Revista Electrónica de Didáctica en Educación Superior

Publicación de acceso libre. ISSN: 1853-3159

Número 23 (2025)

http://ojs.cbc.uba.ar/index.php/redes

Aprendizaje de la ingeniería de software desarrollando soft skills. Un enfoque inmersivo.

Sonia Pamplona
Departamento de Didáctica de las Ciencias Experimentales, Sociales y Matemáticas
Universidad Complutense de Madrid

Nelson Medinilla
Departamento de Lenguajes y Sistemas Informáticos en Ingeniería de Software
Universidad Politécnica de Madrid

Tomás San Feliu
Departamento de Lenguajes y Sistemas Informáticos en Ingeniería de Software
Universidad Politécnica de Madrid

Fecha de recepción:	16 / Mayo /2025
Fecha de aceptación:	27/ Noviembre /2025
Resumen:	Hoy en día, la ingeniería del software ha experimentado un cambio de paradigma para poder afrontar la incertidumbre. En consecuencia, la educación en ingeniería del software debe considerar esta nueva situación. En este trabajo presentamos una investigación sobre la implementación de un enfoque de aprendizaje orientado al desarrollo de competencias propias de la ingeniería de software mediante una colaboración intensiva. Este enfoque holístico implica que el factor humano, y por tanto las soft skills, desempeñan un papel fundamental.

	Se utilizó la metodología de estudio de caso cualitativo para identificar los efectos del enfoque de aprendizaje en el aula y los resultados obtenidos al final del curso. Encontramos tres elementos clave para el rendimiento: el desafío que supone la construcción de un sistema software, la modulación del grado de libertad concedido al alumnado y la motivación generada por dicho desafío. Además, uno de los resultados más relevantes es que los estudiantes tomaron conciencia de la dimensión humana de la ingeniería del software y desarrollaron soft skills muy significativas. El trabajo ofrece dos contribuciones significativas a la comunidad educativa en ingeniería de software: el diseño de nuestro enfoque y los resultados obtenidos con su aplicación. El primero constituye un posible referente para contextos afines, y el segundo permite profundizar en la comprensión de las dinámicas que emergen en el aula al aplicar este enfoque de aprendizaje.
Palabras clave:	Educación en ingeniería de software, soft skills, competencias socioemocionales, competencias socioemocionales, colaboración, factores humanos, estudio de caso cualitativo.
Abstract:	**Learning Software Engineering. Developing Soft Skills: An Immersive Approach** Today, software engineering has made a paradigm shift in order to deal with uncertainty. Consequently, software engineering education must consider this new situation. We present an educational research on the implementation of a learning approach aimed at developing software engineering competencies through intensive collaboration. This holistic approach implies that the human factor and therefore soft skills play a significant role. We conducted a qualitative case study to discover the effects of the learning approach in the classroom and the results obtained at the end of the course. We found three elements that were key to performance: the challenge provided by the construction of a software system, the modulation of the freedom granted to the students, and the motivation provided by the challenge. Moreover, one of the most important results is that students became aware of the human dimension of software engineering and developed very relevant soft skills. We provide two contributions to the educational community: the design of our approach and the results obtained with it. The former can serve as a reference for similar scenarios, and the latter can help to understand what is going on in the classroom when such approaches are being used.
Keywords:	Software engineering education, soft skills, collaboration, human factors, qualitative case study.

Introducción

La Ingeniería del Software (IS) atañe a todos los que se dedican a hacer software porque se ocupa de cómo desarrollar este producto de forma eficaz, eficiente y en tiempo (Yourdon & Constantine, 1979). Su papel es clave en cualquier empresa de software independiente del tamaño y del tipo de proyectos que ejecuten. Por esta causa, las empresas demandan de las Escuelas Universitarias de Informática que sus graduados tengan una primera aproximación a las competencias de IS.

