicación osmótica, mejora reflexiva, seguridad personal., foco, fácil acceso a usuarios expertos y ambiente técnico con prueba automatizada, management de configuración e integración frecuente.

En cuanto a las técnicas, utilizan las denominadas ¨entrevistas de proyectos¨, ¨talleres de reflexión¨., ¨Planeamiento Blitz¨, ¨Estimación Delphi con estimaciones de pericia¨, ¨encuentros diarios de pie¨. . ¨miniatura de procesos¨, "hora extrema", ¨gráficos de quemado¨., programación lado a lado¨.

Existen ocho roles en CC: Patrocinador, Usuario Experto, Diseñador Principal, Diseñador-Programador, Experto en Negocios, Coordinador, Verificador, Escritor. En la versión Crystal Naranja, se agregan aún más roles: Diseñador de IU, Diseñador de Base de Datos, Experto en Uso, Facilitador Técnico, Analista/Diseñador de Negocios, Arquitecto, Mentor de Diseño, Punto de Reutilización.

Este conjunto de metodologías presentan actividades, roles, artefactos, técnicas para el proceso de gestión del proyecto. Respecto al proceso implantación presenta algunos roles y artefactos.

2.2.2.5. Método de Desarrollo de Sistemas Dinámico (DSDM).

DSDM es una metodología de desarrollo de software, (en Inglés, Dynamic Systems Development Method) originalmente basada en las técnicas de Desarrollo Rápido de Aplicaciones (en Inglés Rapid Application Development, RAD). DSDM es un enfoque iterativo e incremental que enfatiza la participación continua del usuario (Pressman R., 2010). DSDM comprende un marco que muestra las fases y cómo se relacionan entre sí. Este modelo de proceso es utilizado por cada proyecto para derivar su ciclo de vida.

El modelo del proceso ágil, llamado ciclo de vida DSDM tiene tres ciclos iterativos distintos precedidos de dos actividades adicionales al ciclo de vida: Estudio de Viabilidad, Estudio del negocio, Iteración del modelo funcional, Diseño e iteración de la construcción e Implementación (Pressman R., 2010).

Esta metodología presenta aspectos vinculados a la gestión del proyecto, los cuales se visualiza en el estudio de viabilidad, la estimación, las fases de pre-proyecto y post-proyecto, la gestión de configuración. Los aspectos vinculados al proceso de implantación se visualizan en la fase de despliegue de la metodología.

2.2.2.6. Proceso Unificado de Rational (RUP).

El RUP (en Inglés, Rational Unified Process) es un proceso de desarrollo de software iterativo e incremental (Péraire C et al., 2007). RUP comprende no solamente las disciplinas de Ingeniería sino que adiciona las disciplinas de soporte.

El proceso de ciclo de vida de RUP se divide en cuatro fases: Incepción, Elaboración, Construcción y Transición; a su vez estas fases se dividen en iteraciones. A través de las fases se desarrollan en paralelo nueve workflows o disciplinas: Modelado de Negocios, Requerimientos, Análisis & Diseño, Implementación, Prueba, Despliegue, Gestión de Configuración & Cambio, Gestión del Proyecto y Entorno.

Los aspectos relacionados al proceso de implantación son tratados en el Flujo de despliegue propuesto por el RUP cuyo objetivo es producir con éxito distribuciones del producto y distribuirlo a los usuarios. La Fase de Transición es la cuarta fase del ciclo de vida del software, se ocupa de que el software sea puesto en manos de la comunidad de usuarios (Péraire C et al., 2007).

El flujo de Despliegue se relaciona con las disciplinas de soporte propuestas por este marco de trabajo ya que requiere de ellas insumos para su ejecución como así también le retribuye productos a las mismas.

2.2.2.7. Métrica V3.

MÉTRICA Versión 3 (PAe, 2001) ofrece a las organizaciones un instrumento útil para la sistematización de las actividades que dan soporte al ciclo de vida del software. Cubre distintos tipos de desarrollo: estructurado y orientado a objetos. A través de interfaces, facilita la realización de los procesos de apoyo u organizativos: Gestión de Proyectos, Gestión de Configuración, Aseguramiento de Calidad y Seguridad.

En METRICA versión 3, se ha considerado la Fase de Implantación y Aceptación del Sistema (cuya abreviatura es IAS) como fase equivalente al proceso estudiado en este trabajo. La Fase de Implantación y Aceptación del Sistema propuesta en esta metodología tiene como objetivo principal la entrega y aceptación del sistema en su totalidad, y la realización de todas las actividades necesarias para el paso a producción del mismo (PAe, 2001).

La Interfaz de Gestión de Proyectos propone tres tipos de actividades: Actividades de Inicio del Proyecto, Actividades de Seguimiento y Control y Actividades de Finalización del Proyecto, Cierre y registro de la documentación de gestión.

La Interfaz de Seguridad tiene el objetivo de incorporar en los sistemas de información, mecanismos de seguridad adicionales a los que se proponen en la propia metodología.

