Software Inspection Adoption: a Mapping Study

Darío Macchi
Universidad ORT Uruguay
Montevideo, Uruguay
macchi@uni.ort.edu.uy

Martín Solari
Universidad ORT Uruguay
Montevideo, Uruguay
martin.solari@ort.edu.uy

***Abstract*—In Software Engineering technical literature the references about the benefits of software inspections are abundant. In contrast, some authors raise the problem of low adoption of this process. From this issue a literature review is made to produce a map on most researched topics in the area, factors causing low adoption and possible solutions. Results showed a list of 64 articles selected using a search protocol, which were classified according to a defined taxonomy. The founded factors were codified and a list of solutions founded in the reviewed papers was made. The main conclusion was that most of the factors causing low adoption are related to developers perceptions about the process, lack of training and some characteristics of the process as the rigidity, complexity and the difficulty of connecting the effort made with the final product quality. These factors should be studied in future works.**

Keywords: Systematic mapping study; Software inspections; Low adoption; Factors; Quality

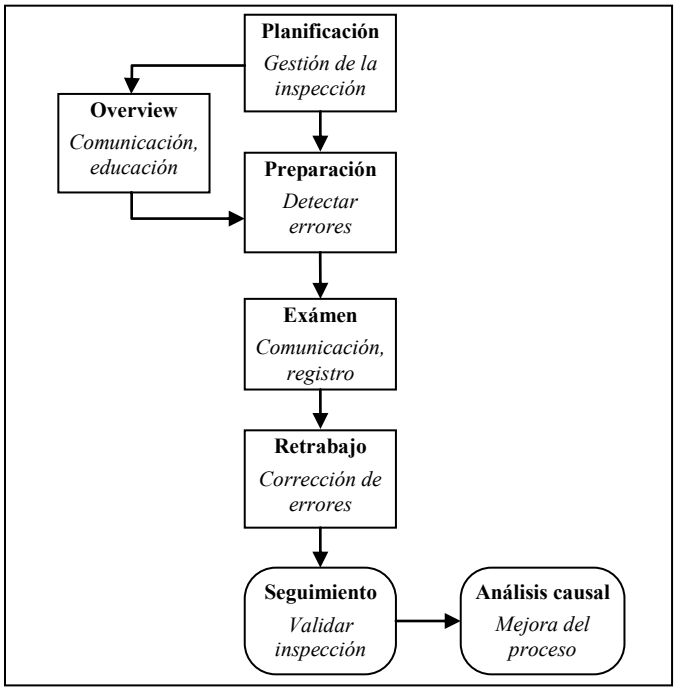
I. INTRODUCCIÓN

Se estima que entre el 50% y el 60% del esfuerzo total de producir software se invierte en tareas de aseguramiento de calidad [1]. Por otra parte, de acuerdo a un estudio realizado en el 2002 por el NIST (*National Institute of Standards and Technology*) la entrega de software defectuoso le cuesta a los Estados Unidos $59.5 miles de millones por año, de los cuales $22.2 miles de millones podrían ahorrarse mejorando la infraestructura de *testing*, revisiones e inspecciones [2].

Humphrey dijo que la calidad de los productos de software depende directamente de la calidad del proceso que los genera [3]. Dentro de estos procesos, los de verificación y validación permiten evaluar si un software cumple con los requerimientos y con el uso previsto. También permiten detectar defectos en los artefactos de software lo que contribuye a aumentar la productividad y reducir el esfuerzo de retrabajo. Estos conceptos no solamente son aplicables al código, sino también a todo artefacto elaborado durante el proceso de desarrollo.

Normalmente cuando se utiliza el término inspección de software se hace en referencia al método introducido por Michael Fagan [4], el cual puede verse en la Figura 1. Esto se debe a que la mayor parte de los métodos de inspección existentes son variantes del método de Fagan. Por ejemplo se quitan o agregan elementos del proceso como en el método de Gilb y Graham [5], se modifica la cantidad de participantes, como en el caso del método *limited logging meeting*, o la distribución en distintas locaciones físicas (*virtual logging meeting*) [6].

Figura 1. Proceso de inspección según IEEE-1028 2008.



Existen distintos métodos manuales (o semi-automáticos) de detectar anomalías que complementan el *testing*, estandarizados por IEEE [7]. La inspección de software es el método más formal en cuanto a sus procedimientos y se realiza con el objetivo de producir resultados más repetibles [4]. Consiste en un proceso de revisión utilizado para mejorar la calidad del software, documentos relacionados y de todo artefacto que sea generado durante el proceso de desarrollo de software. Un método menos formal es el de las revisiones (técnicas y de gestión) cuyo objetivo es examinar uno o varios productos para darlos a conocer y comentar sobre ellos. Por último tenemos el enfoque menos formal de todos llamado *walkthrough* que consiste en un análisis estático guiado que se desarrolla durante una sesión buscando anomalías o violación de estándares. Algunos autores califican al *walkthrough* como una debilitación del proceso formal de inspecciones [8].

