UNIVERSIDAD DE MURCIA

ESCUELA INTERNACIONAL DE DOCTORADO

TESIS DOCTORAL

Sustainable DevOps Adoption Model based on Human Factors

Modelo de Adopción Sostenible de DevOps basado en Factores Humanos.

D. Juan José Pérez Sánchez

2024

UNIVERSIDAD DE MURCIA

ESCUELA INTERNACIONAL DE DOCTORADO

TESIS DOCTORAL

Sustainable DevOps Adoption Model based on Human Factors

Modelo de Adopción Sostenible de DevOps basado en Factores Humanos.

Autor: D. Juan José Pérez Sánchez

Director/es: D. Joaquín Nicolás Ros y Juan Manuel Carrillo de Gea

UNIVERSIDAD DE
MURCIA

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD
DE LA TESIS PRESENTADA EN MODALIDAD DE COMPENDIO O ARTÍCULOS PARA OBTENER EL TÍTULO DE DOCTOR

Aprobado por la Comisión General de Doctorado el 19-10-2022

D./Dña. Juan José Pérez Sánchez

doctorando del Programa de Doctorado en

Informática

de la Escuela Internacional de Doctorado de la Universidad Murcia, como autor/a de la tesis presentada para la obtención del título de Doctor y titulada:

Sustainable DevOps Adoption Model Based on Human Factors / Modelo de Adopción Sostenible de DevOps Basado en Factores Humanos

y dirigida por,

D./Dña. Joaquín Nicolás Ros

D./Dña. Juan Manuel Carrillo de Gea

D./Dña.

DECLARO QUE:

La tesis es una obra original que no infringe los derechos de propiedad intelectual ni los derechos de propiedad industrial u otros, de acuerdo con el ordenamiento jurídico vigente, en particular, la Ley de Propiedad Intelectual (R.D. legislativo 1/1996, de 12 de abril, por el que se aprueba el texto refundido de la Ley de Propiedad Intelectual, modificado por la Ley 2/2019, de 1 de marzo, regularizando, aclarando y armonizando las disposiciones legales vigentes sobre la materia), en particular, las disposiciones referidas al derecho de cita, cuando se han utilizado sus resultados o publicaciones.

Además, al haber sido autorizada como compendio de publicaciones o, tal y como prevé el artículo 29.8 del reglamento, cuenta con:

- *La aceptación por escrito de los coautores de las publicaciones de que el doctorando las presente como parte de la tesis.*
- *En su caso, la renuncia por escrito de los coautores no doctores de dichos trabajos a presentarlos como parte de otras tesis doctorales en la Universidad de Murcia o en cualquier otra universidad.*

Del mismo modo, asumo ante la Universidad cualquier responsabilidad que pudiera derivarse de la autoría o falta de originalidad del contenido de la tesis presentada, en caso de plagio, de conformidad con el ordenamiento jurídico vigente.

En Murcia, a 15 de agosto de 2024

Fdo.: Juan José Pérez Sánchez

Esta DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD debe ser insertada en la primera página de la tesis presentada para la obtención del título de Doctor.

Agradecimientos

Esta tesis es la culminación no solo de mis esfuerzos, sino de los esfuerzos de las personas que me han acompañado en este viaje, sin los cuales no habría sido posible.

Agradezco, en primer lugar, a mis directores de tesis, Joaquín Nicolás Ros y Juan Manuel Carrillo de Gea. Gracias por vuestra paciencia, vuestro apoyo, y vuestro conocimiento. Gracias por haberme guiado y por haberme ayudado a crecer como investigador. Me gustaría agradecer también a Ambrosio Toval, para mí, un tercer director de tesis durante los primeros años de mi doctorado.

A mis compañeros de laboratorio, Manal, José Antonio, Imane, Raimel, y a todos los que he tenido el privilegio de conocer. Gracias a vosotros he sentido que no estaba solo en esta etapa de mi vida. Gracias por las conversaciones espontáneas y por los cafés, por demostrar que se puede ser estudiante de doctorado y mejorar los días de los demás a la vez.

A mis amigos, que por fortuna son muchos, y siempre me han apoyado. A todos los que de corazón me habéis preguntado por cómo me iba la tesis, y a los que me habéis dado ánimos en los momentos difíciles. Gracias por estar ahí.

A mis padres. Me habeis enseñado a ser todo lo que soy. Mis méritos son tan vuestros como míos. Gracias a mi hermana, por ser siempre quien es. Gracias a mis tíos, primos y abuelos. Gracias a los que ya no están, pero que siempre estarán dentro de mi.

A Sara. Me ayudas mucho más de lo que crees. Gracias a ti los buenos momentos son mejores y los momentos difíciles parecen pequeños. Gracias por ser mi compañera de viaje.

Contents

List of Figures

List of Tables

Resumen

1 — Introducción

La necesidad de reducir el impacto humano sobre el planeta es cada vez mayor. La temperatura media de la superficie terrestre ha aumentado en 1°C desde el periodo de 1850-1900 hasta el periodo de 2011-2020. La razón principal para este aumento de temperatura se debe a la emisión de gases de efecto invernadero, principalmente dióxido de carbono y metano. Prueba de la importancia de este problema es la creación de múltiples acuerdos internacionales para reducir las emisiones de gases de efecto invernadero, como el Protocolo de Kioto de 1997 o el Acuerdo de París de 2015.

El área de las tecnologías de la información y la comunicación (TIC) no es ajena a este problema. En 2020, se estimaba que las TIC eran responsables de entre el 1,8% y el 2,8% de las emisiones de gases de efecto invernadero a nivel mundial. A pesar del enorme crecimiento en eficiencia que han experimentado las TIC, sus emisiones aumentan a un ritmo mayor que el ritmo de crecimiento global. Es por ello que se le está dando especial importancia a la sostenibilidad de la industria de las TIC. En este contexto, el Manifiesto de Karlskrona define la sostenibilidad de la Ingeniería del Software como un concepto compuesto por cinco dimensiones: económica, social, medioambiental, técnica e individual.

Esta tesis doctoral hace énfasis en la sostenibilidad individual y social, esto es, en el mantenimiento del capital humano, através de aspectos como la salud, habilidades o el conocimiento, y la preservación de las comunidades sociales. El desarrollo de software es una actividad inherentemente social. El rol de las personas en la Ingeniería del Software ha ganado peso para la investigación científica desde el siglo XX hasta nuestros días. Se ha argumentado repetidamente que el éxito de un proyecto software depende más de las personas que de la tecnología utilizada. De hecho, se ha identificado una relación predecible entre la motivación y la personalidad de los desarrolladores de software y el rendimiento en el desarrollo de software. Sin embargo, pocos estudios han investigado los qué factores humanos influyen en la adopción de DevOps, que es un enfoque dominante en la industria del software de nuestros días.

DevOps es una metodología de desarrollo de software que busca la integración entre los equipos de desarrollo y operaciones. DevOps ha sido relacionado con la colaboración, automatización, y las prácticas continuas. Sin embargo, a pesar de los esfuerzos de la comunidad científica, no existe un consenso sobre qué es DevOps exactamente. DevOps ha demostrado ser una metodología que permite entregas de software más rápidas, y de mayor calidad, pero los estudios disponibles en la literatura científica no responden suficientemente a la pregunta de cómo se adopta DevOps en la práctica.

Esta tesis tiene como objetivo principal facilitar la adopción de DevOps a través de un modelo sostenible basado en factores humanos.

2 — Metodología

La metodología principal que une el trabajo realizado en la tesis doctoral es la Ciencia del Diseño o Design Science. La Ciencia del Diseño es una metodología de investigación que busca la creación de artefactos con valor práctico. En particular, la tesis sigue el marco metodológico definido por Johannesson y Perjons. El marco define cinco actividades de investigación: explicar el problema, definir los requisitos para resolverlo, diseñar y desarrollar un artefacto, demostrar que el artefacto puede resolver el problema, y evaluar hasta qué punto el artefacto resuelve el problema. Johannesson y Perjons afirman que el investigador debe moverse por estas actividades de forma cíclica, pero también permite que el investigador se mueva entre actividades de forma no lineal, es decir, el investigador puede volver a una actividad anterior si es necesario.

La tesis doctoral, en el espíritu de la Ciencia del Diseño, se desarrolla en torno a la creación y mejora de un artefacto. En primer lugar, después del estudio del estado del arte y de la definición de requisitos, se propone una taxonomía de factores humanos que influyen en la adopción de DevOps, como primer artefacto. Este artefacto se valida y evalúa, y se refina para crear un segundo artefacto, una teoría para explicar el rol de los factores humanos en la adopción de DevOps. La teoría se usa para definir los requisitos de un modelo de adopción de DevOps, y como componente del mismo. Finalmente, se propone el artefacto final de la tesis, un modelo de adopción de DevOps compuesto por un modelo de acción para la medición y mejora de los factores humanos, y una herramienta software que implementa el modelo de acción.

3 — Resultados

La tesis doctoral se presenta en modalidad de compendio de publicaciones y está compuesta de cuatro publicaciones científicas.

3.1. DevOps Certification in IT Industry: Preliminary Findings

Para proponer un modelo de adopción de DevOps, el objetivo final de la tesis, es necesario entender en profundidad el estado del arte en la adopción de DevOps. Sin embargo, los investigadores coinciden en que la investigación en DevOps está quedando atrás en comparación con los avances de la industria, y el conocimiento disponible en la literatura científica puede no ser una buena representación del estado actual de DevOps en la industria. Por ello, decidimos abordar el problema a través de la industria. Las certificaciones son una de las formas que tiene la industria de reconocer habilidades y conocimientos en una determinada área, y el mecanismo está bien establecido en las TIC. En este punto, decidimos investigar qué es DevOps, y qué habilidades supuestamente se requieren para trabajar en un entorno DevOps, según la información proporcionada por las certificaciones de DevOps.

En el primer artículo del compendio *DevOps Certification in IT Industry: Preliminary Findings*, se trata de identificar y extraer esta información. En primer lugar, se consideraron todas las certificaciones que aparecieron en las primeras 10 páginas de una búsqueda en Google con los términos “DevOps certification”. Google fue elegido como motor de búsqueda (en lugar de un motor de búsqueda de bases de datos científicas) para identificar certificaciones presentes en la industria, no necesariamente en la academia. Se encontraron un total de 105 certificaciones, proporcionadas por un total de 45 organizaciones certificadoras. Se diseñó una encuesta para recopilar la información sobre las certificaciones directamente de estas organizaciones. Sin embargo, la encuesta no alcanzó el número de respuestas esperado, y se decidió recopilar la información directamente de las páginas web de las organizaciones certificadoras. Este enfoque se presenta en el siguiente artículo del compendio.

3.2. DevOps Certifications for IT Professionals

El siguiente artículo en el compendio *DevOps Certifications for IT Professionals* se centra en el mismo objetivo que el artículo anterior, que quedó incompleto, pero con un enfoque diferente. En este caso, la información se obtuvo directamente de las certificaciones, sin necesidad de que las organizaciones participaran en la encuesta.

Las certificaciones se identificaron a través de una búsqueda en Google, como en el artículo anterior. Sin embargo, en este caso Google fue seleccionado entre cinco motores de búsqueda generalistas populares (Google, Bing, Yandex, Baidu, y Yahoo), repitiendo la misma búsqueda desde diferentes localizaciones alrededor del mundo. Google superó a todos los motores de búsqueda en la tarea de identificar certificaciones de DevOps. Se identificó un total de 45 certificaciones, sin embargo, solo 22 de ellas incluyeron suficiente información para ser analizadas.

Las certificaciones fueron analizadas en términos de qué competencias de DevOps se certifican. Así mismo, las certificaciones se clasificaron en un total de 12 clústeres.

Los resultados del estudio tienen como objetivo informar y asistir a profesionales que estén interesados en obtener una certificación de DevOps, pero también extender el conocimiento de la investigación de las competencias que se requieren en la industria. Los resultados del estudio nos permitieron percibir una falta de atención en cuanto a las habilidades no técnicas de DevOps. Este hecho motiva la investigación sobre las habilidades no-técnicas (o "soft-skills") de DevOps, que están en gran parte inexploradas tanto en la industria como en la investigación científica.

3.3. A Taxonomy on Human Factors that Affect DevOps Adoption

El tercer artículo del compendio *A Taxonomy on Human Factors that Affect DevOps Adoption* es la base para la propuesta de una teoría sobre los factores humanos en la adopción de DevOps. Previamente al desarrollo del artículo, se realizó una breve revisión de la literatura, que sugería que el estudio de los factores humanos en DevOps estaba en gran parte inexplorado, como también se apoyaba en el artículo anterior del compendio. Dado que no había suficiente información disponible para proponer una teoría sobre los factores humanos en DevOps, se diseñó este artículo. El artículo identifica y clasifica los factores humanos que están relacionados con la adopción de DevOps a través de un estudio de mapeo sistemático. Aunque, como se afirma anteriormente, casi no existía información disponible sobre el tema, el estudio se realizó de manera amplia, adaptando el conocimiento presente en otras metodologías (no necesariamente DevOps), que están intrínsecamente relacionadas con DevOps. Este enfoque de adaptar el conocimiento existente a otra disciplina ha sido ampliamente utilizado y apoyado en la investigación de Tecnologías de la Información.

Un total de 21 estudios fueron seleccionados, y un conjunto de 59 factores humanos fueron extraídos de estos estudios. Los factores humanos fueron descritos y clasificados en función de su efecto en la adopción de DevOps.

La información recopilada en este estudio se utilizó como base para proponer una teoría sobre la adopción de DevOps basada en factores humanos, y a su vez para la extensión de la caracterización de DevOps. A través de los factores humanos que afectan a la adopción de DevOps, podemos entender cómo interactúan los miembros de la organización con la metodología, lo que al mismo tiempo, nos permite entender la metodología en sí misma.

3.4. A Theory on Human Factors in DevOps Adoption

El cuarto y último artículo del compendio, *A Theory on Human Factors in DevOps Adoption*, realiza una propuesta de una teoría sobre los factores humanos en la adopción de DevOps. El artículo también extiende el conocimiento sobre la caracterización de DevOps, de la misma manera que el artículo anterior, y sirve como base para el desarrollo del modelo de adopción de DevOps, el objetivo final de la tesis. El artículo es una extensión del anterior, y se basa en el mismo conjunto de factores humanos que se sintetizaron en el artículo anterior. Sin embargo, la búsqueda se extendió para incluir publicaciones hasta julio de 2024.