La Interfaz de Gestión de la Configuración consiste en la aplicación de procedimientos administrativos y técnicos durante el desarrollo del sistema de información y su posterior mantenimiento.

Las actividades propias de la Interfaz de Calidad están orientadas a verificar la calidad de los productos.

2.2.2.8. MoProsoft.

MoProSoft (Secretaría de Economía México, 2005) presenta tres categorías de procesos: Alta Dirección, Gerencia y Operación que reflejan la estructura de una organización. Este modelo presenta aspectos vinculados a la gestión de proyectos, las cuales se las visualiza en el proceso de Administración de Proyectos Específicos. Las actividades vinculadas al proceso de implantación no se explicitan en este modelo de proceso.

2.2.3. Evaluación de las metodologías.

En la Tabla 10, se presenta el grado de cumplimiento de las características consideradas en el análisis del proceso de implantación dentro del proceso de desarrollo de software (ver en la Tabla, Implantación) y para los procesos de gestión de proyectos de software (ver en la Tabla, Gestión del Proyecto). El grado de cumplimiento se considera en tres niveles: total (“T”), parcial (“P”) o nulo (“N”).

	Metodologías/Procesos															
	SCRUM		Programación Extrema (XP)		Desarrollo basado en funcionalidades (FDD)		Metodologías Crystal		Desarrollo de Sistemas Dinámico (DSDM)		Proceso Unificado de Rational (RUP)		METRICA Versión 3		MoProSoft	
	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto	Implantación	Gestión del Proyecto
Fases / Actividades/ Tareas	N	T	N	P	N	T	N	T	P	T	T	T	T	T	N	T
Herramientas	N	P	N	N	N	N	N	N	N	P	T	T	N	N	N	N
Técnicas/Prácticas	N	T	N	P	N	T	N	T	N	P	N	N	T	T	N	N
Artefactos (Insumos/Productos)	N	T	N	P	N	T	P	T	P	T	T	T	T	T	N	T
Roles	N	T	N	P	P	T	P	T	N	T	T	T	T	T	N	T
Competencias	N	N	N	N	N	N	N	N	N	T	N	N	N	N	N	T

Tabla 10. Resultados del grado de cumplimiento de las características.

Los resultados de la evaluación de características a los cuales se ha arribado en la Tabla 10, permiten formular las siguientes conclusiones parciales:

- En relación a la característica ¨*Fases/Actividades/Tareas*¨, en Scrum, XP, FDD, la familia de metodologias Crystal y Moprosoft se evidencia la ausencia de actividades para el proceso de implantación. En XP se denota un grado de cumplimiento parcial para el proceso de gestión del proyecto. DSDM presenta un grado de cumplimiento parcial para el proceso de implantación. Scrum, FDD, la familia de las metodologías Crystal, DSDM, RUP, Métrica versión 3 y Morposoft presentan un grado de cumplimiento total respecto al proceso de gestión del proyecto. RUP y Métrica versión 3, cubren el proceso de implantación con un grado de cumplimiento total.
- En la relación a la característica ¨*Herramientas*¨, en Scrum, XP, FDD, la familia de metodologías Crystal, DSDM, Metrica versión 3 y Moprosoft se evidencia la ausencia de la propuesta de herramientas para el proceso de implantación. XP, FDD, la familia de las metodologías Crystal, Métrica versión 3 y Moprosoft no presentan herramientas para el proceso de gestión. Scrum y DSDM presentan un grado de cumplimiento parcial para el proceso de gestión del proyecto. En RUP se denota un grado de cumplimiento total para el proceso de implantación y para el proceso de gestión del proyecto.
- En relación a la característica ¨*Técnicas/Prácticas*¨, el conjunto de metodologías evaluadas presentan un grado de cumplimiento nulo respecto a las técnicas y prácticas para el proceso de implantación. Scrum y DSDM presentan un grado de cumplimiento parcial para el proceso de gestión. En RUP se denota un grado de cumplimiento total para el proceso de implantación y proceso de gestión.
- Respecto a la característica ¨*Artefactos (Insumos/Productos*)¨, en Scrum, XP, FDD y Moprosoft se denota un grado de cumplimiento nulo para el proceso de implantación. En la familia de Crystal y DSDM se presenta un grado de cumplimiento parcial para el proceso de implantación. XP presenta un grado de cumplimiento parcial para el proceso de gestión del proyecto. RUP y Métrica versión 3 cubren de manera total el proceso de implantación. Scrum, FDD, la familia de las metodologías Crystal, DSDM, RUP, Metrica versión 3 y Moprosoft presentan un grado de cumplimiento total para el proceso de gestión del proyecto.
- En relación a la característica ¨*Roles*¨, Scrum, XP, DSDM y Moprosoft no presentan roles para el proceso de implantación. FDD y la familia de metodologías Crystal denotan un grado de cumplimiento parcial para el proceso de implantación. XP presenta un grado de cumplimiento parcial para el proceso de gestión del proyecto. En RUP y Métrica versión 3, se evidencia un grado de cumplimiento total respecto al proceso de implantación. Scrum, FDD, la familia de las metodologías Crystal, DSDM, RUP, Métrica versión 3 y Moprosoft presentan un grado de cumplimiento total pata el proceso de gestión del proyecto.
- Respecto a la característica ¨*Competencias*¨, ninguna de las metodologías consideradas enuncian competencias para el proceso de implantación. En DSDM y Moprosoft se evidencia un grado de cumplimiento total para el proceso de gestión de proyectos.