978-1-4673-0793-2/12/$31.00 ©2012 IEEE

Según el estándar IEEE 1028-2008 [7], los anteriores conceptos se definen de la siguiente forma:

- Inspección: examen visual de un producto de software con el objetivo de identificar anomalías, errores y desvíos con respecto a estándares o especificaciones.
- Revisión: proceso o reunión donde un producto, varios productos o un proceso de software son presentados al personal del proyecto, gerentes, usuarios, clientes, auditores o cualquier otro interesado a los efectos de ser examinado, comentado o aprobado.
- *Walkthrough*: técnica de análisis estático donde el diseñador o desarrollador guía a los miembros de un equipo de desarrollo a través del producto de software mientras los participantes pueden hacer preguntas o realizar comentarios sobre posibles anomalías, violación de estándares u otros problemas.

Los resultados obtenidos al realizar inspecciones de software son generalmente conocidos y aceptados, encontrándose presentes en muchas publicaciones. Por ejemplo se afirma que este método tiene hasta un 85% de eficiencia en la remoción de defectos, con un promedio del 65% [9]. También se indica que, combinado con prácticas habituales de *testing*, se puede llegar a reducir hasta en un factor de 10 la cantidad de defectos detectados [8]. También se asegura que existen otros beneficios menos cuantificables como permitir la identificación y priorización de áreas de código que requieren una actualización continua, promover el trabajo en equipo e identificar las causas raíces de ciertos defectos a través del análisis causal de los mismos. Otro beneficio es el de ser un medio excelente de transferencia tecnológica y de entrenamiento en estándares, material técnico, cultura y en el propio proceso de inspección [10][11].

Este trabajo se encuentra motivado por el problema mencionado por algunos autores acerca de la baja adopción de procesos de inspección de software [12] [13] en contraposición con la abundancia de reportes positivos respecto a su efectividad en la remoción de defectos. El objetivo de este artículo es obtener evidencia sobre la existencia de este problema y generar un mapa de factores que dificultan la adopción de inspecciones de software. Para ello se realiza un mapeo sistemático de la literatura donde, además de alcanzar los objetivos antes planteados, se busca clasificar los trabajos de investigación recientes del área. De ésta forma también se busca establecer si la comunidad científica ha planteado inquietudes similares a las que motivan a este artículo en los últimos años.

Un mapeo sistemático de la literatura es un tipo de estudio secundario (estudio basado en el análisis de investigaciones previas). Su objetivo es determinar el alcance de la investigación realizada sobre un tema de investigación específico y clasificar conocimiento, a diferencia de una revisión sistemática que busca responder a una pregunta de investigación específica [14][15].

El resto del documento se estructura de la siguiente forma: la sección 2 plantea los trabajos previos a este y definiciones generales; en la sección 3 se explica el método de investigación y los detalles del mapeo sistemático de la literatura llevado a cabo; la sección 4 muestra los resultados de la investigación, (tanto la clasificación de los artículos como el procesamiento de los mismos); en la sección 5 se discuten los hallazgos realizados en la sección anterior y su relación con los objetivos generales del trabajo; por último, la sección 6 resume las conclusiones a las que se llegó durante la realización de este trabajo.

II. TRABAJOS RELACIONADOS

Es comúnmente aceptado que las inspecciones de software son de gran utilidad a la hora de producir software de mejor calidad. Por ejemplo, Aurum *et al.* mencionan como principal beneficio que puede aplicarse a cualquier artefacto generado durante el proceso de desarrollo de software [16]. Laitenberger dice que es un método probado de detección de defectos en artefactos de software ni bien el artefacto es creado [17]. Otros afirman que es uno de los métodos más efectivos de aseguramiento de la calidad dentro de la Ingeniería de Software [18].

Por otro lado, hay autores como Wiegers, que afirman que muy pocos profesionales conocen los métodos de inspección de software y muchos menos realizan inspecciones efectivas [19]. Iisakka *et al.* dicen que no todas las empresas tienen el poder para implementar inspecciones eficientemente y que personalmente nunca vieron ninguna empresa seguir la técnica formal de Fagan[1] [20]. En un estudio centrado en el retrabajo como factor de costo significativo en el desarrollo de software, se afirma que si bien empresas líderes del sector industrial realizan inspecciones (los beneficios se encuentran verificados y documentados), la industria en general no realiza inspecciones [8].

La contraposición de evidencia acerca de los beneficios de las inspecciones de software y la baja adopción en la industria ha sido tratada por varios investigadores. Por ejemplo, Stewart y Priven, realizando un análisis exhaustivo acerca de la evolución de los métodos de inspección, se cuestionan la falta de adopción por parte de la industria y llegan a listar una serie de causas y posibles soluciones [21]. Weller, haciendo un análisis acerca del ROI (*Return of Investment*) de las inspecciones de software, plantea su asombro al ver el número de organizaciones de desarrollo de software que no hacen uso de estos métodos [22].

Otro trabajo relevante es el libro *High Quality, Low Cost Software Inspection* [12], el cual aborda el tema de inspecciones desde un punto de vista práctico planteando algunas causas subyacentes al problema de la falta de adopción. Esta visión práctica se complementa con el trabajo de Ciolkowski *et al.*, quiénes motivados por el mismo problema, realizaron una encuesta para conocer cómo y de que forma la industria realiza inspecciones. Dicha encuesta confirma que el grupo de empresas participantes realizan inspecciones regularmente pero estas no se llevan a cabo de forma sistematizada [13].