Desde los comienzos hasta casi nuestros días (y todavía) la IS ha estado sintonizada con el paradigma de la Ciencia Moderna, nacida en el siglo XVII. Es decir, ha tenido una cosmovisión determinista donde "[….] there is nothing so far distance that one cannot finally reach nor so hidden that one cannot discover" (Descartes, 1649). La Ciencia Moderna y la IS temprana han supuesto que la incertidumbre es erradicable como establece Descartes en su Discurso del Método y como presume el modelo Cascada de Royce (Royce, 1970). Realmente la Cascada es una traducción casi literal del método cartesiano, tres siglos después.

El Método era la clave para llegar a La Verdad. Bacon escribió: "Es tal nuestro método de descubrimientos científicos, que no deja gran cosa a la penetración y al vigor de las inteligencias, antes bien las hace a todas aproximadamente iguales." (Bacon, 1620). En su momento, esta idea fue un paso adelante (democrático) porque se oponía al principio de autoridad medieval (que unos fuesen superiores a otros), pero también tenía un severo efecto negativo porque anulaba la libertad creativa. Si El Método lo resolvía todo no era necesaria la libertad creativa. Bacon, en la citada expresión, lo deja claro: El Método estaba por encima de la inteligencia humana y esta idea ha perdurado por siglos.

La IS también sublimó el valor de El Método y supuso que cualquier desarrollador podría realizar con éxito cualquier software solo aplicando El

Método de forma adecuada. DeMarco relata: "They {companies} were stuck on the method because it gave a comforting sense of completeness; it appeared to them to be The Answer to all their problems. When it didn't solve their problems they blamed themselves and tried harder." (DeMarco & Lister, 1987). La culpa es de los humanos porque aplican mal El Método, la culpa no es de El Método.

Resumiendo, durante mucho tiempo la IS ha percibido su universo de forma simple, compuesto por dos dimensiones claramente independientes entre sí: El Método por un lado y las tecnologías concretas de desarrollo de software por el otro. Los libros de texto lo reflejan de este modo, por ejemplo (Pressman, 2001). Pero tan solo era una ilusión, parafraseando a Einstein.

Hoy la percepción es diferente. Hace algunas décadas la ciencia está cambiando de paradigma en el sentido kuhniano (Prigogine, 1984). Se está consolidando una cosmovisión totalmente distinta a la cosmovisión de la ciencia del siglo XVII. Hoy se aprecia un universo no determinista que considera la incertidumbre como parte de la esencia y por tanto, ineludible (Klir, 1988). Consecuentemente, la infalibilidad de El Método (cualquiera que sea) se desmorona porque no funciona en presencia de incertidumbre. De manera que los factores humanos como la inteligencia, la libertad, la creatividad, que habían sido relegados, ahora han pasado a ser las claves de éxito.

Hoy el factor humano se manifiesta como una dimensión decisiva del universo de la IS (Capretz, 2014; Capretz & Ahmed, 2018), aunque ha tardado tres décadas desde que fue advertido en el libro Peopleware (DeMarco & Lister, 1987).

El salto cualitativo se produjo alrededor del Manifiesto Ágil, es decir la axiomática de la actual corriente de desarrollo de software, que se decanta por la dimensión humana. Sus autores se posicionan firmemente del lado de la inteligencia y la libertad diluyendo el valor de El Método. La corriente Ágil

requiere disciplina (no obediencia), autoorganización, y sobre todo requiere actitud. De lo contrario no se satisfacen ni su Manifiesto, ni sus Principios. Debajo del modelo Ágil está la libertad individual, la libertad colectiva, los conflictos entre ambas y, el desconcierto y miedo que produce la libertad (Fromm, 1994).

Como apunta (Appelo, 2011) la IS se está reorientando hacia el paradigma científico actual. Posiblemente el primer hito de cambio fue el modelo Espiral (Boehm, 1988) que reconoció la presencia de incertidumbre, en su propuesta para manejar riesgo renunciando a la Cascada, a finales de los noventa.

El universo de la IS ha dejado de ser aquel espacio determinista de dos dimensiones independientes. El reconocimiento de la presencia ineludible de incertidumbre y la necesidad de enfrentar esa incertidumbre ha obligado a considerar dos aspectos: que hay una dimensión humana clave, y además que el espacio de trabajo es holístico (Klir, 1988). Por tanto, la triada humano, metodología y tecnología está imbricada, no se pueden separar sus componentes.