3. Conclusiones.

Se ha presentado un proceso de revisión sistemática de la literatura mediante un método de investigación de Ingeniería de software, mapeo sistemático de la literatura (SMS). Este permitió sistematizar la evidencia empírica de las metodologías de desarrollo de software más estudiadas por la comunidad científica y las más utilizadas en la industria del software. Una vez identificadas las metodologías de desarrollo, se realizó una evaluación basada en características en base a los lineamientos del método DESMET. Se consideraron como características de análisis a los elementos de un proceso de software: ¨fases/actividades/tareas¨, ¨herramientas¨, ¨técnicas/prácticas¨, ¨artefactos (insumos/productos)¨, ¨roles¨ y ¨competencias¨. Esto permitió identificar de qué manera las metodologías de desarrollo soportan el proceso de implantación de sistemas informáticos como así también el proceso de gestión del proyecto con el que interactúa. Se logró evidenciar un conjunto de vacancias que presenta el proceso de implantación como proceso dentro del proceso de desarrollo de software.

Bibliografía.

Acuña S., Juristo. N. Moreno A., Mon A. (2005). A software Process Model Handbook for incorporating people `s capabilities. United States of America: Springer Science+Business Media, Inc.

Ambler Scott. Agile Modeling (AM) Home Page. http://www.agilemodeling.com/essays/fdd.htm. Página vigente al 15/11/2018.

Beck, K. 2004. Extreme Programming Explained: Embrace Change, 2da. Edición. Addison-Wesley.

CMMI® para Desarrollo, Versión 1.3 (2010). Mejora de los procesos para el desarrollo de mejores productos y servicios. TECHNICAL REPORT. Editorial Universitaria Ramón Areces.

Cockburn Alistair. (1997). "Software development as Community Poetry Writing. Cognitive, cultural, sociological, human aspects of software development". Annual Meeting of the Central Ohio Chapter of the ACM.

Cockburn Alistair (2000). ¨Balancing Ligthness with Sufficiency¨.

Cockbun, A. (2001). "Agile Software Development". Addison Wesley.

Cockburn A. "Crystal Clear. A human-powered methodology for small teams, including The Seven Properties of Effective Software Projects". Borrador. Humans and Technology, versión del 27 de febrero de 2002.

Genero Bocco Marcela, Cruz-Lemus Jose Antonio y Piattini Velthuis Mario. (2014). Métodos de investigación en ingeniería del software. Editorial Ra-Ma, 2014. ISBN 978-84-9964-507-0.

IEEE Glossary. (1990). IEEE Standard 610.12-1990. Standard Glossary of Software Engineering Terminology. New York, USA: The Institute of Electrical and Electronics Engineers.

IBM. Rational Software. (2007). Péraire C., Edwards M, Fernandes A., Mancin E. y Carroll K. The IBM Rational Unified Process for Systems.

Instituto Nacional de Tecnologías de la Comunicación (INTECO). (2009). ¨Guía: Ingeniería del software: metodologías y ciclos de vida¨. Laboratorio Nacional de Calidad del Software.

ISO,ISO/IEC Standard 12207:1995 (1995).Software Life Cycle Procesess, Ginebra (Suiza). International Organization for Standarization.

Kitchenham, B, Budgen, D., Brereton, P. (2016). Evidence-Based Software Engineering and Systematic Reviews. CRC Press.

Kitchenham, B. and Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering. Versión 2.3 EBSE-2007-01.

Kitchenham, B. (1995). Evaluating Software Engineering Methods and Tools. Parts 1 to 12. SIGSOFT Notes. Starting December 1995.

Kitchenham, B. A. (1996). "DESMET: A Method for Evaluating Software Engineering Methods and Tools," Department of Computer Science, University of Keele, Technical Report TR96-09, August 1996. ISSN:1353-7776

Osterweil, L. (1987). Software processes are software too. En Proceedings of the 9th international conference on Software Engineering, (pp.2-13). Los Alamitos, CA: IEEE Computer Society.

MoProSoft. (2005). Modelo de procesos para la industria del software. Secretaría de Economía México. Versión 1.3.

PAe, Métrica versión.3. (2001). Portal de Administración Electrónica. Gobierno de España. https://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Metrica_v3

Palacio J. (2015). Gestión de Proyecos SCRUM Manager (Scrum Manager I y II). http://www.scrummanager.net. Página vigente al 21/12/2018

Palmer S. and Felsing J. (2002). A Practical Guide to Feature-Driven Development. Upper Saddle River, NJ, Prentice-Hall.