Por último, la revisión sistemática realizada por Kollanus *et al.* clasifica los artículos referentes a inspecciones de software

[1] Técnica que da origen al concepto de inspección de software.

publicados desde 1980 a 2008 y analiza el volumen de trabajos dentro de cada clase de la taxonomía planteada. Esta taxonomía clasifica los trabajos en tres clases: los que tratan temas técnicos, los que tratan aspectos gerenciales de las inspecciones y otros. Luego cada clase se divide en subclases para poder organizar el conocimiento. La principal conclusión a la que dichos autores arribaron es que se deben realizar más estudios empíricos para validar el efecto de las distintas propuestas teóricas en la práctica [1].

III. MÉTODO DE INVESTIGACIÓN

El mapeo sistemático de la literatura es un método definido para construir clasificaciones y conducir análisis temáticos a los efectos de obtener un mapa visual del conocimiento existente dentro de un tema amplio [15]. El análisis de los resultados se realiza categorizando los hallazgos y contando la frecuencia de publicaciones dentro de cada categoría para determinar la cobertura de las distintas áreas de un tema de investigación específico. La información generada se puede combinar para responder a preguntas de investigación más específicas y ahorrar tiempo y esfuerzo de investigación. Para que esto sea posible, los mapeos sistemáticos deben ser de calidad en términos de completitud y rigurosidad. El uso de esta herramienta permite identificar tópicos donde existen suficientes estudios primarios para conducir revisiones sistemáticas y tópicos donde se necesiten generar más estudios primarios [14].

Otro tipo de estudio secundario son las revisiones sistemáticas de la literatura. Las mismas, a diferencia del mapeo sistemático, son utilizadas para encontrar, evaluar y realizar la agregación de toda la evidencia presente en los artículos de investigación relevantes respecto a una pregunta específica de investigación. El objetivo es asegurar que la revisión de la literatura es objetiva, rigurosa y auditable [14]. Sin embargo tiene como principal inconveniente el esfuerzo considerable requerido para su realización [15].

Se decide realizar un mapeo sistemático de la literatura debido a la amplitud de las preguntas necesarias para conocer las investigaciones publicadas recientemente, su relación con la adopción de inspecciones y los factores que pueden estar afectando dicha adopción. El resultado de este estudio presenta un punto de partida para la realización de futuras revisiones sistemáticas sobre las respuestas encontradas. Petersen *et al.* [15] sugieren un procedimiento que consta de 5 etapas: A) Definir preguntas de investigación, B) Realizar la búsqueda literaria, C) Seleccionar estudios, D) Clasificar artículos y E) Extraer y realizar la agregación de datos

A. Preguntas de investigación

Si bien el objetivo general de este estudio se puede resumir en comprender que motiva la baja adopción de inspecciones de software, este objetivo se divide en cuatro preguntas de investigación concretas para obtener un conocimiento más detallado y una visión integral del tema.

Las preguntas de investigación a responder en este estudio son las siguientes:

RQ.1. ¿Qué temas interesan a la comunidad respecto a inspecciones de software dentro de un marco temporal reciente?

RQ.2. ¿Existe evidencia sobre la baja adopción de técnicas de inspecciones de software?

RQ.3. ¿Qué factores son mencionados como causantes de la baja adopción?

RQ.4. ¿Qué soluciones se han planteado al respecto?

B. Fuente de datos y estrategia de búsqueda

La estrategia de búsqueda se elabora teniendo en cuenta el problema de terminología existente respecto a inspecciones de software. La denominación de los distintos procesos de revisión estática de software son frecuentemente utilizados de manera imprecisa y confusa. No existe un acuerdo acerca de qué es un proceso de inspección y qué lo diferencia de otros procesos de revisión de software (*walkthrough*, revisión técnica formal o una revisión de gestión) [16]. Por este motivo la cadena de búsqueda se determina sistemáticamente realizando distintas búsquedas relacionadas con el tema de investigación (Tabla I). De ésta forma se puede refinar el vocabulario y conocer el uso de sinónimos (algunas veces mal utilizados) para poder generar cadenas más potentes.

El motor de búsqueda seleccionado para realizar la búsqueda de documentos es la base de datos SciVerse Scopus[2]. La misma tiene una amplia cobertura de publicaciones del área *computer science* (entre otras) e indexa varios catálogos de publicaciones (incluidas IEEE, ACM, Springer y Elsevier).

Para las búsquedas se tienen en cuenta publicaciones arbitradas del área *computer science*, sin más restricciones en cuanto a las fuentes de los artículos, cubriendo los trabajos del 2007 a la fecha en que se realizó el estudio (Julio/2011). Este período de tiempo fue seleccionado tomando como base el trabajo anterior de Kollanus *et al.* que cubre los trabajos hasta el 2008 [1].

C. Selección de estudios

Para la selección de estudios se definen dos criterios simples.

- Inclusión: se define tener en cuenta todos aquellos trabajos que tratan sobre inspecciones de software.
- Exclusión: se decide quitar aquellos trabajos que no traten de inspecciones de software (ej. que contengan la cadena de búsqueda como un ejemplo de proceso de aseguramiento de calidad), artículos publicados en revistas o actas de conferencias no arbitradas.