La relevancia de los factores humanos en la adopción de DevOps fue evaluada a través de una encuesta dirigida a profesionales. La gran mayoría de los factores humanos propuestos fueron considerados relevantes para la adopción de DevOps, y sólo un factor fue considerado no relevante.

Se propuso una teoría sobre los factores humanos en la adopción de DevOps. La teoría identifica y describe seis constructos primarios: “factor humano”, “actor”, “acción”, “recursos”, “nivel de prioridad” y “rendimiento organizacional”. La teoría también propone una serie de relaciones entre los constructos, y una serie de requisitos para llevar la teoría a la práctica.

3.5. DevOps Human Factor Adoption Model

Esta parte de la tesis no ha sido publicada y no forma parte de las publicaciones del compendio, pero es una parte imprescindible de la tesis doctoral. Junto con los resultados de las publicaciones anteriores, se propone un modelo de adopción de DevOps compuesto por un modelo de acción y una herramienta software que implementa el modelo de acción.

El modelo de acción se basa en el conjunto de factores humanos identificado previamente, que se extendió con el conocimiento extraído de un área de investigación más amplia existente en la literatura. El modelo se diseñó para ser una herramienta práctica que pudiera ser utilizada por las organizaciones para mejorar su proceso de adopción de DevOps. El modelo de acción está compuesto de tres elementos: factores humanos, “preguntas” para realizar mediciones sobre los “factores humanos”, y “acciones” para mejorar el estado de los “factores humanos”.

El modelo de acción propuesto se utilizó como motor teórico de una herramienta software, diseñada para gestionar de forma automática la medición y sugerencia de acciones sobre los factores humanos. La herramienta se denominó “Human DevOps”, y consta de tres componentes principales: una aplicación web front-end, un servidor back-end, y una aplicación Slack. El modelo de acción y la herramienta software, en conjunto, forman el artefacto final de la tesis doctoral.

4 — Conclusiones

Esta tesis doctoral ha investigado los factores humanos en la adopción de DevOps con el objetivo final de proponer un modelo de adopción que asista a las organizaciones en el proceso de adopción de DevOps, considerando factores humanos, que sea positivo para el rendimiento organizacional y para la sostenibilidad individual y social. La tesis ha contribuido al campo de investigación de DevOps a través de los siguientes resultados:

- Se han identificado y analizado los requisitos de la industria de DevOps a través de las principales certificaciones disponibles en el mercado. El análisis de las certificaciones ha mostrado que la industria de DevOps se centra principalmente en las habilidades técnicas, mientras que los aspectos culturales de DevOps apenas se abordan.
- Se ha propuesto una taxonomía de factores humanos de DevOps para identificar y clasificar los factores humanos relacionados con la adopción de DevOps. La taxonomía describe y clasifica los factores humanos en función de sus características y efectos en el proceso de adopción de DevOps.
- Se ha propuesto una teoría de factores humanos de DevOps para explicar el papel de los factores humanos en la adopción de DevOps. La teoría explica cómo los factores humanos están involucrados con otros constructos de desarrollo de software.
- Se ha propuesto un modelo de acción de factores humanos de DevOps para actuar como un marco de medición y recomendación para la adopción de DevOps.
- Se ha desarrollado una herramienta software que implementa el modelo de acción de factores humanos de DevOps para ayudar a las organizaciones a medir y mejorar su proceso de adopción de DevOps.
- DevOps se ha caracterizado a través del estudio de sus aspectos humanos. Los factores humanos, como parte intrínseca de DevOps, explican cómo los profesionales impactan en y son afectados por el proceso de adopción de DevOps, y al mismo tiempo, explican qué es DevOps.
- Los resultados de la tesis pueden mejorar la sostenibilidad individual y social creando un mejor entorno de trabajo, mejorando el bienestar de los profesionales involucrados.

Palabras clave: DevOps, Factores Humanos, Sostenibilidad, Ingeniería del Software

Abstract

Human influence has unequivocally caused radical changes in the climate of many land regions, and partially irreversible damage to all ecosystems on the planet. Consequently, the need to reduce human impact on the planet is increasingly present. The field of Information and Communication Technologies (ICT) is not foreign to this problem. In fact, in the ITC field, emissions have increased at a higher rate than the global growth rate in emissions. In order to be able to study sustainability in Software Engineering, the Karlskrona Manifesto was proposed. The manifesto defines sustainability as a multidimensional concept, composed of five dimensions: individual, social, economic, environmental, and technical. This thesis focuses mainly on the improvement of individual sustainability, which refers to the maintenance of human capital, and social sustainability, which refers to the preservation of social communities.

DevOps is a software engineering paradigm whose objective is to close the gap between Development (Dev) and Operations (Ops) teams. However, there is no consensus in the scientific literature regarding the precise definition of DevOps. High-performance DevOps teams deliver code faster, more frequently, recover more quickly from downtime, and are less likely to fail, but the scientific literature has not yet fully answered the question of how to adopt DevOps in practice. In addition, very few studies address how human factors affect DevOps adoption.

This thesis aims to fill this gap, and improve individual and social sustainability through a sustainable DevOps adoption model based on human factors. The Design Science research methodology was followed, in particular, Johannesson and Perjons framework. An artifact was iteratively designed and developed in order to achieve the research goal. The research problem was defined, and the first artifact requirements were defined, through the study of DevOps certifications in industry, which provided information regarding which DevOps competencies are receiving more attention in industry. The study identified a gap in DevOps non-technical skills, which motivated the next step. The artifact, a DevOps human factor taxonomy, was developed and validated through a survey. The results were used as the base to propose a theory on DevOps human factors, the second artifact of the thesis, which explains the role of human factors in DevOps adoptions. Using the theory as foundation, a DevOps human factor action model was proposed, the third artifact of the thesis, which allows organizations to measure the state of human factors and take actions to improve them. Finally, the last artifact of the thesis, a software implementation of the action model ("Human DevOps"), was developed.

The results of the thesis improve the current understanding of DevOps by characterizing it through its human aspects. The DevOps adoption model proposed, a combination of the artifacts of the thesis, can assist organizations through the uncertain adoption process. In addition, the results of the thesis can improve individual and social sustainability by creating a better work environment, improving the well-being of the professionals involved.

Keywords: DevOps, Human Factors, Sustainability, Software Engineering

Chapter 1

Introduction

1.1 — Sustainability in Software Engineering

The Earth average surface temperature has increased by 1° C from the period 1850-1900 to 2011-2020. This global warming is caused by human action mainly due to the emission of greenhouse gases: carbon dioxide and methane mostly. In fact, the emission of these gases by humanity has increased globally from 38 $GtCO_2eq$ (the equivalent in greenhouse gases to 38 giga-tonnes of carbon dioxide, standard measure of global warming) in 1990 to 59 $GtCO_2eq$ in 2019. Human influence has unequivocally caused radical changes in the climate of many land regions, and partially irreversible damage to all ecosystems on the planet [3].

It is no wonder that the need to reduce human impact on the planet is increasingly present. The creation of the Intergovernmental Panel on Climate Change (IPCC) by the United Nations in 1988, the signing of the United Nations Framework Convention on Climate Change (UNFCCC) in 1992, the adoption of the Kyoto Protocol in 1997, the United Nations Climate Change Conference of 2009, the establishment of the Green Climate Fund in 2010, the Paris Agreement of 2015, or the European Green Deal of 2019 are examples of this.

The field of Information and Communication Technologies (ICT) is not immune to this problem. In fact, in 2020 it was estimated that ICT was responsible for between 1.8% and 2.8% of global greenhouse gas emissions. Despite the enormous efficiency improvement that the ICT industry has experienced in recent decades, emissions have increased at a higher rate than the global growth rate in emissions [4].

Thus, it is natural that research in software engineering has begun to pay attention to the negative impact that the ICT industry has on the world. In this context, as in many other disciplines, the concept of sustainability becomes particularly relevant. Sustainability in Software Engineering is often defined as "the ability to endure" [5]. This consciously simple definition allows to cover a wide range of aspects that may interfere with the ability of the software industry to endure, not only in terms of environmental impact.

In 2015, the Karlskrona Manifesto was created [5]. The Karlskrona Manifesto was born out of the need to create a common framework for the study of sustainability in Software Engineering. The manifesto defines sustainability as a multidimensional concept, composed of five dimensions: individual, social, economic, environmental, and technical. These dimensions are interrelated and influence each other.

1.2 — Human factors in Software Engineering

This thesis focuses mainly on the study of individual sustainability, and partially on the study of social sustainability in Software Engineering. According to the Karlskrona Manifesto, individual sustainability refers to the maintenance of human capital (e.g. health, education, skills, knowledge, leadership, and access to services), and social sustainability refers to the preservation of social communities in their solidarity and services.

Software development is an inherently human activity [6]. Although Software Engineering is a field eternally surrounded by technology, people remain the most important component of the equation. Already in 1965, Edsger W. Dijkstra [7] spoke of non-technical aspects of programming, in particular, the need for software to be written elegantly so that it could be understood by other programmers. In 1971, Gerald M. Weinberg [8] published a book, now considered a classic, entitled "The Psychology of Computer Programming". In it, Weinberg extensively discusses non-technical, behavioral, psychological, and sociological aspects that affect software development.

The role of people in Software Engineering continues to gain weight at the end of the 20th century. In 1977, the term "peopleware" is used for the first time. This term identifies people as the third pillar of software development, along with hardware and software. The term becomes popular in 1987 with the publication of the book "Peopleware: Productive Projects and Teams" by Tom DeMarco and Timothy Lister [9]. In it, DeMarco and Lister argue that the success of a software project depends more on people than on the technology used.

In 1999, Avison et al. [10] pointed out that the absence of the human component in software development methodologies could be the main cause of dissatisfaction with them. On the contrary, they believed that people are the ones who actually make organizations complex, and that any analysis of a real organization must take these aspects into account. In fact, later studies have identified a predictable relationship between motivation, personality traits of software developers, and software development performance [11]. Supporting this idea, in 2010, Jurgen Appelo published the book "Management 3.0: Leading Agile Developers, Developing Agile Leaders" [12], in which he recognizes that the management of software development organizations relies mainly on relationships and people.

Continuing with the growing interest in human factors, between 2005 and 2018, 66 empirical studies were published that evaluated the emotions of software developers [13]. Human factors are, therefore, a fundamental piece of software development and Software Engineering in general. Employers post job advertisements that include a list of requirements that are both technical and non-technical. However, while technical skills can be assessed at the time of hiring, with relative accuracy through the curriculum vitae or technical interviews, the same does not happen with non-technical skills (also known as “soft-skills”) [14].

In the scientific literature, the term “human factor” is often used generically to refer to the non-technical aspects of Software Engineering. However, to the best of our knowledge, there is no precise definition of what is considered a human factor in Software Engineering. In this thesis, human factors are defined as: “the set of psychological, behavioral, and sociological aspects that are related to the software development process”.

Human factors have been studied in different areas of Software Engineering. For example, in agile methodologies [15, 16], or in Lean methodologies [17, 18]. In addition, conferences have emerged that specifically address human factors in Software Engineering, such as the “International Conference on Cooperative and Human Aspects of Software Engineering” (CHASE), or the “International Conference on Applied Human Factors and Ergonomics” (AHFE).

1.3 — DevOps

DevOps is a software engineering paradigm whose objective is to close the gap between Development (Dev) and Operations (Ops) teams [19]. Although the first time DevOps is defined in scientific literature is with this relatively broad definition, more recent studies have related DevOps to collaboration, automation, and continuous practices [20, 19, 21]. In fact, despite the efforts made to identify exactly what DevOps is, there are many definitions of it [22, 23]. Therefore, there is no consensus in the scientific literature regarding the definition of DevOps. In fact, some recent studies have reported a lack of agreement between the definitions used in the literature [20, 24, 25].

Nowadays, software development companies face an important challenge. On the one hand, they must quickly adapt to the constantly changing market conditions while delivering quality software. On the other hand, the systems that make these deliveries possible are more complex than ever and must maintain a high degree of maintainability and availability [26].

DevOps is first mentioned in scientific literature in 2009, but the main problem it tries to solve had already been identified previously [21, 27, 26]. This problem is the fact that the separation between development and operations teams causes delays in the time required to deliver software products. DevOps focuses on this issue, allowing faster deliveries and higher quality products [28], and consequently, making companies more competitive in an ever-changing industry. High-performance DevOps teams deliver code faster, more frequently, recover more quickly from downtime, and are less likely to fail [29, 30]. In fact, DevOps does not emerge as an isolated initiative from the rest of practices, but emerges from continuous practices, as an evolution of agile development, in a context of Lean principles [31].

DevOps, as a concept, is usually accompanied by a series of other concepts, relevant to explain the role of the methodology in the context of a software development organization. One of the main concepts is the team. In DevOps, as in other software development methodologies, the team is the most common organizational unit. The professionals responsible for software production are distributed in teams based on different criteria. López-Fernandez et al. [32] identify four organizational DevOps patterns based on teams:

- Separated development and operations teams. It is considered the organizational pattern with a lower maturity in relation to DevOps practices, while it is still the traditional organizational pattern for software development. It is considered a DevOps pattern because it requires close collaboration between different teams.

- "Dev-Ops" team composed of developers and operators. Although it is a single team, the members of the team may belong to different departments and may not share the same objectives.

- Development team supported by DevOps experts. In this case, the teams are composed exclusively of developers. However, the support of DevOps experts from the organization allows the assumption of all the responsibilities of the software product in development, including the operations part. Usually the team works with close and frequent collaboration and shares the same objective.

- DevOps team composed of multidisciplinary members. It is considered the most mature organizational pattern. Team members share the same objective and assume all or most of the responsibilities of the software product.

- Horizontal DevOps team. López-Fernández et al. do not identify it as a pattern but as a type of team tangential to the previous four patterns. It is a special team that provides support to other teams. In some cases, it is a team that offers the services of a support platform, and in others, a team of experts that offer methodological, technical, or practical support to other teams. This type of team is not an active part of the development of a software product.