Panizzi, M., Hossian, A., García-Martínez, R. (2016). Implantación de Sistemas: Estudio Comparativo e Identificación de Vacancias en Metodologías Usuales. Libro de Actas del XXII Congreso Argentino de Ciencias de la Computación. Pág. 546-555. ISBN 978-987-733-072-4. Universidad Nacional de San Luis. SEDICI: http://sedici.unlp.edu.ar/handle/10915/55716

Panizzi, M., Bertone R., Hossian A. (2017). Proceso de Implantación de Sistemas Informáticos – Identificación de vacancias en Metodologías Usuales. Libro de Actas de la V Conferencia Iberoamericana de Computación Aplicada CIACA 2017. Pag.207 -215. ISBN 978-989-8533-70-8. Vilamoura, Algarve, Portugal.

Piattini, M., García, F., García, I., & Pino, F. (2012). Calidad de sistemas de información. México DF, México: Ra-Ma.

Pressman R. 2010. Ingeniería del software: Un enfoque práctico. Mc Graw Hill. 7ma Edición.

Schwaber K. y Sutherland J. (2016). La Guía Definitiva de Scrum: Las reglas del juego. www.scrum.org

Sierra Jose Maria, Vizcaíno Aurora, Genero Marcela, Piattini Mario. (2017). A systematic mapping study about socio-technical congruence. Information and Software Technology http://dx.doi.org/10.1016/j.infsof.2017.10.004.

Wieringa, R., Maiden, N.A.M, Mead, N.R. and Rolland, C. (2006) Requirements engineering paper classification and evaluation criteria: a proposal and a discussion. Requirements Engineering, 11(1), pp 102-107.

Apéndice A. Listado de estudios primarios utilizados en el SMS.

Abheeshta Putta (2018). Scaling Agile Software Development to Large and Globally Distributed Large-scale Organizations. In ICGSE '18: ICGSE '18: 13th IEEE/ACM International Conference on Global Software Engineering , May 27–29, 2018, Gothenburg, Sweden. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3196369.3196386

Aitken, A., & Ilango, V. (2013). A Comparative Analysis of Traditional Software Engineering and Agile Software Development. 2013 46th Hawaii International Conference on System Sciences, 4751-4760. DOI:10.1109/HICSS.2013.31

Alahyari, H., Berntsson Svensson, R., & Gorschek, T. (2017). A study of value in agile software development organizations. Journal of Systems and Software, 125, 271–288. DOI: 10.1016/j.jss.2016.12.007

Amaral, L. M. G., & Faria, J. P. (2010). A gap analysis methodology for the team software process. In Quality of Information and Communications Technology (QUATIC), 2010 Seventh International Conference on the (pp. 424-429). IEEE**.** DOI: 10.1109/QUATIC.2010.78

Ambler, S. W. (2009). Scaling agile software development through lean governance. In Software Development Governance, 2009. SDG'09. ICSE Workshop on (pp. 1-2). IEEE. DOI: 10.1109/SDG.2009.5071328

Alfonso Fuggetta, Elisabetta Di Nitto. (2014). Software Process. FOSE'14, May 31 – June 7, 2014, Hyderabad, India. ISBN: 978-1-4503-2865-4 doi: 10.1145/2593882.2593883

Balasubramanian, L., & Mnkandla, E. (2016). An evaluation to determine the extent and level of Agile Software Development Methodology adoption and implementation in the Botswana Software Development Industry. In Advances in Computing and Communication Engineering (ICACCE), 2016 International Conference on (pp. 320-325). IEEE. DOI: 10.1109/ICACCE.2016.8073768

Bannerman, P. L., Hossain, E., & Jeffery, R. (2012). Scrum practice mitigation of global software development coordination challenges: A distinctive advantage?. In System Science (HICSS), 2012 45th Hawaii International Conference on (pp. 5309-5318). IEEE. DOI: 10.1109/HICSS.2012.512

Barrios Walter et al. (2011). SCRUM: Experiencia de Aplicación en una Empresa de Desarrollo de Software del NEA. Actas del XVII CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACIÓN (CACIC 2011). Universidad Nacional de La Plata, Provincia de Buenos Aires. 10 al 14 de octubre de 2011. SEDICI: http://sedici.unlp.edu.ar/handle/10915/18745

Basri S., O'Connor R.V. (2010) Understanding the Perception of Very Small Software Companies towards the Adoption of Process Standards. In: Riel A., O'Connor R., Tichkiewitch S., Messnarz R. (eds) Systems, Software and Services Process Improvement. EuroSPI 2010. Communications in Computer and Information Science, vol 99. Springer, Berlin, Heidelberg. ISBN 978-3-642-15666-3