Debido a que en un principio el volumen de información con la que se debe trabajar es desconocido, se realiza una búsqueda exploratoria para obtener una primera impresión. Algunas de las distintas cadenas de búsqueda utilizadas no se cuentan en el resultado final debido a dos motivos diferentes. El primero, *high recall* (HR), se da al obtenerse toda la

[2] Accesible a través del portal Timbó, al cual accede la comunidad de investigadores en Uruguay gracias a la Agencia Nacional de Investigación Innovación (ANII).

información relevante más una gran cantidad de información de poca utilidad para la investigación. El segundo, *low precisión* (LP), sucede al ser muy chica la proporción entre la cantidad de documentos relevantes obtenidos en la búsqueda y el total de resultados [23].

TABLA I. CADENAS DE BÚSQUEDA UTILIZADAS (EN NEGRITA LAS TENIDAS EN CUENTA PARA EL MAPEO)

Cadenas de búsqueda	#	Motivo	# aplican	Precisión
"software inspection"	**57**		**41**	**72%**
"software review"	**22**		**12**	**55%**
inspection	~2000	HR,LP		~0%
walkthrough	~200	HR,LP		~0%
walkthrough AND inspection	23	LP		~0%
walkthrough AND verification	**6**		**3**	**50%**
"reading techniques"	**22**		**8**	**36%**
"inspection methods"	167	LP		~0%
Total			**64**	

La precisión que vemos en la tabla se obtiene de calcular el porcentaje entre aquellos trabajos que se encuentran comprendidos por el criterio de inclusión y los que no lo hacen (revisando títulos y abstracts). El resultado final[3] se obtiene de la unión (sin duplicados) de los resultados correspondientes a las cadenas que sí aplican.

D. Clasificación de artículos

Para realizar la clasificación de la Tabla II se utiliza la taxonomía usada por Kollanus *et al*. [1] la cual define los criterios para determinar la pertenencia o no a una categoría que se resumen a continuación.

La clase Vista Técnica incluye trabajos que indican cómo se deben llevar a cabo las inspecciones teniendo en cuenta distintos aspectos del proceso. Una excepción a dicha definición son los trabajos sobre herramientas de inspección. La primera subclase dentro de esta clase es la de Factores de Efectividad que se refiere a los distintos factores que pueden afectar la efectividad del proceso de inspección (en términos de eficacia y eficiencia). La siguiente subclase es la de Técnicas de Lectura que abarca trabajos sobre diferentes formas de encontrar defectos en documentos durante la fase de preparación. En la subclase de Procesos se clasifican aquellas propuestas de modificaciones del proceso tradicional de inspección y nuevos métodos. La última subclase, la de Otros Temas Técnicos incluye aquellos trabajos que entran en la descripción de la clase Vista Técnica pero no se corresponden con ninguna de las subclases anteriores.

La siguiente clase, Vista de Gestión, reúne aquellos trabajos que tratan de aspectos prácticos del proceso de inspección y su influencia en la gestión de otros procesos. La principal subclase es la de Impacto de Inspecciones en Proceso de Desarrollo que trata de los trabajos pertenecientes a las primeras investigaciones realizadas sobre inspecciones, típicamente casos de estudio en empresas específicas. La subclase Otros Temas Técnicos contiene otros trabajos sobre gestión como ser el estudio de métricas, modelos de medición, procesos de control y demás.

La última clase es la de Otros Temas Principales dentro de los cuales aparecen técnicas de estimación de defectos, herramientas y aspectos más integrales del proceso. Por ejemplo, la subclase Vista Integral trata sobre temas que incluyen varias de las clases y subclases de la taxonomía definida. La subclase Estimación de Defectos incluye trabajos sobre formas confiables de determinar el número de defectos presentes en un artefacto luego de ser inspeccionado. En la subclase de Herramientas de Gestión encontramos todos los artículos relacionados con las herramientas computacionales orientadas a facilitar el trabajo de inspección. La subclase Aprendizaje no se encuentra definida en la taxonomía original y fue agregada ya que existen una serie de artículos que tratan sobre gestión de conocimiento y aspectos cognitivos que se desean destacar y separar de la subclase Temas Sin Clasificar.

E. Extracción de datos y síntesis

Las guías de Petersen *et al.* [15] sugieren la exploración de algunas partes del artículo solo en aquellos casos donde el *abstract* no se encuentra bien estructurado o es impreciso. Para esta investigación se decide realizar la lectura completa de cada artículo para poder responder todas las preguntas planteadas ya que el solo hecho de analizar títulos y *abstracts* no basta. También se incluyen en la extracción y síntesis aquellos trabajos que tratan el tema de la adopción directamente (mencionados en la sección 2).

Se elabora una planilla de extracción de datos para cada una de las cadenas de búsqueda consideradas en el estudio donde se registran los hallazgos junto con toda la información bibliográfica correspondiente. También se agrega la clasificación (de la taxonomía) a la que corresponde el artículo según los criterios definidos en la sub-sección anterior junto con una breve justificación. Luego se realiza la unión de todas las planillas en una planilla única quitando los artículos duplicados para luego realizar los cálculos de frecuencias.

IV. RESULTADOS

Luego de aplicados los criterios de inclusión y exclusión y realizada la unión de resultados de las distintas cadenas de búsquedas, se obtiene un total de 64 trabajos que aplican al tema de investigación. Los resultados del mapeo sistemático de la literatura se presentan como respuesta a cada una de las preguntas planteadas en la planificación.

La respuesta a la pregunta RQ.1 sobre los temas de interés para la comunidad en un marco temporal reciente se presenta como una clasificación de los artículos obtenidos en la búsqueda y su resultado puede verse en la Tabla II.