Blum (Blum, 1992) advirtió el carácter holístico de la IS hace más de dos décadas pero tampoco fue oído. Maslow ya había escrito "[…]cuesta mucho implementar y utilizar la visión holística como una forma de ver el mundo." (Maslow, 1987).

La docencia de la IS también ha comenzado a despertar. Por ejemplo, Capretz et al expresan: "Students need to repeatedly see how software engineering is not just about technology." (Capretz, 2014; Capretz & Ahmed, 2018). Donde "is not just about technology" significa los aspectos humanos de la IS agrupados en la expresión soft skills como contraste de la expresión hard skills relacionada con la tecnología. Esta necesidad ha sido también puesta de manifiesto por una revisión de literatura reciente que considera que los soft skills son tan importantes como los hard skills, si no más (Assyne et al., 2022; Cico et al.,

2021; Ellis et al., 2019; Garousi et al., 2019, 2020; Liargkovas et al., 2021; Licorish et al., 2022; Oguz & Oguz, 2019) La importancia de incluir los aspectos humanos de la Ingeniería del Software in university curricula fue ya advertida hace más de 10 años (Hazzan, 2010).

El objetivo del enfoque llevado a cabo en este trabajo es promover el desarrollo de competencias de los estudiantes para manejar los aspectos humanos, individuales y sociales, enfrentados a la incertidumbre y a la holística asociada.

Poner todo esto en juego es un reto docente. Capretz et al lo confirman: "Educators willing to venture into this area face an arduous task if they try to convince their colleagues and software engineering zealots of the importance of soft skills materials." (Capretz, 2014; Capretz & Ahmed, 2018). Es difícil convencer de algo a quienes han relegado ese algo toda su vida. Pero este es uno de los problemas frecuentes cuando se cambia de paradigma, como sucede actualmente con la IS y su docencia.

Pocos cursos de ingeniería del software abordan la dimensión humana, la naturaleza holística de la disciplina y la colaboración a gran escala. En 2010 se publicó un nuevo plan de estudios en el que se introdujo el concepto de Software Factory (Tvedt et al., 2010), pero no existen estudios sobre los resultados de su implementación. Broman et al. (Broman et al., 2012) propusieron un enfoque alternativo a los proyectos finales tradicionales que se realizan al término de los estudios de informática. El curso contaba con aproximadamente 120 estudiantes divididos en cuatro empresas de software distintas, pero el estudio no profundiza en la adquisición de soft skills. Existen otros trabajos que tratan las soft skills, aunque en ellos la colaboración se limita a grupos pequeños de estudiantes. Paasivaara et al. (Paasivaara et al., 2018) midieron los cambios en las actitudes hacia la dificultad e importancia de ciertos aspectos presentes en el desarrollo de software, incluidas las soft y hard skills. En ese caso, los equipos estaban formados por 8 estudiantes. Raibulet et al.

(Raibulet & Arcelli Fontana, 2018) describieron un estudio de caso para recoger la percepción del estudiantado sobre el uso de mecanismos y herramientas que apoyan dos soft skills importantes: la colaboración y la comunicación. En este curso, los equipos estaban compuestos por 3 a 5 estudiantes. Además, hasta donde sabemos, en estos cursos todos los miembros de un equipo reciben la misma calificación.

La originalidad de nuestro estudio radica tanto en el enfoque diseñado como en la investigación educativa realizada. Nuestro enfoque es inmersivo y resalta la colaboración e incorpora todas las dimensiones de la ingeniería del software (humana, técnica y metodológica). La colaboración se potencia porque, en lugar de trabajar con pequeños equipos de proyecto, los 39 estudiantes del curso se organizan como una empresa simulada que desarrolla un único proyecto. Asimismo, hemos diseñado un sistema de evaluación que orienta al alumnado teniendo en cuenta tanto el trabajo en equipo como el trabajo individual, y que permite una evaluación externa (cliente o stakeholder) y una evaluación interna (responsables de la empresa). Otro aspecto por considerar es que nuestro enfoque implica trabajar con un cliente real, dado que la participación efectiva de agentes industriales en los cursos de ingeniería del software sigue siendo limitada (Cico et al., 2021). La investigación educativa realizada también es novedosa, ya que considera la experiencia del estudiantado de manera holística sin partir de variables ni hipótesis predefinidas.