Bioul, G. J. A., Escobar, F., Álvarez, M., Nardin, A., & Ricci Aparicio, E. (2010). Metodologías Ágiles, análisis de su implementación y nuevas propuestas. Libro de Actas del XVI Congreso Argentino de Ciencias de la Computación. p. 597-606. Universidad de Morón, Provincia de Buenos Aires. 18 al 22 de octubre de 2010. ISBN: 978-950-9474-49-9 ISBN: 978-950-9474-49-9. SEDICI: http://sedici.unlp.edu.ar/handle/10915/18803

Bollati, V. A., Gaona, G., Pletsch, L. C., Gonnet, S., & Leone, H. (2017, September). The state of agile development adoption in Argentine software companies. In Computer Conference (CLEI), 2017 XLIII Latin American (pp. 1-10). IEEE. DOI: 10.1109/CLEI.2017.8226394

Capitas Cristina, Hurtado Nuria, Ruiz Mercedes. Sim-XPerience: (2014). Simulación Basada en Agentes Aplicada al Desarrollo de Software con XP. Actas de las XIX Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2014). Págs. 117- 130- Cádiz, septiembre de 2014. Sitio: 11705/JISBD/2014/019

Choi, S., Kim, D. K., & Park, S. (2012, June). ReMo: a recommendation model for software process improvement. In Proceedings of the International Conference on Software and System Process (pp. 135-139). IEEE Press. DOI: 10.1109/ICSSP.2012.6225957

Choudhary, B., & Rakesh, S. K. (2016, February). An approach using agile method for software development. In Innovation and Challenges in Cyber Security (ICICCS-INBUSH), 2016 International Conference on (pp. 155-158). IEEE. DOI: 10.1109/ICICCS.2016.7542304

Chow, T., & Cao, D. B. (2008). A survey study of critical success factors in agile software projects. Journal of systems and software, 81(6), 961-971. https://doi.org/10.1016/j.jss.2007.08.020

Colombani, M. A., Pérez, M. M., & Falappa, M. A. (2016). Metodologías para el desarrollo de software en PYMES. Libro de Actas del XXII Congreso Argentino de Ciencias de la Computación. Universidad Nacional de San Luis. San Luis. 3 al 7 de octubre de 2016.ISBN: 978-987-733-072-4. SEDICI: http://sedici.unlp.edu.ar/handle/10915/55716

Colla, P. E. (2016). Uso de opciones reales para evaluar la contribución de metodologías KANBAN en desarrollo de software. Simposio Argentino de Ingeniería de Software (ASSE 2016)-JAIIO 45. Universidad Nacional de Tres de Febrero. 5 al 9 de septiembre de 2016. ISSN: 2451-7593 2451-7593 SEDICI: http://sedici.unlp.edu.ar/handle/10915/57079

Cusumano, M. A., MacCormack, A., Kemerer, C. F., & Crandall, W. (2009). Critical decisions in software development: Updating the state of the practice. IEEE software, 26(5), 84-87. DOI: 10.1109/MS.2009.124

Dapozo, G. N., Greiner, C. L., Irrazábal, E., Medina, Y., Ferraro, M. D. L. A., & Lencina, B. (2015). Características del desarrollo de software en la ciudad de Corrientes. Libro de Actas del Congreso Argentino de Ciencias de la Computación. UNNOBA Junín, del 5 al 9 de octubre de 2015. ISBN: 978-987-3724-37-4. SEDICI: http://sedici.unlp.edu.ar/handle/10915/50028

del Nuevo, E., Piattini, M., & Pino, F. J. (2011). Scrum-based methodology for distributed software development. In Global Software Engineering (ICGSE), 2011 6th IEEE International Conference on (pp. 66-74). IEEE. http://doi.org/10.1016/j. jss.2010.03.077

Dingsøyr, T., Moe, N. B., Fægri, T. E., & Seim, E. A. (2018). Exploring software development at the very large-scale: a revelatory case study and research agenda for agile method adaptation. Empirical Software Engineering, 23(1), 490-520. https://link.springer.com/article/10.1007/s10664-017-9524-2

Domah, D., & Mitropoulos, F. J. (2015). The NERV methodology: A lightweight process for addressing non-functional requirements in agile software development. In SoutheastCon 2015 (pp. 1-7). IEEE. DOI: 10.1109/SECON.2015.7133028

Dzamashvili Fogelström, N., Gorschek, T., Svahnberg, M., & Olsson, P. (2010). The impact of agile principles on market-driven software product development. Journal of software maintenance and evolution: Research and practice, 22(1), 53-80. ISSN: 1532-060X

Elhag, A. A., Elshaikh, M. A., Mohamed, R., & Babar, M. I. (2013). Problems and future trends of software process improvement in some Sudanese software organizations. In Computing, Electrical and Electronics Engineering (ICCEEE), 2013 International Conference on (pp. 263-268). IEEE. DOI: 10.1109/ICCEEE.2013.6633945

Esquivel Salvador (2016). Communication Issues in Agile Software Development. Libro de Actas del XXII Congreso Argentino de Ciencias de la Computación. Universidad Nacional de San Luis. San Luis. 3 al 7 de octubre de 2016.ISBN: 978-987-733-072-4. SEDICI: http://sedici.unlp.edu.ar/handle/10915/55716

Esterkin, V., & Pons, C. (2017). Evaluación de calidad en el desarrollo de software dirigido por modelos. Ingeniare. Revista chilena de ingeniería, 25(3), 449-463. ISSN 0718-3305. http://dx.doi.org/10.4067/S0718-33052017000300449.