[3] Resultados detallados de la revisión bibliográfica, http://fi.ort.edu.uy/innovaportal/file/2038/1/macchi_mapeoadopcioninspecciones.pdf

TABLA II. RESULTADOS (ABSOLUTOS Y RELATIVOS) CLASIFICADOS SEGÚN TAXONOMÍA.

Clase/subclase	#	Porcentaje
Vista Técnica	***(34)***	***52,3%***
Factores de Efectividad	15	23,1%
Técnicas de Lectura	2	3,1%
Procesos	12	18,5%
Otros Temas Técnicos	5	7,7%
Vista de Gestión	***(6)***	***9,2%***
Impacto de Inspecciones en Proceso de Desarrollo	3	4,6%
Otros Temas de Gestión	3	4,6%
Otros temas Principales	***(25)***	***38,5%***
Vista Integral	3	4,6%
Estimación de Defectos	3	4,6%
Herramientas de Inspección	8	12,3%
Aprendizaje	6	9,2%
Temas sin Clasificar	5	7,7%

Respecto a la pregunta RQ.2 sobre si existe evidencia que confirme la sospecha de la baja adopción de procesos de inspección, la respuesta es que sí. Se encontró que no se ha logrado un amplio uso de inspecciones de software [24], que no han tenido el éxito esperado [12] a pesar de los esfuerzos para mejorar el proceso [25][26] y que existe un gap entre el conocimiento sobre su utilidad y el estado real de la práctica [27]. Existe un gran número de empresas de desarrollo de software que no utilizan inspecciones [22] [27] y las que sí, lo hacen de forma no sistemática y con pocos conocimientos [13] [25]. También hay autores que se cuestionan abiertamente el hecho de que a pesar de los beneficios comprobados de las inspecciones, estas no se practiquen [12] [22] [19].

Para responder a la pregunta RQ.3 acerca de los factores causantes de la baja adopción de inspecciones de software se realiza una nueva lectura de los artículos y se extrae de cada uno los factores que cada autor piensa son los causantes. De los 64 artículos seleccionados y releídos, 20 aportaron factores, obteniéndose una lista de 64 factores en total. Debido al volumen de factores encontrados y la diversidad, se usa una codificación para hacer más manejable la lista. La codificación unifica los términos utilizados para referirse a factores que son conceptualmente lo mismo, pero que se encuentran escritos de distintas formas en diferentes artículos.

Por ejemplo, si Komssi *et al.* dicen que "profesionales de la industria experimentan las inspecciones como inefectivas y difíciles" [28] y Mishra & Mishra que "Lo métodos formales (de inspección) son rigurosos en su implementación" [24], ambos son codificados bajo el factor "Características propias del proceso o percibidas como parte del mismo". Dentro de esta misma codificación existen factores más subjetivos que valen la pena investigar ya que indican que las inspecciones son vistas como actividades aburridas, que no generan valor y que provocan pérdidas de tiempo [29] [30] [28].

También existen afirmaciones que indican como principal factor la falta de capacitación o el desconocimiento de las técnicas de inspección [21] [12] mientras que Shull *et al.* mencionan la curva de aprendizaje como principal dificultad [30]. Por último existen factores de costo a considerar.

La Tabla III muestra la frecuencia con la cual aparecen cada uno de los 13 factores codificados correspondientes a los artículos procesados.

TABLA III. CODIFICACIÓN DE FACTORES CAUSANTES DE LA BAJA ADOPCIÓN (FRECUENCIA).

Factor	#
Características propias del proceso o percibidas como parte del mismo	19
Falta de conocimiento y entrenamiento de los inspectores	9
Inspecciones son consideradas costosas (aumento del costo upfront)	5
Falta de adaptación y mejoras del proceso según el contexto donde se aplique	4
Falta de herramientas de gestión, soporte, análisis del proceso y sus resultados	4
Falta de tiempo asignado a las inspecciones durante la planificación	4
Falta de monitoreo y registro de la ejecución del proceso y de resultados	3
Malas experiencias previas y experiencias fallidas sin reportar	3
Falta o consumo intensivo de recursos	2
Resistencia al cambio	2
Desarrollo distribuido	1
Rol de facilitador	1
Otros	7

Por último y para responder a la pregunta RQ.4 sobre las soluciones planteadas en la literatura estudiada se generó una lista de ellas en la Tabla IV. La misma se elaboró con lo hallado durante la misma ronda de lectura de artículos utilizada para identificar los factores causantes de la baja adopción.

TABLA IV. SOLUCIONES PLANTEADAS RESULTANTES DEL MAPEO SISTEMÁTICO DE LA LITERATURA.