Another organizational concept widely used by software development organizations is the department. Leite et al. [33] identify 4 organizational patterns based on departments, in which clear parallels can be seen with the organizational patterns based on teams identified by López-Fernández et al. [32]:

- Isolated departments. It corresponds to a traditional organizational pattern in which there are separate departments dedicated to development and operations. In these cases, collaboration between departments is usually limited.
- Collaborative departments. It is similar to the pattern of isolated departments, although in this case the development departments actively collaborate in operations.
- Individual departments. This organizational form is common in small companies without large scalability requirements. Each department is responsible for both software development and the maintenance of the necessary infrastructure, so there are no operations departments.
- Departments mediated by APIs. Developers are responsible for all operations of the product, while operators are responsible for the infrastructure. Communication is occasional.

Additionally, Diaz et al. [34] identify the following constructs related to DevOps (excerpt from the article):

- **Management**: Represents the organization's management capabilities to ensure that software development and maintenance are carried out in an organized, systematic, and quantified manner.
- **Culture**: Represents the common set of principles, values, and practices that impact how people in an organization relate to each other, and make decisions and actions during the software product lifecycle.
- **Automation**: Represents the set of activities to automate the processes that are part of the software product lifecycle, as well as the management of the infrastructure where the software is deployed.
- **Platform**: Represents the technology that enables the automated management of the infrastructure and the software lifecycle.
- **Silo**: It can be organizational or cultural. Represents a part of an organization that does not communicate or collaborate with other parts of the organization.
- **Collaboration**: The range of collaboration from absolute absence to daily collaboration.
- **Communication**: The range of communication from poor or infrequent to frequent communication.

In practice, organizations that adopt DevOps are mainly motivated by what is considered an excessive time until the release of a new version. Other reasons for the adoption of DevOps that occur less frequently are (among others, from most to least frequent): 1) The natural evolution due to continuous improvement methodologies, adaptation to market demands, or the need for transformation due to architectural changes or technological obsolescence. 2) Lack of standardization or automation, as DevOps is related to process automation. 3) Organizational or cultural silos, where the different teams do not share the same goals, knowledge, or mindset, as opposed to DevOps, which is based on collaboration [35].

1.4 — DevOps adoption

DevOps is a development methodology that can improve an organization's ability to produce software quickly and effectively. This is why many organizations in the industry have begun to adopt DevOps, reporting benefits qualitatively [23]. However, the scientific literature has not yet fully answered the question: *how is DevOps adopted?* [36]. The adoption of DevOps often involves a technical change, but in all cases it involves a cultural change at the organizational level. Depending on how DevOps is implemented, organizations will need to update their processes, teams, roles, and management methods to adapt to the new methodology.

DevOps is a methodology that arises in a specific context and moment of Software Engineering. This is why, although we define DevOps as an isolated concept, in reality it arises closely related to other methodologies and practices, pre-existing or born later. For example, the adoption of DevOps is closely related to the adoption of agile methodologies and continuous practices [31].

In the literature, numerous articles have been published that study the implementations of DevOps, dividing them into different levels of "maturity", describing the characteristics of these levels, and the transitions between them [37, 38, 39, 40, 41, 42]. Scientific literature has also referred to these maturity models as "competence models" or "readiness models". Although these models study the adoption of DevOps directly, increasing the level of DevOps maturity, as an action in itself, they do not improve organizational performance. In a recent case study [43], it is shown that organizational performance only improves when the improvement in the level of maturity is combined with other interventions and strategies. In addition, maturity models as a concept are based on a series of false assumptions, such as that two different organizations will reach a similar state if they reach the same level of maturity, that the increase in maturity level will have a positive impact on productivity without the need to measure the results, or that the requirements of an organization to reach maturity will not change over time [29].

In a recent study, Amaro et al. [44] study the adoption of DevOps in terms of what capabilities an organization must have to adopt DevOps. Among the most frequently identified capabilities are (from most to least frequent): 1) collaboration and communication between teams, 2) continuous integration, 3) continuous delivery and deployment automation, and 4) proactive monitoring, observability, and automatic scalability. In the study, Amaro et al. group the capabilities into 4 categories. Among them, the category to which the most frequently identified capability belongs is that of “Cultural capabilities”. This supports the importance of the human component in the adoption of DevOps. The adoption of DevOps is, therefore, a process that involves people and the organization from different dimensions, among which the “human dimension” is one of the fundamental pillars.

1.5 — Human factors in DevOps

Human factors are always present in software development [14]. However, very few articles refer to human factors in a DevOps context [45]. In fact, in a recent literature review [46], whose objective is the study of human factors in Software Engineering, not a single study was found that referred to human factors in DevOps. On the other hand, the cultural aspects of DevOps have been identified as the main difficulty in successfully adopting the methodology, compared to the technical aspects [47]. In addition, the adoption process can fail at any of the hierarchical levels of the organization if human factors are not managed appropriately [48]. The adoption of DevOps mainly involves a cultural change that is not shared with other software development methodologies, and that is why the set of human factors involved is different from other methodologies [45]. To the best of our knowledge, no article has studied in depth the relationship between human factors and the adoption of DevOps.

1.6 — Thesis objectives

The doctoral thesis aims to evaluate the following initial hypothesis:

Hypothesis
Proper management of human factors can improve the success and sustainability of DevOps adoption.

In order to evaluate the hypothesis, the following main objective is proposed:

Main objective
The proposal of a model that facilitates the adoption of DevOps in relation to human factors, and that at the same time has a positive impact on individual and social sustainability of the organization.

To achieve the main objective, the following specific objectives are proposed:

- **O1** DevOps characterization. Prior to be able to make a proposal, it is necessary to know the current state of DevOps in relation to human factors. Taking into account the lack of scientific studies in this area, this objective focuses on the study of DevOps through industry information, able to provide, in this case, much closer information than scientific literature about this topic.

- **O2** Proposal of a theory on human factors in the adoption of DevOps. First, it is intended to identify which human factors are related to the adoption of DevOps, and what each of them consists of. Second, it is intended to study the effect of these human factors, how they relate to each other, and the context in which they are found.

- **O3** Proposal of a DevOps adoption model based on human factors. The model must be based on the theory proposed in the previous objective, and must provide implementable actions for the sustainable adoption of DevOps in relation to human factors.

1.7 — Acknowledgements

This doctoral thesis has been completed in the context of the projects “BIZDEVOPS-GLOBAL-UMU” (RTI2018-098309-B-C33) and “OASSIS-UMU” (PID2021-122554OB-C32), both funded by the Spanish Ministry of Science (AEI /10.13039/501100011033/), Innovation, and Universities, and by the European Regional Development Fund. The thesis was also financially supported by the Network of Excellence in Software Quality and Sustainability (TIN2017-90689-REDT). This thesis has been carried out in the Computer Science doctoral program, offered by the International Doctoral School of the University of Murcia (EIDUM).

The author of this thesis was related under a predoctoral research contract with the Software Engineering research group, Department of Computer Science and Systems, University of Murcia. The author has also been related with the Lappeenranta-Lahti University of Technology (LUT) through a research stay in the Department of Software Engineering, that occurred from July 1st, 2022, to September 30th, 2022, for a total of 3 months.

This thesis is presented as a compendium of 4 publications [49, 50, 51, 52], according to the regulations of the International Doctoral School of the University of Murcia. The publications are summarized in Chapter 3 and reported in Chapter 5.

1.8 — Thesis structure

This thesis is structured as follows: Chapter 2 describes the methodology used to achieve the objectives. Chapter 3 presents the results of the study. Chapter 4 presents the conclusions of the study. Chapter 5 presents the articles that make up the doctoral thesis. Finally, Chapter A presents a list of abbreviations used in the thesis.

Chapter 2

Methodology

This doctoral thesis follows the Design Science methodology. Design Science is a research methodology that focuses on creating artifacts, or products, that solve practical problems.

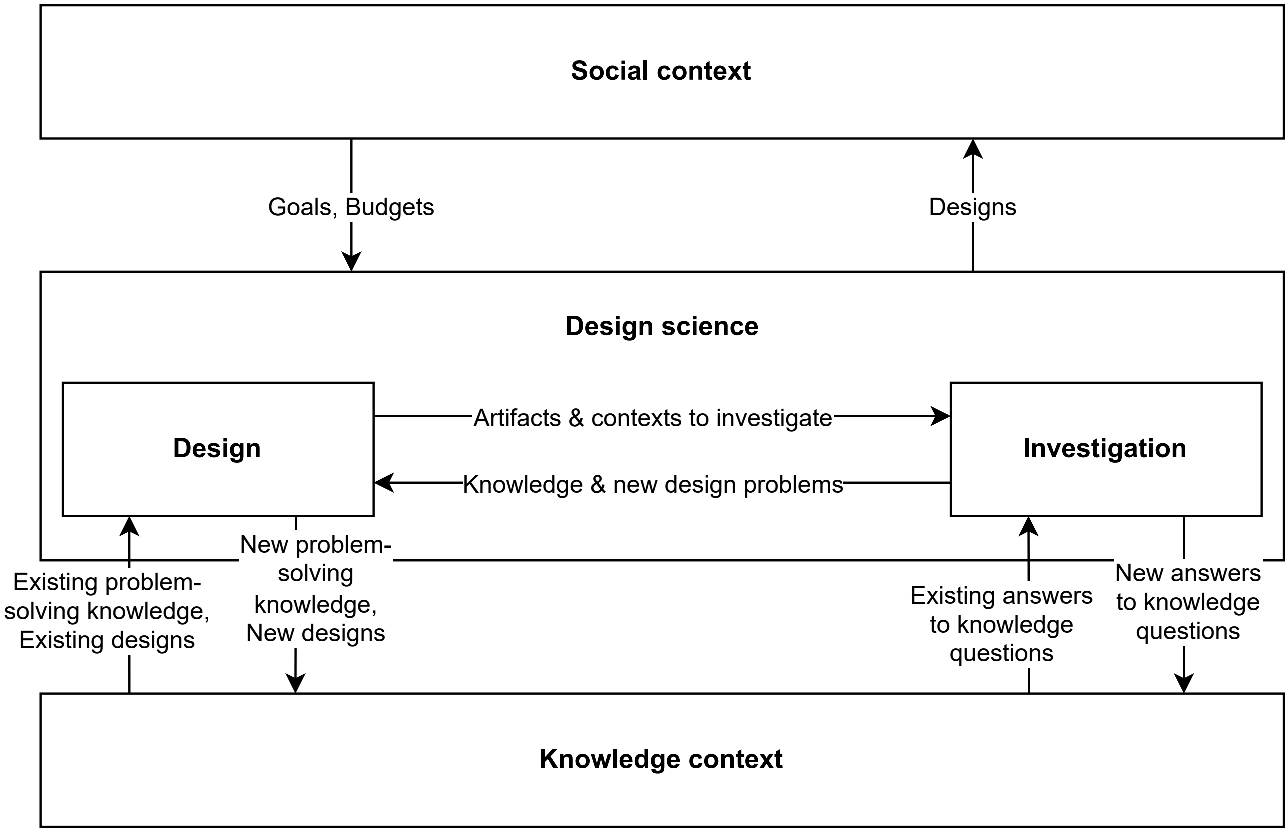

Figure 2.1. Elements of Design Science and their relationship to each other. Extracted from [1].

The main elements of Design Science are shown in Figure 2.1. In the center, the activities of Design Science, i.e., Design and Research, are shown. These activities are related to each other, providing new artifacts to investigate and new knowledge to design new artifacts. In turn, in the same way that research serves society as a whole, the activities of Design Science are directed by the objectives of the social context, receiving objectives and economic budgets, and providing new designs to society. Finally, Design Science is related to the knowledge that is possessed about the field of study (Contextual Knowledge) in its two main activities, Design and Research. In the Design activity, Contextual Knowledge provides knowledge and previous designs, and receives new knowledge and new designs, while in the Research activity, contextual knowledge provides the answers that are already known about theoretical questions, and receives new answers to previously unanswered theoretical questions [1].

Design Science follows an iterative cycle in which activities are carried out over and over again, obtaining new knowledge and new artifacts in each iteration. Figure 2.2 shows the engineering cycle of Design Science, in which a problem is investigated, a treatment for that problem is designed, the treatment is validated, and it is implemented in a practical scenario. The cycle starts again when the implementation is evaluated, a step that would be equivalent to investigating a new problem. Wieringa [1] defines treatment as the interaction between the artifact and the context, with the aim of solving the problem that has been identified.

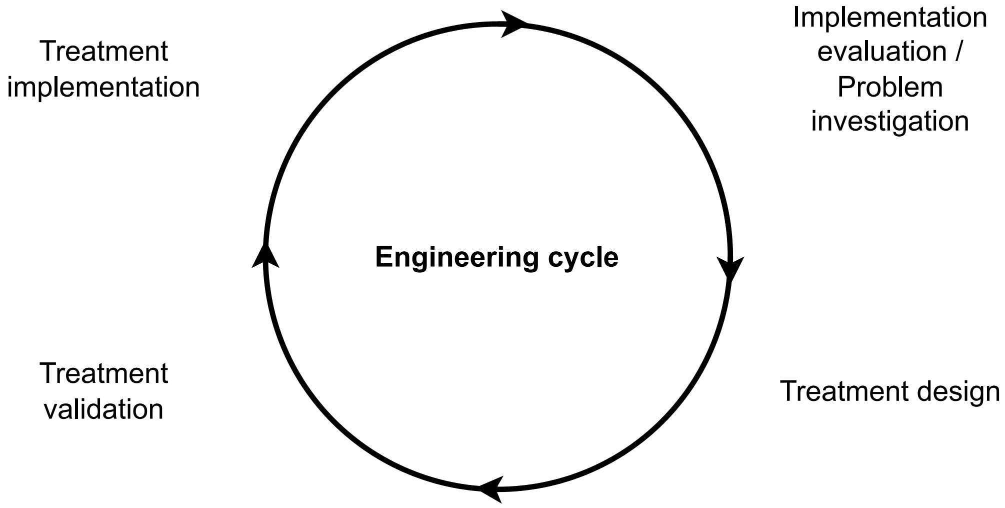

Figure 2.2. Engineering cycle of the Science of Design. Extracted from [1].

In particular, this doctoral thesis follows the Design Science framework defined by Johannesson and Perjons [53]. In the framework, the following main research activities are defined:

1. Explain the problem. Activity in which a practical problem is investigated and explained. The problem must be of general interest and must be formulated and justified precisely.

2. Define the requirements. This activity defines a solution to the identified problem through an artifact and its requirements.