Estler, H.-C., Nordio, M., Furia, C. A., Meyer, B., & Schneider, J. (2012). Agile vs. Structured Distributed Software Development: A Case Study. 2012 IEEE Seventh International Conference on Global Software Engineering. DOI: 10.1109/ICGSE.2012.22

Fontana, R. M., Fontana, I. M., da Rosa Garbuio, P. A., Reinehr, S., & Malucelli, A. (2014). Processes versus people: How should agile software development maturity be defined?. Journal of Systems and Software, 97, 140-155. https://doi.org/10.1016/j.jss.2014.07.030

Gasca-Hurtado, G. P., Álvarez, M. C. G., Manrique-Losada, B., & Arias, D. M. (2015). Diagnostic on teaching-learning of software desing by using the Personal Software Process framework. In Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on (pp. 1-7). IEEE. DOI: 10.1109/CISTI.2015.7170385

Ghandi, L., Silva, C., Martínez, D., & Gualotuña, T. (2017). Mobile application development process: A practical experience. In Information Systems and Technologies (CISTI), 2017 12th Iberian Conference on (pp. 1-6). IEEE. DOI: 10.23919/CISTI.2017.7975825

Hossain, E., Bannerman, P. L., & Jeffery, R. (2011). Towards an understanding of tailoring scrum in global software development: a multi-case study. In Proceedings of the 2011 International Conference on Software and Systems Process (pp. 110-119). ACM. ISBN: 978-1-4503-0730-7 DOI: 10.1145/1987875.1987894

Ikoma, M., Ooshima, M., Tanida, T., Oba, M., & Sakai, S. (2009). Using a validation model to measure the agility of software development in a large software development organization. In Software Engineering-Companion Volume, 2009. ICSE-Companion 2009. 31st International Conference on (pp. 91-100). IEEE. DOI: 10.1109/ICSE-COMPANION.2009.5070967

Ionica, A., Leba, M., & Dovleac, R. (2017). A QFD based model integration in Agile software development. In Information Systems and Technologies (CISTI), 2017 12th Iberian Conference on (pp. 1-6). IEEE. DOI: 10.23919/CISTI.2017.7975995

Ji, F., & Sedano, T. (2011). Comparing extreme programming and Waterfall project results. In Software Engineering Education and Training (CSEE&T), 2011. DOI: 10.1109/CSEET.2011.5876129

Jha, M. M., Vilardell, R. M. F., & Narayan, J. (2016). Scaling Agile Scrum Software Development: Providing Agility and Quality to Platform Development by Reducing Time to Market. In Global Software Engineering (ICGSE), 2016 IEEE 11th International Conference on (pp. 84-88). IEEE. DOI: 10.1109/ICGSE.2016.24

Kettunen, P. (2009). Adopting key lessons from agile manufacturing to agile software product development—A comparative study. Technovation, 29(6-7), 408-422. https://doi.org/10.1016/j.technovation.2008.10.003

Khan, M. I., Qureshi, M. A., & Abbas, Q. (2010). Agile methodology in software development (SMEs) of Pakistan software industry for successful software projects (CMM framework). In Educational and Network Technology (ICENT), 2010 International Conference on (pp. 576-580). IEEE. DOI: 10.1109/ICENT.2010.5532104

Könnölä, K., Suomi, S., Mäkilä, T., Jokela, T., Rantala, V., & Lehtonen, T. (2015). Agile methods in embedded system development: Multiple-case study of three industrial cases. Journal of Systems and Software, 118, 134-150. https://doi.org/10.1016/j.jss.2016.05.001

Kunz, M., Dumke, R. R., & Zenker, N. (2008). Software metrics for agile software development. In Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on (pp. 673-678). IEEE. DOI: 10.1109/ASWEC.2008.4483261

Kuranuki, Y., Ushio, T., Yasui, T., & Yamazaki, S. (2014). A new business model of custom software development for agile software development. Proceedings of the International Workshop on Innovative Software Development Methodologies and Practices - InnoSWDev 2014. ISBN: 978-1-4503-3226-2 DOI: 10.1145/2666581.2666584

Kusumasari, T. F., Supriana, I., Surendro, K., & Sastramihardja, H. (2011, July). Collaboration model of software development. In Electrical Engineering and Informatics (ICEEI), 2011 International Conference on (pp. 1-6). IEEE. DOI: 10.1109/ICEEI.2011.6021769

Larrucea, X., O'Connor, R. V., Colomo-Palacios, R., & Laporte, C. Y. (2016). Software process improvement in very small organizations. IEEE Software, 33(2), 85-89. DOI: 10.1109/MS.2016.42

Letelier, P., & Penadés, M. C. (2016). AgileRoadmap: Un modelo y estrategia para implantación de prácticas ágiles. Actas de las XXI Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2016). Universidad de Salamanca, España. Setiembre 2016. Biblioteca Sistedes: http://biblioteca.sistedes.es/biblioteca/conferencias/jisbd/jisbd-2016

Livermore, J. A. (2008). Factors that Significantly Impact the Implementation of an Agile Software Development Methodology. JSW, 3(4), 31-36. DOI: 10.4304/jsw.3.4.31-36

Lopez, D. M., & Blobel, B. G. (2009). A development framework for semantically interoperable health information systems. International journal of medical informatics, 78(2), 83-103.