Soluciones	Ref.
Nuevos procesos de inspección fáciles de implementar, que no requieren casi documentación y adaptables.	[24] [25]
Mejorar la habilidad de modificar código a través de inspecciones.	[29]
Utilizar la información generada durante una inspección para entender mejor las salidas del proceso.	[30]
Otros usos de inspecciones como inspeccionar casos de test (identificar "test smells").	[31]
El uso generalizado de inspecciones de software es más un tema de liderazgo que técnico.	[28]
Soporte a la innovación por parte de la gerencia.	[28] [32]
Definir un *champion* dedicado.	[28] [32]
Material de apoyo para los inspectores.	[28] [32]
Información mostrando resultados de inspecciones en contextos similares.	[28] [32]
Adaptar el proceso al contexto sin quitar las partes más importantes.	[28] [13]
Integrar el proceso de inspección al de desarrollo dejando de ser opcional su uso.	[13]
Uso de técnicas de lectura sistemáticas para disminuir dependencia respecto a la experiencia del inspector.	[13]

V. DISCUSIÓN

Del resultado del mapeo sistemático de la literatura se realizan observaciones que coinciden con algunas de las realizadas por otros autores. Por ejemplo, Laitenberger & DeBaud [18] mencionan que han habido muchas contribuciones en forma de nuevas metodologías lo que concuerda con los resultados del estudio dado que una de las

subclases de mayor actividad es la de Procesos. También se mencionan contribuciones en forma de mejoras incrementales (prometiendo aumentar los beneficios de las inspecciones) lo que concuerda con que la subclase Factores de Efectividad sea una de las más activas.

El volumen de trabajo en la subclase Procesos es evidencia del problema planteado por Shull *et al.* acerca de que en forma frecuente nuevas tecnologías son propuestas pero nunca llegan a salir del ambiente académico y las que si lo hacen generan muy poca información empírica [33]. De esta forma la industria tiene una gran variedad de tecnologías para elegir pero muy pocas guías y evidencias que contribuyan a la toma de decisiones acertadas. Esto conduce a afirmar que uno de los problemas que ha llevado a que las inspecciones no sean una práctica habitual en la industria es la confusión entre los distintos métodos de revisión [8].

También se encontró un gran volumen de trabajos correspondientes a la clase Vista Técnica de la taxonomía lo que podría indicar que quizás los problemas de adopción se han estado intentando atacar creando o modificando técnicas de inspección en lugar de tratar de entender que pasaba con las ya existentes. Esta sospecha podría estar sustentada por la falta de datos empíricos que no permiten validar propuestas teóricas mencionadas por Shull *et al.* y Kollanus *et al.* [1] [33].

El procesamiento de los resultados del mapeo sistemático de la literatura dejó una lista de evidencias sobre el problema de la adopción de inspecciones de software. Este hecho confirma que el tema de la adopción es de interés para la comunidad y que se debe trabajar más en él. Otro resultado de este análisis es que las empresas que realizan inspecciones no lo hacen de forma sistemática [13] [25], lo que agrega al problema de la adopción otro factor que es el de la calidad con que se realiza dicha adopción.

Con respecto a los factores causantes de la baja adopción, se puede ver en los resultados de la Tabla III que el que aparece con mayor frecuencia tiene que ver con características propias del proceso o percibidas como parte del mismo. Dentro de este grupo, hay coincidencias al expresar que el proceso de inspección es muy rígido y riguroso [34] [35] [24] y que su complejidad evita que se adopte en pequeñas y medianas empresas, donde se hace difícil su implementación con pocos recursos [24]. A esto se le suma el hecho de que es un proceso no tecnológico [12] que depende mucho de la experiencia de los inspectores [36]. Además, el esfuerzo realizado es de difícil conexión con la calidad final del producto [25], lo que ayuda a que se considere como un actividad poco disfrutable [12]. También, como opiniones más subjetivas, se trata a la inspección como un proceso aburrido [37] [28], laborioso [38] [13], pesado [30] y poco creativo [28]. Por otra parte se encuentran aquellos que cuestionan al proceso en sí mismo, viéndolo como una pérdida de tiempo [28], una actividad desconectada del día a día de los desarrolladores [30], que no resuelven problemas reales del equipo [30] llegando a cuestionar la efectividad del proceso [13] [28].

El siguiente factor es el relacionado con aspectos asociados con la formación de inspectores y capacitación en general. Como ejemplos relacionados con la formación, se menciona la falta de entrenamiento [13], la dificultad de realizar inspecciones correctamente [12] y la curva de aprendizaje que implica que a los desarrolladores les lleve algún tiempo entender cómo encontrar defectos efectivamente [30]. Respecto a la capacitación en general, la falta de conocimiento sobre el uso de inspecciones [21] y la confusión entre los distintos procesos de revisión [8] parecen ser los más frecuentes. De todas formas esta falta de conocimiento no solo se da a nivel técnico (de quiénes lo aplican) sino que también se da a nivel gerencial. A este nivel se manifiesta una falta de entendimiento del proceso de inspección, cuáles son sus beneficios y que responsabilidades tiene la gerencia en la adopción de este tipo de procesos [21].

Respecto al tercer factor de la lista, este trata sobre el costo adicional asociado al proceso de inspección. Si bien algunos autores solo mencionan la adición de costo al proceso de desarrollo [28] [12] [13] otros describen el costo como una gran inversión inicial [8] [25].

El resto de los factores que aparecen con menor frecuencia no dejan de ser importantes y es evidente que algunos se encuentran relacionados. Por ejemplo, sería interesante determinar si existe algún tipo de relación entre los factores de percepción del proceso de inspecciones y los factores correspondientes a malas experiencias o experiencias fallidas sin reportar. Por otra parte, también sería interesante estudiar cómo afectaría la adaptación de un proceso de inspección y el monitoreo de resultados en la percepción acerca del proceso de desarrolladores y niveles gerenciales.