El objetivo de este estudio es describir nuestra experiencia con el enfoque de inmersión mediante un estudio de caso cualitativo que contempla tanto el proceso de aprendizaje como sus resultados. Este estudio cierra un ciclo de tres años consecutivos aplicando la inmersión como enfoque docente.

2. La asignatura Ingeniería de Software

Las decisiones de diseño de la asignatura se han realizado como consecuencia de un proceso de mejora, teniendo en cuenta los resultados obtenidos en cursos anteriores. En este apartado se realiza una descripción del curso con una perspectiva longitudinal, desde el inicio de esta asignatura.

En primer lugar, se describen las competencias que se espera que desarrollen los estudiantes durante el curso. En segundo lugar, se explica la evolución de la asignatura de IS desde sus inicios hasta el diseño del enfoque inmersivo que se presenta en este trabajo.

2.1 Competencias

Las competencias de IS que se esperan de un grado en informática están dirigidas al desempeño de trabajo en el contexto de la industria de desarrollo de software. Por tanto, son competencias de iniciación o de alcance medio. No se pretende madurez. La profesionalización en IS es tarea de los másteres en esta especialidad y de los años de experiencia.

El presente trabajo se acota a la asignatura de IS del Grado de Matemática e Informática, que es una titulación de informática acreditada por EuroInfo. La asignatura se da en el sexto semestre en un grado de ocho semestres. Es una asignatura de seis créditos ECTS (European Credit Transfer and Accumulation System). Cada crédito implica 27 horas de trabajo de los estudiantes, distribuidas en dieciséis semanas que son alrededor de diez horas semanales incluidas cuatro horas de trabajo en el aula.

La asignatura aborda competencias técnicas, metodológicas y soft skills. Los dos primeros tipos de competencias tratan temas de la IS tradicional, por ejemplo, requisitos, diseño, ciclos de vida, gestión. Mientras que el tercer tipo se dirige al desempeño de habilidades fundamentales para la cosmovisión actual, es decir la corriente Ágil. Por ejemplo: trabajo individual, trabajo en equipo y

con equipos, aprendizaje autónomo, trabajo en situaciones pobres de información y bajo presión. Pero, sobre todo, confianza en sí mismos. "La aptitud más importante es hacer frente a lo inesperado" (Papert, 1993).

Las competencias técnicas y metodológicas fueron el centro de atención de la IS durante años. Por esta causa eran el centro de atención de la asignatura y también, porque se pueden abordar desde la pizarra, los libros y de forma individual. Sin embargo, a medida que la IS se adentraba en la corriente Ágil, creció la importancia de la dimensión humana. La metodología se hizo más colaborativa, más experiencial, con menos pizarra y menos libros. Las soft skills se hicieron más indispensables.

Las competencias permanecieron iguales en el documento oficial de la asignatura, pero progresivamente se variaron los pesos. La dimensión humana, muy entrelazada con la metodológica, pasó al primer plano.

A continuación, se realiza una descripción longitudinal de la asignatura Ingeniería de Software desde sus inicios (Tabla 1).

2.2 Primeros cursos

Desde su comienzo la asignatura de IS del Grado de Matemáticas e Informática fue favorable a la innovación docente porque no tenía compromisos con la ortodoxia y además, porque tenía pocos estudiantes. Al inicio, solo hubo doce estudiantes, después la cifra ha ido creciendo. Los primeros cursos fueron prácticos, casi ausentes de lecciones magistrales. Se desarrollaban, por equipos, proyectos pequeños de carácter académico y del dominio técnico de los estudiantes. Cada equipo tenía su propio proyecto que realizaba de forma evolutiva con libertad de organización y creación. Cíclicamente se hacían discusiones sobre los proyectos en desarrollo. El profesor asesoraba, inducía reflexiones, evaluaba y calificaba a los equipos. Todos los miembros del equipo recibían la misma nota para evadir la dificultad de evaluar al individuo en una

acción colectiva. Se supuso que las soft skills se adquirían implícitamente con la práctica, pero no había un mecanismo para comprobarlo. La asignatura conservó este diseño durante los tres primeros cursos y se podría clasificar como una asignatura práctica relativamente convencional.