Mahmud, D. M., & Abdullah, N. A. S. (2015, December). Reviews on agile methods in mobile application development process. In Software Engineering Conference (MySEC) (pp. 161-165). DOI: 10.1109/MySEC.2015.7475214

Martelo, R. J., Jiménez-Pitre, I., & Moncaris González, L. (2017). Guía Metodológica para el Mejoramiento del Desarrollo de Software a través de la Aplicación de la Técnica Árboles de problemas. Información. tecnológica. vol.28 no.3 La Serena 2017. ISSN 0718-0764.

Martinez N., Ramon H., & Bertone, R. (2012). Aplicabilidad de competisoft a partir de un método ágil como Scrum: Un caso práctico. Libro de Actas del XVIII Congreso Argentino de Ciencias de la Computación. Universidad Nacional del Sur, Bahía Blanca. 8 al 12 de Octubre de 2012. ISBN 978-987-1648-34-4. SEDICI: http://sedici.unlp.edu.ar/

Matharu, G. S., Mishra, A., Singh, H., & Upadhyay, P. (2015). Empirical study of agile software development methodologies: A comparative analysis. ACM SIGSOFT Software Engineering Notes, 40(1), 1-6. DOI: 10.1145/2693208.2693233

Mendes Calo, K., Estévez, E. C., & Fillottrani, P. R. (2009). Un framework para evaluación de metodologías ágiles. Libro de Actas del XV Congreso Argentino de Ciencias de la Computación. Universidad Nacional de Jujuy, Jujuy. 5 al 9 de octubre de 2009. ISBN: 978-950-34-0756-1 SEDICI http://sedici.unlp.edu.ar/handle/10915/20803

Mnkandla, E. (2009). About software engineering frameworks and methodologies. In AFRICON, 2009. AFRICON'09. (pp. 1-5). IEEE. DOI: 10.1109/AFRCON.2009.5308117

Meng, C. X. (2009). A goal-driven measurement model for software testing process. In Information Technology and Applications, 2009. IFITA'09. 2009 WRI World Congress on Software Engineering. DOI: 10.1109/WCSE.2009.27

Nan, N., & Harter, D. E. (2009). Impact of budget and schedule pressure on software development cycle time and effort. IEEE Transactions on Software Engineering, 35(5), 624-637. DOI: 10.1109/TSE.2009.18

Papadopoulos, G. (2015). Moving from Traditional to Agile Software Development Methodologies Also on Large, Distributed Projects. Procedia - Social and Behavioral Sciences, 175, 455-463. https://doi.org/10.1016/j.sbspro.2015.01.1223.

Peres, A. L., & Meira, S. L. (2015). Towards a framework that promotes integration between the UX design and SCRUM, aligned to CMMI. In Information Systems and Technologies (CISTI), 2015 10th Iberian Conference on (pp. 1-4). IEEE. DOI: 10.1109/CISTI.2015.7170443

Quelal, R. E., Villavicencio, M., & Mendoza, L. E. (2018). A survey of agile software development methodologies in Ecuador. In 2018 13th Iberian Conference on Information Systems and Technologies (CISTI) (pp. 1-6). IEEE. DOI: 10.23919/CISTI.2018.8399186

Qureshi, M. R. J. (2012). Agile software development methodology for medium and large projects. IET software, 6(4), 358-363. DOI: 10.1049/iet-sen.2011.0110

Rahimian, V., & Ramsin, R. (2008). Designing an agile methodology for mobile software development: A hybrid method engineering approach. In Research Challenges in Information Science, 2008. RCIS 2008. Second International Conference on (pp. 337-342). IEEE. DOI: 10.1109/RCIS.2008.4632123

Sarkan, H. M., Ahmad, T. P. S., & Bakar, A. A. (2011). Using JIRA and Redmine in requirement development for agile methodology. In Software Engineering (MySEC), 2011 5th Malaysian Conference in (pp. 408-413). IEEE. DOI: 10.1109/MySEC.2011.6140707

Scharff, C., Gotel, O., & Kulkarni, V. (2010). Transitioning to Distributed Development in Students. In 2010 Fifth International Conference on Software Engineering Advances (pp. 388-394). IEEE. DOI: 10.1109/ICSEA.2010.66