3. Design and develop the artifact. This activity generates an artifact that solves the identified problem and meets the defined requirements.

4. Demonstrate the artifact. This activity is responsible for proving that the artifact is feasible and can solve an instance of the identified problem.

5. Evaluate the artifact. This activity determines how and to what extent the artifact solves the identified problem.

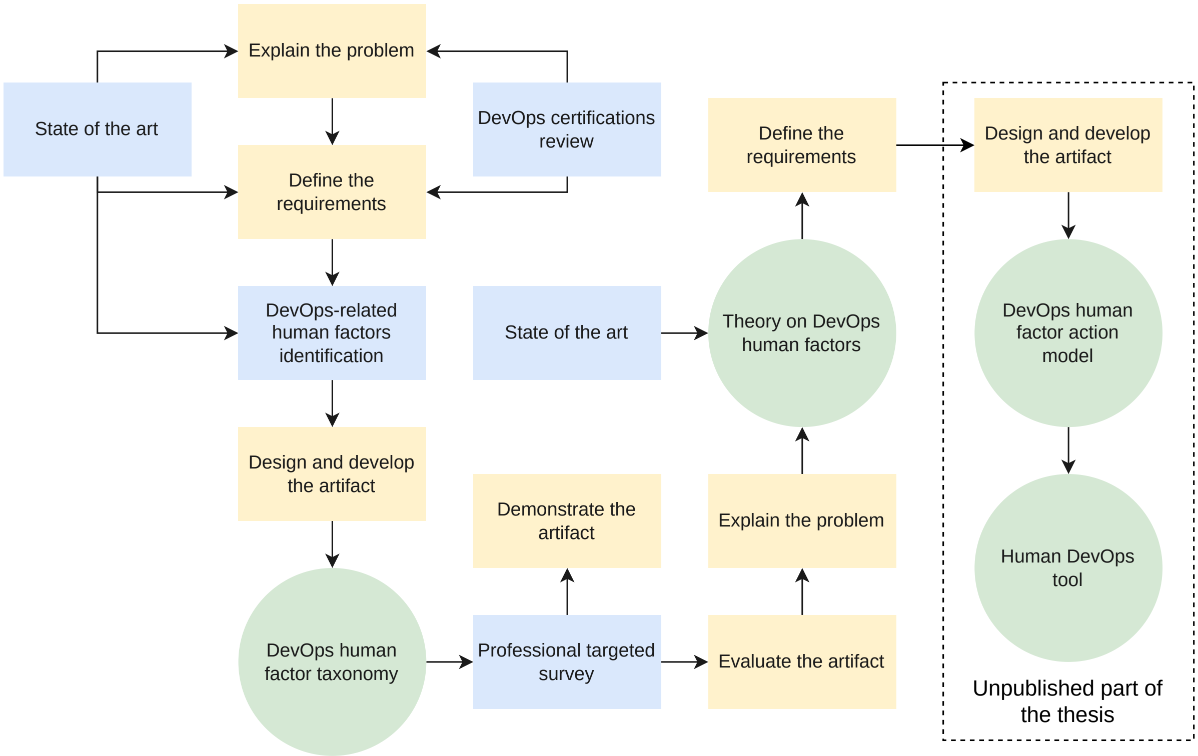

Figure 2.3. Activities carried out in this doctoral thesis and their relationship to the activities of the Johannesson and Perjons framework. Legend: Yellow, activities of the Johannesson and Perjons framework; Blue, activities carried out in this doctoral thesis; Green: Artifacts.

Despite the fact that these activities are presented and connected sequentially, Design Science is iterative by nature, and the researcher must move back and forth between the activities of the framework.

Figure 2.3 shows a simplification of the activities carried out in the doctoral thesis, how they affect the cycle of Design Science activities, and what artifacts have been produced and refined. A complete cycle of activities is carried out, from explaining the problem to evaluating the artifact, and a new cycle is started until the design and development of the artifact.

The revision of DevOps certifications, an activity included in the objective O1, corresponding to the papers reported in Section 5.1 and Section 5.2, serves as a starting point to explain the problem in question, and to define what possible artifact could solve it. Then, the identification of human factors related to DevOps, included in objective O2 and detailed in

Section 5.3, leads to the first artifact of the doctoral thesis, a taxonomy that identifies and describes the human factors that are related to the adoption of DevOps. The artifact is demonstrated and evaluated through a survey aimed at professionals with experience in DevOps. The demonstrated and evaluated artifact serves as a basis for the proposal of a more detailed theory about the context of human factors in the adoption of DevOps, which resulted in the final publication of this thesis, reported in Section 5.4. This new theory partially solves the initial problem, and at the same time explains it in greater depth, leading to a new cycle of activities, which ends with the proposal of two new artifacts, a practical model for measuring and improving human factors in the adoption of DevOps, and a software tool to support the exploitation of this model. These last artifacts respond to objective O3 of this doctoral thesis.

Chapter 3

Summary of results

In this chapter, a summary of the results obtained through the development of the thesis is presented. The results correspond to each one of the papers included in the compendium, ordered by date of publication.

3.1 DevOps Certification in IT Industry: Preliminary Findings

In order to propose a DevOps adoption model, the final objective of the thesis, it is necessary to deeply understand the state of the art in DevOps adoption. However, researchers agree that DevOps research is falling behind the industry advances, and the knowledge available in scientific literature may not be a good representation of the current state of DevOps in industry. Therefore, we decided to approach the issue through industry. Certifications are one of the ways that industry has to acknowledge skills and knowledge in a certain area, and the mechanism is well-established in IT. At this point, we decided to investigate what is DevOps, and which skills are supposedly required to work in a DevOps environment, according to the information provided by DevOps certifications.

In the first paper of the compendium *DevOps Certification in IT Industry: Preliminary Findings* [49], the approach to identify and extract this information is presented. Firstly, every certification that appeared in the first 10 pages of a Google search with the terms "DevOps certification" were considered. Google was chosen as the search engine instead of a scientific database search engine in order to identify certifications present in industry, not necessarily academia. A total of 105 certifications were found, provided by a total of 45 certifying organizations. A survey was designed to collect the information about the certifications directly from these organizations. The survey was structured as follows:

- Questions about the organizations
  - Demographic questions, such as the name of the organization, founding year, number

of employees, etc.

- Partner questions. We found that some organizations delegated their responsibilities on other organizations through a partnership relationship. The questions were designed to identify which partners were involved and what was their role.

- Questions about the certifications
  - Demographic questions, such as the name of the certification, price, number of certifications issued, etc.
  - Exam questions. The main method to assess if a person is ready to receive a certification is through an exam. The questions were designed to identify the methodology of the exam, such as whether it was required or not, online availability, price, etc.
  - Training questions. For some organizations identified, official training was offered to achieve the level of knowledge required to obtain the certification. For some organizations, the official training was both required and the only assessment to obtain a certification. The questions were designed to identify the methodology of the training, such as whether it was required or not, online availability, price, etc.
  - Tools questions. Part of the certifications identified were partially or totally focused on the use of a specific tool. These questions were aimed at identifying which tools were required to obtain the certification.
  - Competencies questions. In order to assess which DevOps competencies were required to obtain the certification, the practices identified by Jabbari et al. [20] were used. Every certification was cross-checked with these practices to identify which competencies were required.

However, after every organization was invited to participate in the survey, for the period of 21 days when the survey was accepting responses, only 2 organizations agreed to participate, with a total of 2 certifications. This was a very low response rate, and the results obtained were not enough to draw any conclusions. These results motivated a change of approach. Instead of requiring the participation of the organizations, the information was collected directly from the certifications publicly available on the organizations' websites. This approach was presented in the second paper of the compendium, described in the next section.

3.2 — DevOps Certifications for IT Professionals

The next paper in the compendium *DevOps Certifications for IT Professionals* [50] focuses on the same objective as the previous paper, which was left incomplete, but with a different approach. In this case, the information was obtained from the certifications directly, without the need for the organizations to participate in the survey.

The certifications were identified through a Google search, as in the previous paper. However, Google was selected among five popular general-purpose search engines (Google, Bing, Yandex, Baidu, and Yahoo), repeating the same search from different locations around the world. Google outperformed every search engine in the task of identifying DevOps certifications. A set of 45 certifications were identified, but only 22 of them included enough data to be analyzed.

Figure 3.1 shows the certifications that were identified, the organizations that provide them, and other relevant demographic data.

Organization	Certification	Nature	Nº DO cert.	Price	Validity period	Access	T.T.	T/C
DevOps Institute	DevOps Foundation	DOCF	7	$245.00	Lifetime	Online	16	2
DevOps Institute	Site Reliability Engineering Foundation	DOCF	7	$245.00	Lifetime	Online	16	8
DevOps Institute	Continuous Delivery Ecosystem Foundation	DOCF	7	$245.00	Lifetime	Online	16	1.6
DevOps Institute	DevOps Test Foundation	DOCF	7	$245.00	Lifetime	Online	16	8
DevOps Institute	Certified Agile Service Manager	DOCF	7	$245.00	Lifetime	Online	16	8
Docker	Docker Certified Associate	NCF	1	$195.00	2 years	Online	20	10
Kubernetes	Certified Kubernetes Administrator	NCF	3	$300.00	3 years	Online	18	4.5
Kubernetes	Certified Kubernetes Application Developer	NCF	3	$300.00	3 years	Online	35	8.75
Amazon Web Service	AWS Certified DevOps Engineer	NCF	1	$300.00	3 years	Online/On-site	20	10
Microsoft Azure	Azure DevOps Engineer Expert	NCF	1	$165.00	2 years	On-site	40	5
DevOps Agile Skills Association	Fundamentals	DOCF	7	$810.00*	Lifetime	Online/On-site	16	8
Puppet	Puppet Professional Certification	NCF	1	$200.00	Lifetime	Online	48	24
Edureka	DevOps Engineer Certificate	CF	1	$449.00	Lifetime	Exam not needed	36	5.14
Cisco DevNet	DevNet Associate	NCF	3	$300.00	3 years	Online	64	16
Cisco DevNet	DevNet Specialist	NCF	3	$300.00	3 years	Online	24	2.4
Cisco DevNet	DevNet Professional	NCF	3	$300.00	3 years	Online	88	8.8
Google Cloud	Professional Cloud DevOps Engineer	NCF	1	$200.00	2 years	On-site	60	6
ICAgile	Professional: Foundations of DevOps	CF	3	$1,695.00	Lifetime	Exam not needed	14	2
ICAgile	Professional: Implementing DevOps	CF	3	$2,450.00	Lifetime	Exam not needed	14	4.67
BCS	BCS Foundation Level Certificate in DevOps	CF	1	$595.00	Lifetime	Online	12	1.71
Simplilearn	DevOps Certification Course	CF	1	$799.00	Lifetime	Exam not needed	56	14
Star Certification	Star certified DevOps expert	CF	1	N/A	Lifetime	Online	N/A	N/A

Figure 3.1. Demographic data of the certifications analyzed. Legend. Access: refers to the way in which the exam can be accessed (Online, On-site, both, or exam not needed). Nature: refers to the nature of the organization that provides the certification, DOCF (DevOps Certification Focused), CF (Certification Focused), NCF (Not Certification Focused). Nº DO cert.: number of certifications provided by the organization. T.T.: Training time required to acquire the knowledge necessary to pass the certification (hours). T/C: Time per competency (hours/competency), calculated as the ratio between the training time and the number of competencies.

Some certifications, for example the DevOps Foundation certification, required a relatively low training time if the number of competencies being certified is taken into account (training time per competency). While the data on how the training time is distributed among the competencies was not available, i.e. we do not know how much training time is spent on each competency, a low average training time per competency could indicate that the competencies are not being deeply analyzed. Therefore, we highlighted the opposite effect for certifications that, on average, a relatively high amount of training time is spent per competency, as shown in Figure 3.2 for the 10 certifications with the highest training time per competency.

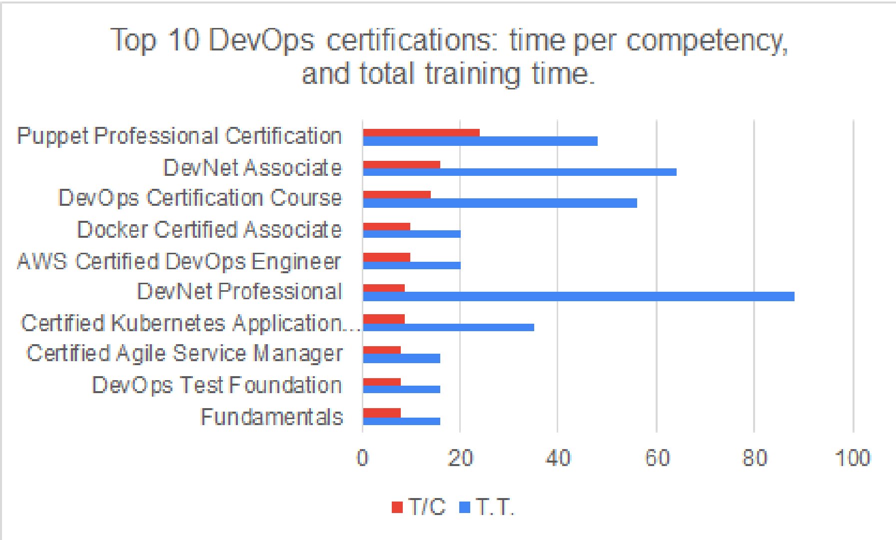

Figure 3.2. The T.T. (Training Time) and T/C (Time per Competency) of the 10 certifications with the highest T/C.

The certifications were analyzed in terms of the competencies that were required to obtain them, in other words, the competencies that are being certified. As in the previous paper, the competencies were extracted from the practices identified by Jabbari et al. [20]. Using the list of competencies presented in each certification, the certifications were classified using a hierarchical clustering technique. The classification can be seen in Table 3.1. A set of 12 clusters was generated.

The certifications were also classified in terms of their context. We differentiated between certifications that were focused on a particular cloud framework (AWS, Azure, and Google Cloud), certifications that were focused on a particular software tool that enables DevOps practices (Docker, Kubernetes, and Puppet), and certifications that were not focused on any particular tool or cloud framework, which were labeled as "Cultural change" certifications. The classification can be seen in Figure 3.3.