Otro resultado obtenido del mapeo sistemático de la literatura fue una serie de sugerencias realizadas por parte de distintos autores para solucionar el problema de la adopción. Dichas soluciones, así como también cualquier otra que surja del análisis de los factores hallados, deberán ser estudiadas en mayor profundidad. La principal dificultad de la incorporación de algunas de estas sugerencias es que se desconoce que implicancias pueden llegar a tener en procesos de desarrollo de software ya establecidos. Por ejemplo, no se sabe el impacto que puede tener la definición de un rol encargado de velar por la forma en que se realizan las inspecciones (*champion*) en la actividad cotidiana, dado que se agrega un *overhead* al trabajo de una persona. No sería una buena solución la incorporación de cambios (a causa de los factores encontrados) al proceso tradicional de inspección sin un profundo análisis del impacto de los mismos en la efectividad del proceso. Por ejemplo, quitarle rigurosidad o rigidez al proceso quizás haga que los inspectores se sientan más a gusto con el mismo pero por otra parte haga que el número de defectos encontrados disminuya [39].

VI. CONCLUSIONES

El objetivo de esta investigación fue obtener un mapa de factores que dificultan la adopción de inspecciones de software. Esto fue posible gracias al uso de la herramienta mapeo sistemático de la literatura que permite explorar un tema amplio como el de inspecciones de software y responder varias preguntas con diferentes niveles de complejidad.

La pregunta respecto a los temas o áreas que son de interés para la comunidad, fue respondida utilizando el resultado de la búsqueda y el análisis de títulos y *abstracts*. La clasificación de

trabajos permitió conocer que existen subclases como ser las de Factores de Efectividad y Procesos que han sido muy activas concentrando el 42% del total de artículos estudiados (correspondientes a los últimos años). Esto significa que existen muchas propuestas nuevas de mejoras que deben ser probadas experimentalmente para medir las mismas.

Por otra parte se encontró evidencia en trabajos previos que confirman que la comunidad científica ha planteado inquietudes similares a las que motivan este artículo. De todas maneras se encontró un enfoque distinto en alguno de estos trabajos ya que tratan, no solo la adopción, sino también la calidad con la que se lleva a cabo dicho proceso.

Respecto a los factores causantes de la baja adopción, se elaboró una lista de los mismos. Por motivos de espacio se decidió generar una lista de factores más amplios (una codificación) de forma de concentrar la información surgida del análisis de los trabajos resultantes del estudio. El resultado indica que el mayor número de factores se encuentran relacionados con la percepción (subjetiva) que tienen los desarrolladores respecto al proceso, la falta de capacitación y algunas características propias del proceso como la rigidez, la complejidad y la dificultad de conectar el esfuerzo realizado con la calidad del producto final.

Los próximos pasos a seguir deberían ir por dos caminos. Por un lado sería interesante investigar si existe una relación entre el volumen de trabajo (superior al resto) dentro de la clase Vista Técnica de la taxonomía utilizada para clasificar los hallazgos y las causas de adopción mencionadas. Por otra parte se espera poder tomar algunos de los factores sugeridos como los causantes del problema planteado y realizar algún tipo de estudio empírico. El objetivo sería comprobar que la baja adopción de inspecciones de software, en ciertas condiciones y bajo determinados parámetros, puede ser causada por ese subconjunto de factores seleccionados.

VII. Referencias

[1] S. Kollanus and J. Koskinen, "Survey of Software Inspection Research," *Open Software Engineering Journal*, vol. 3, pp. 15–34, 2009.

[2] G. Tassey, "The economic impacts of inadequate infrastructure for software testing," *National Institute of Standards and Technology*, 2002.

[3] W. S. Humphrey, *Managing the software process*. Addison-Wesley Longman Publishing Co., Inc., 1989, p. 494.

[4] M. Fagan, "Design and code inspections to reduce errors in program development," *IBM Systems Journal*, vol. 15, no. 3, pp. 182-211, 1976.

[5] T. Gilb and D. Graham, "Software inspection." Addison-Wesley, p. 471, 1993.

[6] I. Tervonen, L. Harjumaa, and J. Iisakka, "The virtual logging meeting : a web-based solution to resource problems in software inspection," in *Proceedings of the Sixth European Conference on Software Quality*, 1999, pp. 1-10.

[7] IEEE, "IEEE Standard 1028-2008, IEEE Standard for Software Reviews and Audits," 2008.

[8] B. Brykczynski, R. Meeson, and D. Wheeler, "Software Inspection: Eliminating Software Defects," in *In Proceedings of the Sixth Annual Software Technology Conference*, 1994, p. 12.

[9] C. Jones, "Measuring defect potentials and defect removal efficiency," *CrossTalk The Journal of Defense Software Engineering*, vol. 21, no. 6. pp. 11-13, 2008.

[10] E. Doolan, "Experience with Fagan's inspection method," *Software: Practice and Experience*, vol. 22, no. 2, pp. 173–182, Feb. 1992.

[11] S. Kollanus, "ICMM—a maturity model for software inspections," *Journal of Software Maintenance*, vol. 23, no. 5, pp. 327-341, Apr. 2011.

[12] R. Ronald, *High Quality Low Cost Software Inspections*. Paradoxicon Publishing, 2001, p. 479.

[13] M. Ciolkowski, O. Laitenberger, and S. Biffl, "Software reviews: The state of the practice," *IEEE software*, vol. 20, no. 6, pp. 46–51, 2003.