Schindler, C. (2008, December). Agile software development methods and practices in Austrian IT-industry: Results of an empirical study. In Computational Intelligence for Modelling Control & Automation, 2008 International Conference on (pp. 321-326). IEEE. DOI: 10.1109/CIMCA.2008.100

Soundararajan, S., Arthur, J. D., & Balci, O. (2012). A methodology for assessing agile software development methods. In Agile Conference (AGILE), 2012 (pp. 51-54). IEEE. DOI: 10.1109/Agile.2012.24

Straccia, Luciano; Pytel, Pablo; Pollo-Cattaneo, María Florencia (2016). Metodología para el desarrollo de software en proyectos de I+D en el nivel universitario basada en Scrum. Libro de Actas del XXII Congreso Argentino de Ciencias de la Computación. Universidad Nacional de San Luis. San Luis. 3 al 7 de octubre de 2016. ISBN: 978-987-733-072-4. SEDICI: http://sedici.unlp.edu.ar/handle/10915/55716

Strode, D. E., Huff, S. L., Hope, B., & Link, S. (2012). Coordination in co-located agile software development projects. Journal of Systems and Software, 85(6), 1222-1238. DOI:10.1016/j.jss.2012.02.017

Sultania, A. K. (2015, February). Developing software product and test automation software using Agile methodology. In Proceedings of the 2015 Third International Conference on Computer, Communication, Control and Information Technology (C3IT) (pp. 1-4). IEEE. DOI: 10.1109/C3IT.2015.7060120

Tuli, A., Hasteer, N., Sharma, M., & Bansal, A. (2014). Empirical investigation of agile software development: cloud perspective. ACM SIGSOFT Software Engineering Notes, 39(4), 1-6. DOI: 10.1145/2632434.2632447

Valenzuela, J., & Pavlich-Mariscal, J. A. (2014, September). Hacia un Modelo de Madurez para apoyar el Desarrollo de Software Dirigido por Modelos. In Computing Colombian Conference (9CCC), 2014 9th (pp. 162-167). IEEE. DOI: 10.1109/ColumbianCC.2014.6955350

Veselá, V., & Krbecek, M. (2016, October). Development methodologies of mobile applications. In Interactive Mobile Communication, Technologies and Learning (IMCL), 2016 International Conference on (pp. 3-4). IEEE. DOI: 10.1109/IMCTL.2016.7753760

Vijayasarathy, L. R., & Butler, C. W. (2016). Choice of software development methodologies: Do organizational, project, and team characteristics matter?. IEEE software, 33(5), 86-94. DOI: 10.1109/MS.2015.26

Vivas, H. L., Cambarieri, M., García Martínez, N., Muñoz Abbate, H., & Petroff, M. (2013). Un Marco de Trabajo para la Integración de Arquitecturas de Software con Metodologías Ágiles de Desarrollo. Libro de Actas del Congreso Argentino de Ciencias de la Computación. Universidad CAECE, Mar del Plata. 21 al 25 de octubre de 2013. ISBN: 978-987-23963-1-2. SEDICI: http://sedici.unlp.edu.ar/handle/10915/31095

Vlaanderen, K., Jansen, S., Brinkkemper, S., & Jaspers, E. (2011). The agile requirements refinery: Applying SCRUM principles to software product management. Information and software technology, 53(1), 58-70. https://doi.org/10.1016/j.infsof.2010.08.004

Vladimir Rubin, Irina Lomazova, Wil M.P. van der Aalst. (2014). Agile Development with Software Process Mining. International Conference on Communication and Signal Processing (ICSSP'14) May 26–28, 2014, Nanjing, China. DOI: 10.1145/2600821.2600842

Wang, X. (2011, August). The combination of agile and lean in software development: An experience report analysis. In Agile Conference (AGILE), 2011 (pp. 1-9). IEEE. DOI: 10.1109/AGILE.2011.36

Wang, X., Conboy, K., & Cawley, O. (2012). "Leagile" software development: An experience report analysis of the application of lean approaches in agile software development. Journal of Systems and Software, 85(6), 1287-1299. https://doi.org/10.1016/j.jss.2012.01.061

Wei, Q., Danwei, G., Yaohong, X., Jingtao, F., Cheng, H., & Zhengang, J. (2014, September). Research on software development process conjunction of scrum and UML modeling. In Instrumentation and Measurement, Computer, Communication and Control (IMCCC), 2014 Fourth International Conference on (pp. 978-982). IEEE. DOI: 10.1109/IMCCC.2014.206

Yu, X., & Petter, S. (2014). Understanding agile software development practices using shared mental models theory. Information and Software Technology, 56(8), 911-921. Yu, X., & Petter, S. (2014). Understanding agile software development practices using shared mental models theory. Information and Software Technology, 56(8), 911-921. https://doi.org/10.1016/j.infsof.2014.02.010

Zhang, J. M. (2010, February). The software development process methodology of resource-based access control. In Computer and Automation Engineering (ICCAE), 2010 The 2nd International Conference on (Vol. 4, pp. 111-117). IEEE. DOI: 10.1109/ICCAE.2010.5451762