Table 3.1

Clustering of the certifications based on the competencies identified. Legend: ***(GSC)*** *General scope.* ***(MON)*** *Monitoring.* ***(TES)*** *Testing.* ***(CID)*** *Continuous integration and delivery.* ***(DEP)*** *Deployment.* ***(MAD)*** *Monitoring and delivery.* ***(GSQ)*** *General scope and quality assurance.* ***(DAT)*** *Delivery and testing.* ***(CON)*** *Configuration.* ***(CDI)*** *Continuous integration, deployment and IaC.* ***(GSM)*** *General scope and change management.* ***(MCT)*** *Monitoring, continuous integration and testing.*

Certification	**Cluster**
DevOps Foundation Continuous Delivery Ecosystem Foundation DevOps Engineer Certificate DevNet Specialist DevNet Professional Professional Cloud DevOps Engineer	GSC
Site Reliability Engineering Foundation	MON
DevOps Test Foundation	TES
Certified Agile Service Manager	CID
Docker Certified Associate	DEP
Certified Kubernetes Administrator Certified Kubernetes Application Developer AWS Certified DevOps Engineer	MAD
Azure DevOps Engineer Expert Professional: Foundations of DevOps	GSQ
Fundamentals	DAT
Puppet Professional Certification	CON
DevNet Associate Professional: Implementing DevOps	CDI
BCS Foundation Level Certificate in DevOps	GSM
DevOps Certification Course Star Certified DevOps Expert	MCT

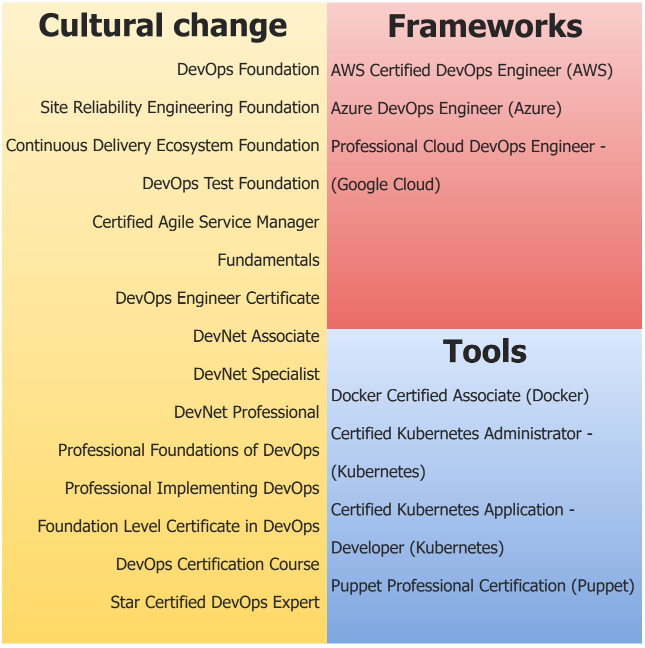

Figure 3.3. Certifications classified by context.

With all the information gathered, a decision diagram was created to help professionals choose the certification that best suits their needs. It should be noted that there are plenty of possible different decision diagrams using the information that was gathered. We opted to present a decision diagram based on the competencies that the certifications include, and in the cluster categorization previously mentioned. The diagram can be seen in Figure 3.4.

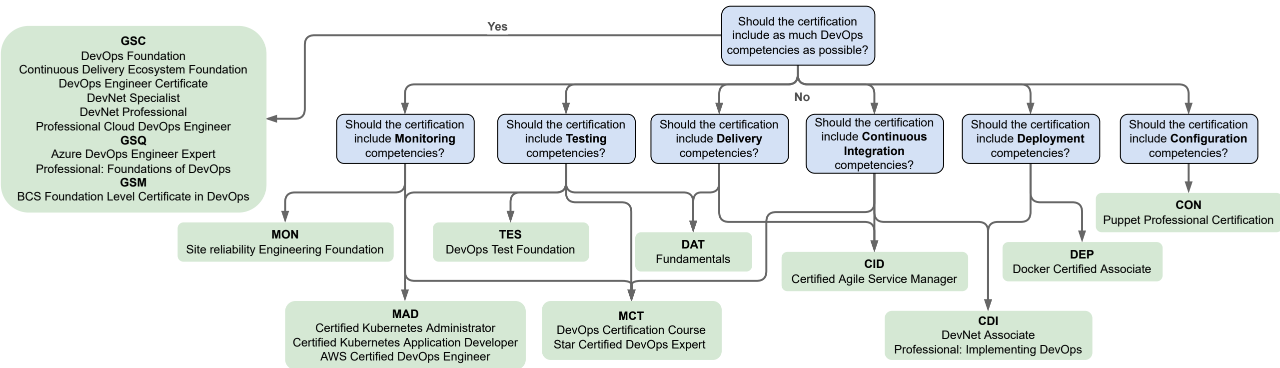

Figure 3.4. One of the possible decision diagrams. The leaf nodes (green) represent one or more clusters that are labeled in bold.

The results of the study were aimed to inform and assist professionals who are interested in obtaining a DevOps certification, but also to extend the research knowledge of the competencies that are required in the industry. This knowledge is directly related with objective O1 of the thesis, which is DevOps characterization. The results of the study allowed us to perceive a lack of attention regarding DevOps non-technical skills. While they were scarcely present in the study written by Jabbari et al. [20], they were even more scarce in the certifications analyzed. This fact motivates the further research regarding DevOps soft-skills, which are both mostly unexplored both in industry and in scientific research.

3.3 — A Taxonomy on Human Factors that Affect DevOps Adoption

The third paper in the compendium *A Taxonomy on Human Factors that Affect DevOps Adoption* [51], is the base for objective O2, i.e., the proposal of a theory on human factors in DevOps adoption. Previously to the development of the paper, a brief literature review was performed, that suggested that the study of DevOps human factors was mostly unexplored, as also supported by the previous paper in the compendium. Since not enough information was available to propose a theory on DevOps human factors, we designed this paper. The paper identifies and classifies the human factors that are related to DevOps adoption through a systematic mapping study. While, as stated, almost no information was available on the subject, the study was performed in a broad way, adapting the knowledge present in other methodologies (not necessarily DevOps), which are intrinsically related to DevOps. This approach of adapting existing knowledge to another discipline has been extensively used and supported in Information Technology research [54].

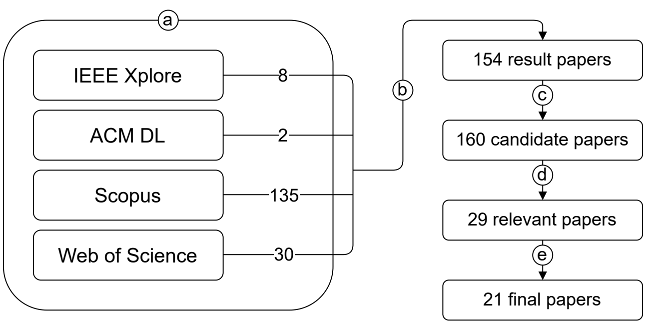

Figure 3.5. Systematic mapping study process diagram. Legend: a) Perform the automatic search using the search string. b) Collect the results returned by the automatic search. c) Snowballing. d) Apply the selection criteria to the candidate papers. e) Apply the quality criteria to the relevant papers.

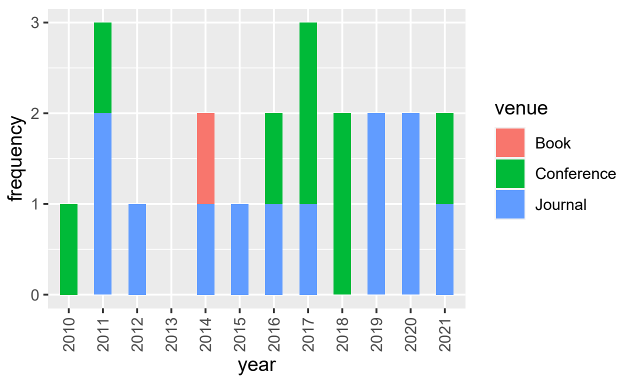

Figure 3.6. Publication venues over 11 years tendency.

After the search protocol depicted in Figure 3.5 was applied, a total of 21 papers were selected that were found topic-relevant, and passed the quality criteria. A steady rate of publication was observed over the last 7 years as shown in Figure 3.6, with a slightly higher frequency in journals than in conferences or books. Regarding the methodology, the majority of the papers used an empirical validation method, always related to industry data, such as practitioner targeted interviews and surveys. However, a substantial amount of selected studies (43%) did not include any means of validation or evaluation.

Most of the selected studies were identified in agile (33%) and lean management (24%) contexts. The rest of them were identified in, from more frequent to less frequent: software process improvement, software development teams, technology adoption, process management automation, and DevOps. It should be noted that only one paper of the 21 selected directly addressed DevOps human factors, which supports the idea that the study of DevOps human factors is mostly unexplored.

A set of 59 human factors was synthesized from the selected studies, based on the original names and descriptions of the factors. The factors were classified in 3 basic categories based on the effect they have on DevOps adoption (based on their synthesized description):

- Positive human factors. These factors are those that, when present, make the adoption of DevOps practices easier. They are the factors that organizations should aim to have in order to facilitate the adoption of DevOps. Their absence is always negative. An example of a positive human factor is "Motivation for change".

- Negative human factors. These factors are those that, when present, make the adoption of DevOps practices harder. They are the factors that organizations should aim to avoid in order to facilitate the adoption of DevOps. Their absence is always positive. An example of a negative human factor is "Negative experiences".

- Ambivalent human factors. These factors are always present, however, their effect depends on how are they treated in the organization, potentially being positive or negative. An example of an ambivalent human factor is "Social influence".

The factors are distributed in the categories as follows: 25 positive human factors (42%), 23 ambivalent human factors (39%), and 11 negative human factors (19%).

The information gathered in this study was used as the foundation to propose a DevOps adoption human factor theory, but also for the extension of DevOps characterization (O1). Through the human factors that affect DevOps adoption, we can understand how stakeholders interact with the methodology, which as the same time, allows us to understand the methodology itself.

3.4 — A Theory on Human Factors in DevOps Adoption

The fourth and last paper in the compendium, *A Theory on Human Factors in DevOps Adoption* [52], achieves objective O2, which is the proposal of a theory on human factors in DevOps adoption. The paper also extends the knowledge on DevOps characterization (O1), in the same way as the previous paper, and serves as the foundation for the development of the DevOps adoption model (O3). The main purpose of the theory is to explain DevOps human factors, which are one of the centerpieces of the doctoral thesis. Since, as supported by the previous paper, almost no knowledge is available on the subject, the theory fills a research gap that would otherwise severely limit the results of the thesis.

The paper is an extension of the previous one, and it is based on the same set of human factors that were synthesized in the previous paper. However, the search was extended to include publications up to July 2024. The extension of the search provided 2 more selected papers that were published in 2022 and 2023. As stated, the added papers did not provide new human factors, but they did support some human factors already identified.

The relevance of the human factors in DevOps adoption was assessed through a practitioner targeted survey. The survey was designed to be answered by professionals who work in IT, and who have experience in DevOps adoption. The participants answered questions about the relevance of each human factor, and a small amount of demographic questions.

The survey was answered by 15 professionals with experience in DevOps environments. As seen in Figure 3.7, most of the participants had between 1 and 7 years of experience in DevOps environments, with 33% of the total participants having exactly 5 years of DevOps experience. The participants were distributed in organizations of different sizes, as shown in Figure 3.8, however, most of them belonged to big organizations with more than 500 employees. Figure 3.9 shows the job position of the participants, presenting a wide variety of positions with no apparent data trends. A majority of 11 participants claimed that their organization managed human factors.

Through a 5 point Likert scale, the participants were asked to rate the relevance of each human factor for DevOps adoption, where 1 was "Strongly irrelevant", and 5 was "Strongly relevant". Only one human factor was found slightly irrelevant by the participants, which was "Neuroticism" (mean = 2.93, median = 3, s.d. = 0.704). Figure 3.10 shows the average responses for the relevance of the identified set of human factors. The figure presents a wide majority of human factors that were found relevant by the participants (81% of them being above 3.5), and almost no human factors that were found irrelevant on average. Therefore, we considered that the participants validated our set of human factors for DevOps adoption, which were used to propose a theory on human factors in DevOps adoption.

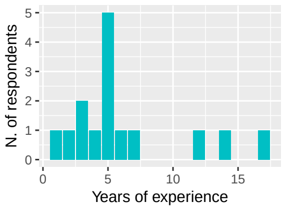

Figure 3.7. Participants' DevOps experience distribution.

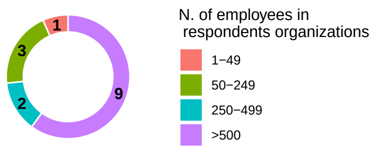

Figure 3.8. Participants' organization size distribution.

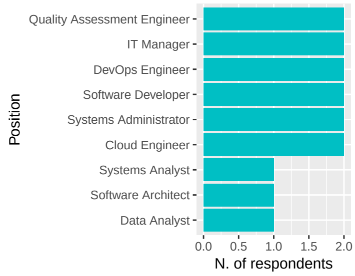

Figure 3.9. Participants' occupation in industry.

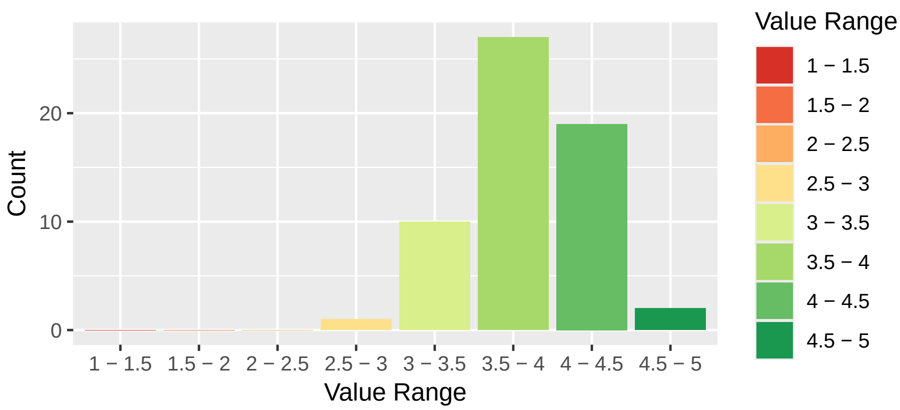

Figure 3.10. Average responses for the relevance of all human factors.