[14] B. Kitchenham, D. Budgen, and O. Pearl Brereton, "Using mapping studies as the basis for further research – A participant-observer case study," *Information and Software Technology*, vol. 53, no. 6, pp. 638-651, Jun. 2011.

[15] K. Petersen, R. Feldt, and S. Mujtaba, "Systematic mapping studies in software engineering," in *Proceedings of the 12th International Conference on Evaluation and Assessment in Software Engineering*, 2008, pp. 1-10.

[16] A. Aurum, H. Petersson, and C. Wohlin, "State-of-the-art: software inspections after 25 years," *Software Testing, Verification and Reliability*, vol. 12, no. 3, pp. 133-154, Sep. 2002.

[17] O. Laitenberger, "A survey of software inspection technologies," in *Handbook on Software Engineering and Knowledge Engineering*, vol. 2, Citeseer, 2002, pp. 517-556.

[18] O. Laitenberger and J.-M. DeBaud, "An Encompassing Life-Cycle Centric Survey of Software Inspection," *Journal of Systems and Software*, vol. 50, no. 1, pp. 5-31, 2000.

[19] K. Wiegers, "The more things change," *Better Software*, vol. 8, no. 1, pp. 30-34, 2006.

[20] J. Iisakka, I. Tervonen, and L. Harjumaa, "Experiences of painless improvements in software inspection," *Project Control for Software Quality, ESCOM-SCOPE*, vol. 99, pp. 321–327, 1999.

[21] R. Stewart and L. Priven, "How to Avoid Software Inspection Failure and Achieve Ongoing Benefits," *CrossTalk: The Journal for Defense Software Engineering*, pp. 23-27, 2008.

[22] E. Weller, "Calculating the Economics of Inspections," *StickyMinds*, 2002. [Online]. Available: http://www.stickyminds.com/s.asp?F=S3161_ART_2. [Accessed: 07-Jun-2011].

[23] C. J. van Rijsbergen, "Information Retrieval," in *Information Retrieval*, Butterworth, 1979, p. 208.

[24] D. Mishra and A. Mishra, "Simplified software inspection process in compliance with international standards," *Computer Standards & Interfaces*, vol. 31, no. 4, pp. 763-771, Jun. 2009.

[25] C. Denger and F. Shull, "A Practical Approach for Quality-Driven Inspections," *IEEE Software*, vol. 24, no. 2, pp. 79-86, Mar. 2007.

[26] J. Remillard, "Source code review systems," *IEEE Software*, vol. 22, no. 1, pp. 74-77, 2005.

[27] S. Kollanus, "The Role of Different Approaches in Inspection Process," *Software Quality Journal*, vol. 35, no. 2, pp. 231-245, Jan. 2009.

[28] M. Komssi, M. Kauppinen, M. Pyhajarvi, J. Talvio, and T. Mannisto, "Persuading Software Development Teams to Document Inspections: Success Factors and Challenges in Practice," *18th IEEE International Requirements Engineering Conference*, pp. 283-288, Sep. 2010.

[29] D. A. McMeekin, B. R. V. Konsky, E. Chang, and D. J. A. Cooper, "Checklist Based Reading's Influence on a Developer' s Understanding," in *19th Australian Software Engineering Conference*, 2008.

[30] F. Shull, R. L. Feldmann, C. Seaman, M. Regardie, and S. Godfrey, "Fully employing software inspections data," *Innovations in Systems and Software Engineering*, May 2010.

[31] F. Lanubile and T. Mallardo, "Inspecting Automated Test Code : A Preliminary Study," *Agile Processes in Software Engineering and Extreme Programming*, vol. 4536, pp. 115-122, 2007.

[32] F. Shull and C. Seaman, "Inspecting the History of Inspections: An Example of Evidence-Based Technology Diffusion," *IEEE Software*, vol. 25, no. 1, pp. 88-90, 2008.

[33] F. Shull, J. Carver, G. H. Travassos, J. C. Maldonado, R. Conradi, and V. R. Basili, "Replicated studies: building a body of knowledge about software reading techniques," *Lecture notes on empirical software engineering*, pp. 39–84, 2003.

[34] S. Nazir, N. Fatima, and S. Malik, “Effective Hybrid Review Process (EHRP),” *2008 International Conference on Computer Science and Software Engineering*, pp. 763-771, 2008.

[35] D. Mishra and A. Mishra, “Efficient software review process for small and medium enterprises,” *IET Software*, vol. 1, no. 4, pp. 132 -142, 2007.

[36] B. Xu, “Cost Efficient Software Review in an E-Business Software Development Project,” *2010 International Conference on E-Business and E-Government*, pp. 2680-2683, May 2010.

[37] V. Suma and T. R. G. Nair, “Enhanced Approaches in Defect Detection and Prevention Strategies in Small and Medium Scale Industries,” *2008 The Third International Conference on Software Engineering Advances*, pp. 389-393, Oct. 2008.

[38] Ö. Albayrak and D. Davenport, “Impact of Maintainability Defects on Code Inspections,” *Empirical Software Engineering and Measurement*, pp. 9-12, 2010.

[39] L. Harjumaa, I. Tervonen, and P. Vuorio, “Improving software inspection process with patterns,” *Fourth International Conference onQuality Software, 2004. QSIC 2004. Proceedings.*, pp. 118-125, 2004.