A theory on DevOps human factors was proposed based on Gregor's theory study [2]. The theory, according to Gregor, is a Type I theory, i.e., a theory meant to analyze what is a phenomenon. It is the most basic level of theorizing, needed when very little knowledge is available, such as DevOps human factors' case. The theory is primarily composed of primary constructs and statements of relationship. Table 3.2 shows a summary overview of the theory characteristics.

Table 3.2

Theory overview table (adapted from Gregor [2]).

Theory overview	
The theory on DevOps human factors identify which human factors affect a DevOps adoption, how can they affect the organization, how are they related to people, and how can they be affected by people or by other human factors.	
Theory Component	**Instantiation**
Means of representation	Textual description, diagrams, tables, and figures.
Primary constructs	Human factor, Actor, Action, Resources, Priority level, Organizational performance.
Statements of relationship	An example: *Human factors can, in most cases, be affected voluntarily by the actions of an actor.*
Scope	The relationships between the primary constructs are always present, or depend on the particular primary construct.
Causal explanations	Not present
Testable propositions	Not present
Prescriptive statements	Not present

A total of 6 primary constructs were identified in the theory. The main primary construct, "Human factor" is the centerpiece of the theory. As stated in the introduction, we consider that "human factors" are the set of psychological, behavioral, and sociological aspects of human behavior that are related to the software process. In this theory, the definition still applies. In addition, in the theory, the "Human factor" construct refers to the set of human factors that were identified previously, but it is not limited by it, since the theory is open to the inclusion of new human factors, or modification of existing ones. Human factors exist in the theory with a "degree of presence", that is, a percentage that represents how much a human factor is happening in a particular context in the organization. In a simplified example, we could consider that a professional is experiencing around a 50% of "Imposition" if half of the actions that the professional undertakes are not considered as voluntary. These percentages can also be applied as an average value to measure a human factor in a group of people. Human factors, as human behavior, are not isolated concepts, and they are strongly related to each other in a complex network of relationships. The theory proposes that human factors can affect each other (positively or negatively), and in some cases, they can even pose a requirement relationship between them. For example, "Recognition and rewards" could potentially be a strong requirement for "Respect from management".

Figure 3.11 depicts the theory graphically. The 6 primary constructs appear in the figure, with the relationships between them. The "Actor" primary constructs defines the humans, or in other words, the people that is affected or affects the human factors. In this theory, the actor is always a stakeholder, since every person involved in this context, including the members of the development team, operations, management and client, are directly related to the software product. An actor can affect a human factor actively through one or more "Actions", which requires "Resources" to be performed, both "Action" and "Resources" being primary constructs of the theory. Every action is considered to require resources, since the bare minimum that is required to perform an action is time, which is a relevant resource. It also includes monetary resources, human resources, and any other resource that could be required to perform an action. The theory classifies actions depending on the actor and depending on the frequency. Depending on the actor, they can be classified as actions that can be taken by any actor, or actions that can only be taken by a management role of the organization. Depending on the frequency, they can be classified on actions that are only required once to maintain an effect, and actions that need to be continuously taken to maintain the effect. Finally, every human factor has a "Priority level" in an organization, which is a combined value of the impact of the human factor on the "Organizational performance", and the relevance that the organization gives to the human factor. Every human factor has an impact on organizational performance, may it be directly or indirectly through affecting other human factors.

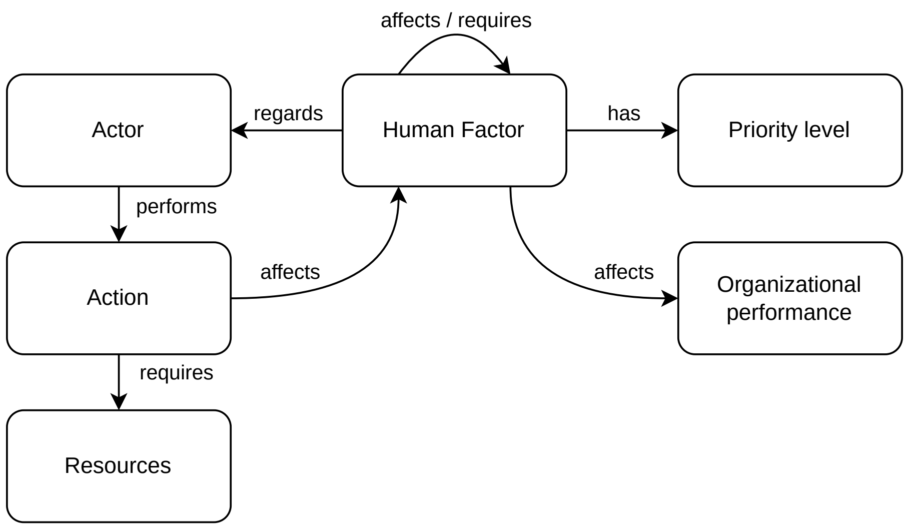

Figure 3.11. Theory graphical description.

The theory proposed is composed by this conceptual framework, the human factor set, and the analysis of the human factors performed. These results complete the research planned for objective O2 of the doctoral thesis. For the completion of the next objective (O3, the proposal of a DevOps adoption model), and the completion of the thesis main objective, the theory proposed describes what human factors are, in which context they are present, and how they affect the DevOps adoption process. This knowledge has been used to define the requirement of a DevOps adoption model that exploits the theory. Besides the knowledge foundation, in order to practically apply the theory, the following requirements were identified:

- Data collection. This mainly means the measurement of the human factors, which requires the design of a measurement instrument. The theory application would benefit from a quantitative and continuous measurement of the human factors. While this is theoretically possible without software support, the use of software would make the process more efficient and less error-prone. Additionally, data collection also means the identification of the actors and human factor priorities.

- Data anonymization. Since the information gathered is highly sensitive, keeping measurements anonymous is a requirement not only to maintain privacy, but also to improve data reliability, as the participants would be more willing to provide honest answers.

- Organizational support. The theory application requires the support of the organization, since the measurement of human factors requires the participation of the organization members. The organization should be willing to provide the necessary resources to perform the measurement, and to act upon the results.

3.5 — DevOps human factor adoption model

This section presents the results of the thesis that are not included in the compendium, but that are part of the thesis main objective. In particular, the results described in this section complete objective O3, and finalize the thesis main objective. In summary, the papers in the compendium previously described DevOps in terms of the competencies that are required in industry and in terms of the human factors that affect its adoption (objective O1). The papers in the compendium also proposed a theory on human factors in DevOps adoption that identifies the human factors that are related to DevOps adoption, deeply study them, and how they affect the process (objective O2). In conjunction with the requirements for the practical application of the theory provided in the last paper of the compendium, the DevOps human factor action model was proposed.

The DevOps human factor action model was based on the set of human factor previously identified, that was extended with existing horizontal knowledge from the literature. The model was designed to be a practical approach that could be used by organizations to improve their DevOps adoption process. The model is composed of 3 elements:

- Human factors. The main element of the action model, they are related to other human factors through dependency relationships (in other words, the source human factor is affected by the target human factor), and to actions.

- Questions. The method to measure each human factor in the action model. The questions are designed to be answered by the organization members. The model differentiates between two organization members: a specific person with a management role, and any member of the team, which also includes the person with a management role. The model is designed to be used in a software development team context, where there is at least one person that acts as the leader, manager, coordinator, or contact with the organization management. Some questions are specifically designed to be answered by this person, as they are related to the management of the team or the organization.

- Actions. The recommendations that the action model provides to improve the state of the human factors in the organization. The actions are designed to be initiated by the management role, in a top-down support spirit.

Figure 3.12 shows an extract of the model, where the skill cluster human factors are included. In the model, every human factor has a directly related action, or depends on another factor, in a relationship that always ends with possible actions, therefore no human factors are left without a possible action. The model is designed to measure human factors with the use of percentage values, as reported in the theory proposed in the previous section. The model is also designed to be used in a continuous way, where the human factors are measured periodically, and the actions are taken to improve the state of the human factors, with a few exceptions, as there are some human factors that do not change over time.

The proposed action model was then used as the theoretical engine of a software tool, designed to automatically manage human factor measurement and action suggestion. The tool was named "Human DevOps", and consist of three main components, a front-end web application, a back-end server, and a Slack app, as depicted in Figure 3.13.

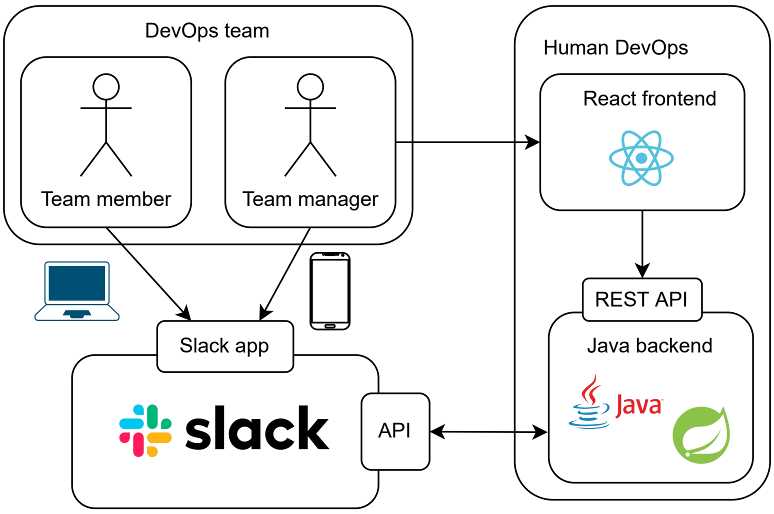

Figure 3.13. Use and implementation diagram of Human DevOps.

The non-management role members of the team are only required to use the tool through the Slack app interface. This was a design choice in order to avoid the need for the team members to access any external tool that is not included in their daily work environment, as Slack

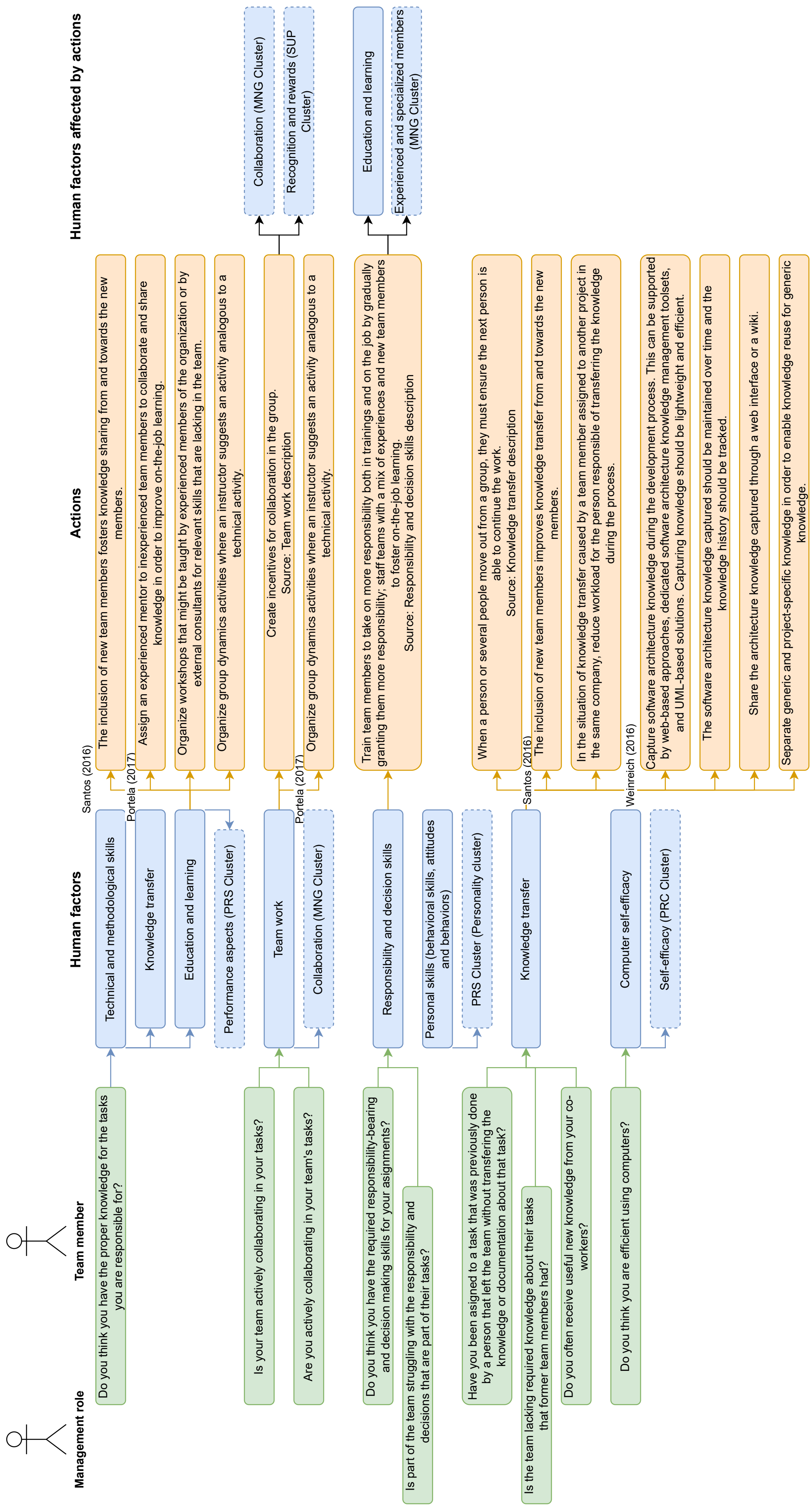

Figure 3.12. DevOps human factor action model extract. Skill cluster.

is one of the most used messaging tools in software development [55, 56]. The model questions are periodically sent to the Slack app, and the team members are required to answer them, taking into account that some questions will only be sent to the team manager. The answers are then sent to the back-end server, where they are processed, and the human factor values are updated. On the other end, the team manager can access the front-end web application, where the human factor values are presented in a graphical way, as presented in Figure 3.14. The network diagram shows every human factor in the model as a dot, colored dark green if the value of the human factor is 100%, dark red if the value of the human factor is 0%, and a gradient of colors in between. Gray dots represent human factors for which not enough answers were collected. At this part of the application, only average values are presented, thus, maintaining individual responses anonymous. The human factors are connected with other human factors based on their dependency relationships mentioned in the model.

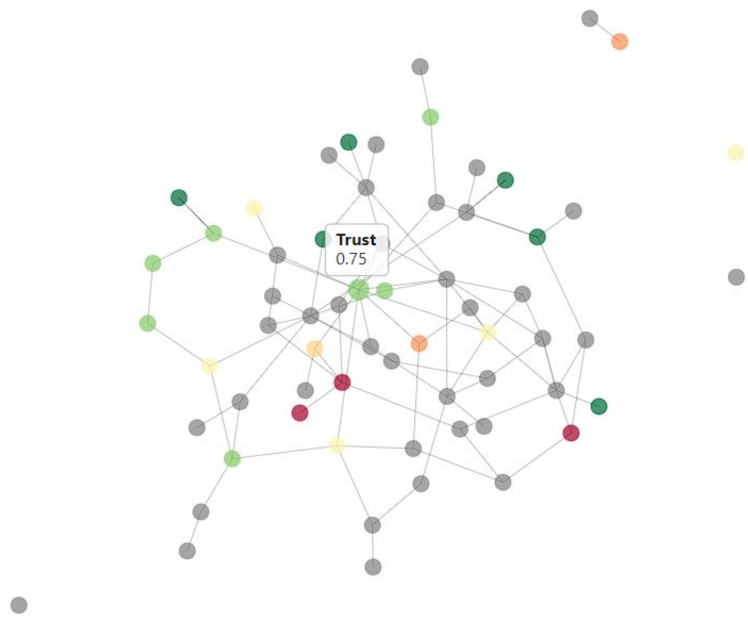

Figure 3.14. Human factor network diagram in Human DevOps.

The team manager can also access the actions recommended by the model, based on the human factor values. The actions are presented in a list, as shown in Figure 3.15. The actions are ordered by a recommendation value, which is also ranging between 0% and 100%, and is calculated based on the human factor affected by the action, and their dependency relationships.

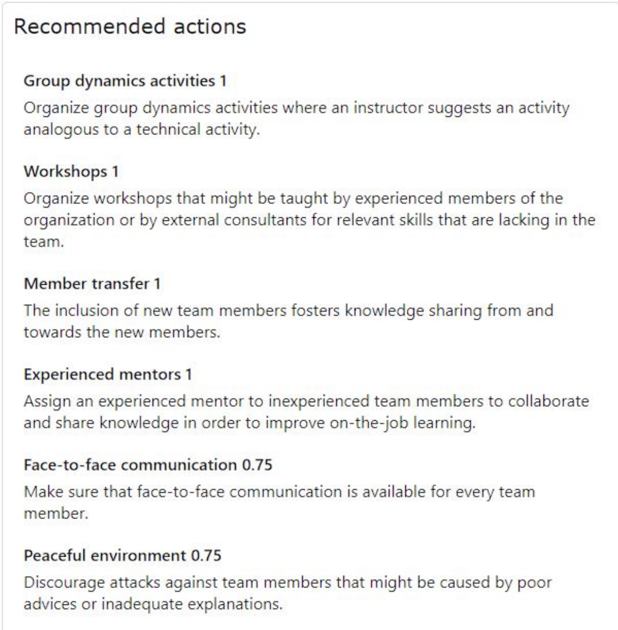

Figure 3.15. Example of actions recommended by Human DevOps.

The tool also includes basic functionality to add and remove members of the team, and change when the questions are sent, besides authentication and password modification.

The action model and the tool "Human DevOps" described in this section are the final results of the thesis, and build a DevOps human factor adoption model, the main objective of the thesis. This human factor adoption model is also aimed to improve both individual and social sustainability. The human factors included in the adoption model are not only related to DevOps adoption success, but also to the well-being of the professionals involved in the process. A healthier and more supportive environment can help professionals maintain a sustainable productivity, and at the same time, can help organizations maintain their human resources, overall improving the sustainability of the software development process.

Chapter 4

Conclusions and future work

DevOps is a software development methodology that is born from the need to improve the software development process, reducing product delivery time, and increasing quality. However, scientific literature does not conclusively answer the questions of what is DevOps and how is DevOps adopted. DevOps adoption is a complex process that involves technical and cultural change. While technical change has been far more studied, cultural change studies have been scarce and insufficient. In particular, human factors, which have been reportedly crucial for the software engineering industry in general, have been barely studied in the context of DevOps.

This thesis had addressed this research gap with the main objective of proposing an adoption model that assists organizations in the DevOps adoption process by considering human factors, that is positive for the organizational performance and for individual and social sustainability. This main objective has been divided in three smaller objectives: to characterize DevOps and identify possible research gaps, to propose a DevOps human factor theory to serve as a theoretical foundation for the adoption model, and to propose a DevOps human factor action model that allows the measurement and improvement of human factors for DevOps adoption. Therefore, the doctoral thesis initial hypothesis, "proper management of human factors can improve the success and sustainability of DevOps adoption", was demonstrated feasible. In summary, this thesis contributed to the DevOps research field through the following results:

- DevOps industry requirements have been identified and analyzed through the main certifications available in the market. The analysis of the certifications has shown that the DevOps industry mainly focuses on technical skills, while the cultural aspects of DevOps are barely addressed.

- A DevOps human factor taxonomy has been proposed to identify and classify the human factors that are related to DevOps adoption. The taxonomy describes and classifies the human factors based on their characteristics and effect on the DevOps adoption process.

- A DevOps human factor theory has been proposed to explain the role of human factors in DevOps adoption. The theory explains how are human factors involved with other software development constructs.

- A DevOps human factor action model has been proposed to serve as a measuring and recommendation framework for DevOps adoption.

- Human DevOps, a software tool that implements the DevOps human factor action model, has been developed to help organizations to measure and improve their DevOps adoption process. The tool has been designed to continuously assist during the DevOps adoption process, and even after the adoption process has been completed, to help organizations to maintain their DevOps practices.

- DevOps was characterized through the study of their human aspects. The human factors, as an intrinsic part of DevOps, explain how professionals impact and are affected by the DevOps adoption process, and at the same time, explain what DevOps is.

- The results of the thesis can improve individual and social sustainability by creating a better work environment, improving the well-being of the professionals involved.

4.1 — Future work

The conduction of this doctoral thesis leaded to the emergence of several future research opportunities. Firstly, while this thesis characterized DevOps through its human factors, DevOps characterization still requires its study from other perspectives to achieve a higher comprehension of what is DevOps. Regarding DevOps human factors, the proposed taxonomy identified human factors mainly belonging to DevOps component methodologies. However, the lack of empiric studies specifically focused on DevOps human factors still remains. From another perspective, the information available for each human factor is scarce, even though human factors represent complex parts of human behavior and cognition that require a deep study to be fully understood. This further study of human factors should extend our knowledge of each phenomenon represented, and how are they inter-related. This might benefit from the adaptation from other fields of knowledge, such as psychology, sociology, or anthropology.

The study of DevOps competencies in the industry has suggested a lack of professional knowledge of DevOps cultural aspect. This lack of knowledge could be addressed by the development of training programs that focus or include the management of human factors, both in industry certifications or in academic programs.

The proposals of the theory, the action model, and the software tool, are a first step towards the understanding and improvement of human factors in DevOps adoption. However, the empirical validation of these proposals is still pending. The theory, as previously stated, is categorized as a type I theory, which is the lowest level of theory. Further study on the topic could lead to the development of more a more detailed theory that could offer testable hypotheses and predictions. The software tool has been developed as a proof of concept. The tool could be improved by adding more features, such as the integration with other messaging software, more detailed reports, and more options for configuration. The tool might also offer raw empirical data if implemented in real scenarios, that could potentially be used to further understand the human factors and actions included in the model. In particular, the training of an AI model with the data collected by the tool could be used to make predictions about the human factors, measure their real impact, and recommend actions to improve them.

Additionally, the adoption model proposed is specifically tailored for the improvement of human factors. The model could be included as a part of a bigger model that takes into account other aspects of DevOps adoption, such as technical change, or organizational change. Furthermore, the knowledge included in the model and in the taxonomy could be evaluated in other software development methodologies, as it might be generalized to a broader field of research.

Chapter 5

Publications that compose the doctoral thesis

5.1 DevOps Certification in IT Industry: Preliminary Findings

Abstract: DevOps is a methodology which pursues the improvement of software delivery time and quality. IT companies lack models or standards that help them through the adoption process. Furthermore, there is no consensus of what DevOps is composed of, and many researchers have identified several challenges in the adoption process. In this paper we describe our attempts to analyze the approaches of industry for DevOps adoption. A survey aimed for DevOps certifying organizations was carried out. The responses of the survey were scarce, and the preliminary findings have been reported.

Title	DevOps Certification in IT Industry: Preliminary Findings
Authors	Juanjo Pérez-Sánchez, Joaquín Nicolás Ros, Juan Manuel Carrillo de Gea
Conference	WorldCIST 2021
ICORE Rank (2021)	C
Proceedings	Trends and Applications in Information Systems and Technologies
Date	March 2021
ISSN	2194-5357
EISSN	2194-5365
DOI	https://doi.org/10.1007/978-3-030-72654-6_45
State	Published
Contribution	Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization

5.2 DevOps Certifications for IT Professionals

Abstract: DevOps is a software development methodology which is now mainstream and has been adopted, as reference, by sector leading companies like Google, Amazon, or Microsoft. As DevOps has become a part of the IT industry, IT organizations require trained professionals to cope with this new development methodology. Regarding training, professional certifications have been the main way to achieve skill recognition. However, choosing the right certification is not an easy task. In fact, through a simple Google search, we were able to find more than 100 DevOps-related certifications. In order to facilitate the selection of a certification, our manuscript reviews, describes, and synthesizes 45 certifications available on market. The information of the practices present in each certification has been described and synthesized in order to assist professionals in the selection of the certification that best fits their needs.

Title	DevOps Certifications for IT Professionals
Authors	Juanjo Pérez-Sánchez, Joaquín Nicolás Ros, Juan Manuel Carrillo de Gea, José Luis Fernández-Alemán
Journal	Computer
Impact factor (2022)	2.2
JCR Rank (2022)	Computer Science, Software Engineering: 58/108
Publisher	IEEE Computer Society
Date	November 2022
ISSN	0018-9162
EISSN	1558-0814
DOI	https://doi.org/10.1109/MC.2022.3144068
State	Published
Contribution	Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization

5.3 A Taxonomy on Human Factors that Affect DevOps Adoption

Abstract: DevOps is a software development methodology created to reduce or even remove the division between the Development (Dev) and Operations (Ops) teams. However, DevOps adoption requires overcoming several impediments, and between them, culture change and human factors have the biggest impact. Therefore, this paper addresses the challenge of DevOps adoption from the perspective of DevOps culture and human factors. A systematic mapping study was carried out to create a taxonomy of human factors affecting DevOps adoption. A total of 21 studies were selected, and 59 human factors were included in the taxonomy after the extraction and synthesis processes.

Title	A Taxonomy on Human Factors that Affect DevOps Adoption
Authors	Juanjo Pérez-Sánchez, Saima Rafi, Juan Manuel Carrillo de Gea, Joaquín Nicolás Ros, José Luis Fernández Alemán
Conference	WorldCIST 2024
ICORE Rank (2023)	C
Proceedings	Trends and Applications in Information Systems and Technologies
Date	May 2024
ISSN	2194-5357
EISSN	2194-5365
DOI	https://doi.org/10.1007/978-3-031-60221-4_31
State	Published
Contribution	Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization

5.4 A Theory on Human Factors in DevOps Adoption

Abstract: Context: DevOps is a software engineering paradigm that enables faster deliveries and higher quality products. However, DevOps adoption is a complex process that is still insufficiently supported by research. In addition, human factors are the main difficulty for a successful DevOps adoption, although very few studies address this topic. **Objective:** This paper addresses two research gaps identified in literature, namely: (1) the characterization of DevOps from the perspective of human factors, i.e. the description of DevOps' human characteristics to better define it, and (2) the identification and analysis of human factors' effect in the adoption of DevOps. **Method:** We employed a hybrid methodology that included a Systematic Mapping Study followed by the application of a clustering technique. A questionnaire for DevOps practitioners ($n = 15$) was employed as an evaluation method. **Results:** A total of 59 human factors related to DevOps were identified, described, and synthesized. The results were used to build a theory on DevOps human factors. **Conclusion:** The main contribution of this paper is a theory proposal regarding human factors in DevOps adoption. The evaluation results show that almost every human factor identified in the mapping study was found relevant in DevOps adoption. The results of the study represent an extension of DevOps characterization and a first approximation to human factors in DevOps adoption.

Title	A theory on human factors in DevOps adoption
Authors	Juanjo Pérez-Sánchez, Saima Rafi, Juan Manuel Carrillo de Gea, Joaquín Nicolás Ros, José Luis Fernández Alemán
Journal	Computer Standards & Interfaces
Impact factor (2023)	4.1
JCR Rank (2022)	Computer Science, Software Engineering: 20/131
Publisher	Elsevier
Date	August 2024
ISSN	0920-5489
EISSN	1872-7018
DOI	https://doi.org/10.1016/j.csi.2024.103907
State	Published
Contribution	Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Data curation, Conceptualization

Bibliography

[1] Roel J. Wieringa. Design Science Methodology for Information Systems and Software Engineering. Springer Berlin Heidelberg, 2014.

[2] Shirley D Gregor. The nature of theory in information systems. MIS Q., 30:611–642, 2006.

[3] Katherine Calvin, Dipak Dasgupta, Gerhard Krinner, Aditi Mukherji, Peter W. Thorne, Christopher Trisos, José Romero, Paulina Aldunce, Ko Barrett, Gabriel Blanco, William W.L. Cheung, Sarah Connors, Fatima Denton, Aïda Diongue-Niang, David Dodman, Matthias Garschagen, Oliver Geden, Bronwyn Hayward, Christopher Jones, Frank Jotzo, Thelma Krug, Rodel Lasco, Yune-Yi Lee, Valérie Masson-Delmotte, Malte Meinshausen, Katja Mintenbeck, Abdalah Mokssit, Friederike E.L. Otto, Minal Pathak, Anna Pirani, Elvira Poloczanska, Hans-Otto Pörtner, Aromar Revi, Debra C. Roberts, Joyashree Roy, Alex C. Ruane, Jim Skea, Priyadarshi R. Shukla, Raphael Slade, Aimée Slangen, Youba Sokona, Anna A. Sörensson, Melinda Tignor, Detlef van Vuuren, Yi-Ming Wei, Harald Winkler, Panmao Zhai, Zinta Zommers, Jean-Charles Hourcade, Francis X. Johnson, Shonali Pachauri, Nicholas P. Simpson, Chandni Singh, Adelle Thomas, Edmond Totin, Andrés Alegría, Kyle Armour, Birgit Bednar-Friedl, Kornelis Blok, Guéladio Cissé, Frank Dentener, Siri Eriksen, Erich Fischer, Gregory Garner, Céline Guivarch, Marjolijn Haasnoot, Gerrit Hansen, Mathias Hauser, Ed Hawkins, Tim Hermans, Robert Kopp, Noëmie Leprince-Ringuet, Jared Lewis, Debora Ley, Chloé Ludden, Leila Niamir, Zebedee Nicholls, Shreya Some, Sophie Szopa, Blair Trewin, Kaj-Ivar van der Wijst, Gundula Winter, Maximilian Witting, Arlene Birt, and Meeyoung Ha. Ipcc, 2023: Climate change 2023: Synthesis report. contribution of working groups i, ii and iii to the sixth assessment report of the intergovernmental panel on climate change [core writing team, h. lee and j. romero (eds.)]. ipcc, geneva, switzerland., 7 2023.

[4] Charlotte Freitag, Mike Berners-Lee, Kelly Widdicks, Bran Knowles, Gordon S. Blair, and Adrian Friday. The real climate and transformative impact of ict: A critique of estimates, trends, and regulations. Patterns, 2:100340, 9 2021.

[5] Christoph Becker, Ruzanna Chitchyan, Leticia Duboc, Steve Easterbrook, Birgit Penzenstadler, Norbert Seyff, and Colin C. Venters. Sustainability design and software: The Karlskrona Manifesto. Proceedings - International Conference on Software Engineering, 2:467–476, 2015.

[6] Chintan Amrit, Maya Daneva, and Daniela Damian. Human factors in software development: On its underlying theories and the value of learning from related disciplines. a guest editorial introduction to the special issue. Information and Software Technology, 56:1537–1542, 12 2014.

[7] E. Dijkstra. Programming considered as a human activity, page 1–9. Yourdon Press, USA, 1979.

[8] G. M. Weinberg. The psychology of computer programming. Van Nostrand Reinhold Co., USA, 1988.

[9] Tom DeMarco and Timothy Lister. Peopleware: productive projects and teams. Dorset House Publishing Co., Inc., 1987.

[10] David E. Avison, Francis Lau, Michael D. Myers, and Peter Axel Nielsen. Action research. Communications of the ACM, 42:94–97, 1999.

[11] Konstantina Richter and Reiner R. Dumke. Modeling, Evaluating, and Predicting IT Human Resources Performance. Auerbach Publications, 3 2015.

[12] J. Appelo. Management 3.0: Leading Agile Developers, Developing Agile Leaders. Addison-Wesley signature series. Addison-Wesley, 2010.

[13] Mary Sánchez-Gordón and Ricardo Colomo-Palacios. Taking the emotional pulse of software engineering — a systematic literature review of empirical studies. Information and Software Technology, 115:23–43, 11 2019.

[14] Luiz Fernando Capretz, Faheem Ahmed, and Fabio Q.B. da Silva. Soft sides of software. Information and Software Technology, 92:92–94, 12 2017.

[15] Aline Chagas, Melquizedequi Santos, Célio Santana, and Alexandre Vasconcelos. The impact of human factors on agile projects. In 2015 Agile Conference, pages 87–91, 2015.

[16] Viktoria Stray, Raluca Florea, and Lucas Paruch. Exploring human factors of the agile software tester. Software Quality Journal, 30:455–481, 2022.

[17] Cristiano Jesus, Eduardo Pontes, Rui M. Lima, and Sérgio Ivan Lopes. Adopting industry 4.0 and Lean practices in heavy metalworking: Impact of human factors on productivity. In WorldCIST 2023, volume 801, pages 241–252, 2023.

[18] Benjamin Adrian, Sven Hinrichsen, Alexander Nikolenko, and Frederic Meyer. How to combine Lean, human factors and digital manufacturing – a teaching concept. In Advances in Human Factors and Systems Interaction, volume 959, pages 45–55, 2019.

[19] Johannes Wettinger, Uwe Breitenbücher, and Frank Leymann. DevOpSlang – Bridging the gap between development and operations. In Computers in Industry, pages 108–122, Berlin, Heidelberg, feb 2014. Springer Berlin Heidelberg.

[20] Ramtin Jabbari, Nauman bin Ali, Kai Petersen, and Binish Tanveer. What is DevOps? In Proceedings of the Scientific Workshop Proceedings of XP2016, volume 24-May-201, pages 1–11, New York, NY, USA, may 2016. ACM.

[21] Lucy Ellen Lwakatare, Terhi Kilamo, Teemu Karvonen, Tanja Sauvola, Ville Heikkilä, Juha Itkonen, Pasi Kuvaja, Tommi Mikkonen, Markku Oivo, and Casper Lassenius. DevOps in practice: A multiple case study of five companies. Information and Software Technology, 114(April):217–230, oct 2019.

[22] Aymeric Hemon, Barbara Lyonnet, Frantz Rowe, and Brian Fitzgerald. From Agile to DevOps: Smart skills and collaborations. Information Systems Frontiers, 22(4):927–945, aug 2020.

[23] F. M. A. Erich, C. Amrit, and M. Daneva. A qualitative study of DevOps usage in practice. Journal of Software: Evolution and Process, 29, 6 2017.

[24] Daniel Stahl, Torvald Martensson, and Jan Bosch. Continuous practices and DevOps: beyond the buzz, what does it all mean? In 2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pages 440–448. IEEE, 9 2017.

[25] Ruth W. Macarthy and Julian M. Bass. An empirical taxonomy of DevOps in practice. Proceedings - 46th Euromicro Conference on Software Engineering and Advanced Applications, SEAA 2020, pages 221–228, 2020.

[26] Patrick Debois. DevOps: A software revolution in the making? Cutter IT Journal, 24(8):1–41, 2011.

[27] Jez Humble and Joanne Molesky. Why enterprises must adopt DevOps to enable continuous delivery. Cutter IT Journal, 24(8):6–412, 2011.

[28] Mali Senapathi, Jim Buchan, and Hady Osman. DevOps capabilities, practices, and challenges: Insights from a case study. ACM International Conference Proceeding Series, Part F1377, 2018.

[29] Nicole Forsgren, Jez Humble, and Gene Kim. Accelerate: The Science of Lean Software and DevOps Building and Scaling High Performing Technology Organizations. IT Revolution Press, 1 edition, 2018.

[30] DORA and Google Cloud. State of DevOps report 2023. Technical report, DORA and Google Cloud, 2023.

[31] Lucy Ellen Lwakatare, Pasi Kuvaja, and Markku Oivo. Relationship of DevOps to Agile, Lean and Continuous Deployment. In Pekka Abrahamsson, Andreas Jedlitschka, Anh Nguyen Duc, Michael Felderer, Sousuke Amasaki, and Tommi Mikkonen, editors, Product-Focused Software Process Improvement, volume 10027, pages 399–415. Springer International Publishing, 2016.

[32] Daniel Lopez-Fernandez, Jessica Diaz, Javier Garcia, Jorge Perez, and Angel Gonzalez-Prieto. DevOps team structures: Characterization and implications. IEEE Transactions on Software Engineering, 48:3716–3736, 10 2022.

[33] Leonardo Leite, Nelson Lago, Claudia Melo, Fabio Kon, and Paulo Meirelles. A theory of organizational structures for development and infrastructure professionals. IEEE Transactions on Software Engineering, 49:1898–1911, 4 2023.

[34] Jessica Díaz, Jorge Pérez, Isaque Alves, Fabio Kon, Leonardo Leite, Paulo Meirelles, and Carla Rocha. Harmonizing DevOps taxonomies — a grounded theory study. Journal of Systems and Software, 208:111908, 2 2024.

[35] Jessica Díaz, Daniel López-Fernández, Jorge Pérez, and Ángel González-Prieto. Why are many businesses instilling a DevOps culture into their organization? Empirical Software Engineering, 26:25, 3 2021.

[36] Leonardo Leite, Carla Rocha, Fabio Kon, Dejan Milojicic, and Paulo Meirelles. A survey of DevOps concepts and challenges. ACM Computing Surveys, 52:1–35, 1 2020.

[37] Rico de Feijter, Sietse Overbeek, Rob van Vliet, Erik Jagroep, and Sjaak Brinkkemper. DevOps competences and maturity for software producing organizations. In Enterprise, Business-Process and Information Systems Modeling, pages 244–259, Cham, 2018. Springer International Publishing.

[38] Sher Badshah, Arif Ali Khan, and Bilal Khan. Towards process improvement in DevOps: A systematic literature review. In Proceedings of the 24th International Conference on Evaluation and Assessment in Software Engineering, EASE '20, page 427–433, New York, NY, USA, 2020. Association for Computing Machinery.

[39] Aymeric Hemon, Barbara Lyonnet, Frantz Rowe, and Brian Fitzgerald. Conceptualizing the transition from Agile to DevOps: A maturity model for a smarter is function. In Smart Working, Living and Organising, pages 209–223, Cham, 2019. Springer International Publishing.

[40] Daniel Teixeira, Rúben Pereira, Telmo Henriques, Miguel Mira Da Silva, João Faustino, and Miguel Silva. A maturity model for DevOps. International Journal of Agile Systems and Management, 13(4):464–511, 2020.

[41] Ineta Bucena and Marite Kirikova. Simplifying the DevOps adoption process. In BIR Workshops, 2017.

[42] Saima Rafi, Wu Yu, Muhammad Azeem Akbar, Sajjad Mahmood, Ahmed Alsanad, and Abdu Gumaei. Readiness model for DevOps implementation in software organizations. Journal of Software: Evolution and Process, 33(4):e2323, 2021. e2323 JSME-20-0106.R1.

[43] Carl Marnewick and Josef Langerman. Devops and organizational performance: The fallacy of chasing maturity. IEEE Software, 38:48–55, 9 2021.

[44] Ricardo Amaro, Ruben Pereira, and Miguel Mira da Silva. Capabilities and practices in DevOps: A multivocal literature review. IEEE Transactions on Software Engineering, 49:883–901, 2 2023.

[45] Mary Sánchez-Gordón and Ricardo Colomo-Palacios. Characterizing DevOps culture: A systematic literature review. In Communications in Computer and Information Science, volume 918, pages 3–15, 2018.

[46] Luz Marcela Restrepo-Tamayo and Gloria Piedad Gasca-Hurtado. Human aspects in software development: A systematic mapping study. In International Conference on Collaboration Technologies and Social Computing, volume 13632 LNCS, pages 1–22. Springer Science and Business Media Deutschland GmbH, 2022.

[47] M. Zulfahmi Toh, Shamsul Sahibuddin, and Mohd Naz'ri Mahrin. Adoption issues in DevOps from the perspective of continuous delivery pipeline. In Proceedings of the 2019 8th International Conference on Software and Computer Applications, volume Part F1479, pages 173–177. ACM, 2 2019.

[48] Mauro Lourenço Pedra, Mônica Ferreira da Silva, and Leonardo Guerreiro Azevedo. DevOps adoption: Eight emergent perspectives. ArXiv, 9 2021.

[49] Juanjo Pérez-Sánchez, Joaquín Nicolás Ros, and Juan Manuel Carrillo de Gea. DevOps certification in IT industry: Preliminary findings. In Trends and Applications in Information Systems and Technologies - WorldCIST 2021, pages 473–479, 2021.

[50] Juanjo Pérez-Sánchez, Joaquín Nicolás Ros, Juan Manuel Carrillo de Gea, and José Luis Fernández Alemán. DevOps certifications for IT professionals. Computer, 55(11):62–73, 2022.

[51] Juanjo Pérez-Sánchez, Saima Rafi, Juan Manuel Carrillo de Gea, Joaquín Nicolás Ros, and José Luis Fernández Alemán. A taxonomy on human factors that affect DevOps adoption. In Álvaro Rocha, Hojjat Adeli, Gintautas Dzemyda, Fernando Moreira, and Aneta Poniszewska-Maranda, editors, Good Practices and New Perspectives in Information Systems and Technologies - WorldCIST 2024, Volume 3, Lodz, Poland, 26-28 March 2024, volume 987 of Lecture Notes in Networks and Systems, pages 314–324. Springer, 2024.

[52] Juanjo Pérez-Sánchez, Saima Rafi, Juan Manuel Carrillo de Gea, Joaquín Nicolás Ros, and José Luis Fernández Alemán. A theory on human factors in DevOps adoption. Computer Standards & Interfaces, 92:103907, 2025.

[53] Paul Johannesson and Erik Perjons. An Introduction to Design Science. Springer International Publishing, 9 2021.

[54] Annemette Kjærgaard and Morten Thanning Vendelø. The role of theory adaptation in the making of a reference discipline. Information and Organization, 25:137–149, 7 2015.

[55] Esteban Parra, Mohammad Alahmadi, Ashley Ellis, and Sonia Haiduc. A comparative study and analysis of developer communications on Slack and Gitter. Empirical Software Engineering, 27:40, 3 2022.

[56] Viktoria Stray and Nils Brede Moe. Understanding coordination in global software engineering: A mixed-methods study on the use of meetings and Slack. Journal of Systems and Software, 170:110717, 12 2020.

Appendix A

Abbreviations

Table A.1

List of abbreviations and their meanings.

Abbreviation	**Meaning**
°C	Celsius degrees
AHFE	International Conference on Applied Human Factors and Ergonomics
AI	Artificial Intelligence
CHASE	International Conference on Cooperative and Human Aspects of Software Engineering
EIDUM	International Doctoral School of the University of Murcia
$GtCO_2eq$	Gigatons of carbon dioxide equivalent
IaC	Infrastructure as Code
ICT	Information and Communication Technologies
IPCC	Intergovernmental Panel on Climate Change
IT	Information Technologies
LUT	Lappeenranta-Lahti University of Technology
TIC	Tecnologías de la Información y la Comunicación
UNFCCC	United Nations Framework Convention on Climate Change