Tabla 1. Descripción longitudinal de la asignatura Ingeniería de Sofware

		Primeros cursos	**Primer hito**	**Primera inmersión**	**Segunda inmersión**	**Tercera inmersión**
Número de estudiantes		Menos de 20	27	32	37	39
Stakeholder externo			X	X	X	X
Desafío técnico			X	X	X	X
Trabajo en equipo	**Equipos trabajando de forma independiente**	X				
	Equipos trabajando de forma competitiva		X			
	Equipos trabajando de forma Colaborativa			X	X	X
Modulación de la libertad	**Evaluación sistemática comunicada desde el inicio del curso**				X	X

	Liderazgo obligatorio para todos los miembros de cada equipo				X	X
	Los líderes externos actuan como expertos				X	
	Los líderes externos actúan como jefes de proyecto					X

2.3 Primer hito: el stakeholder, el desafío técnico y la competencia.

En el cuarto año de impartición de la asignatura se dio un vuelco a la asignatura introduciendo tres factores relevantes: 1) un stakeholder externo que solicitaba un proyecto y lo seguía hasta el final. 2) un desafío técnico para los estudiantes porque el proyecto sobrepasaba las fronteras técnicas de las asignaturas previas de la carrera y se dejó que los estudiantes resolvieran por sí solos este desafío. 3) la competencia entre los equipos porque todos los equipos realizaban el mismo proyecto y competían por el premio ofrecido por el stakeholder. Aprovechando la presencia del stakeholder se simularon relaciones "económicas" de los estudiantes con la "empresa cliente" mediante la participación del stakeholder en la evaluación de los estudiantes. Esto redujo el carácter coercitivo del profesor y aumentó su rol de observador. Igual que antes, la calificación de cada estudiante era la misma que la puntuación del grupo al que pertenecía. Participaron veinte y siete estudiantes.

Resultados del hito. El stakeholder externo y la complejidad técnica del proyecto, constituyeron un elevado reto y riesgo para la asignatura. El stakeholder aportó una inestimable aproximación a la realidad actual de las relaciones entre los desarrolladores de software y quienes solicitan ese desarrollo. En otras palabras, el stakeholder aportó una experiencia vívida del complejo tema de requisitos. Por otra parte, el desafío técnico demostró, sobre todo a los estudiantes, su potencial para adentrarse en el mundo real de la producción de software. Al final, el reto y el riesgo fueron vencidos porque los resultados del proyecto fueron aceptables y los estudiantes estuvieron satisfechos. Pero la competencia entre equipos produjo efectos desagradables de manera que el profesor decidió cambiar las tornas y convertir la colaboración en el objetivo central desde la perspectiva humana y la técnica. Primero, todos los estudiantes debían colaborar en un mismo proyecto que abarcase a la clase completa. Segundo, el aumento del tamaño del proyecto que aumenta la colaboración interna del software por tener más módulos debía justamente resaltar la conveniencia de reducir las dependencias entre los módulos del software mediante el diseño. Resumiendo, de estas ideas había nacido la inmersión que se presenta en este trabajo.

2.4 Primera inmersión.

En el siguiente curso se concretó la inmersión. Se simuló que la clase era una empresa de software encargada de realizar un proyecto real y grande. En apariencia solo se trataba de extrapolar la experiencia de trabajar en pequeños grupos independientes a trabajar en un grupo grande dividido en pequeños grupos dependientes entre sí. El proyecto software trataba un tema interesante (recepción y distribución de paquetes) que involucraba técnicas nuevas para los estudiantes como aplicaciones móviles y desarrollos web. El stakeholder externo (una persona que trabajaba en una empresa de desarrollo software)

propuso el proyecto y lo siguió hasta el final, como se hizo antes.

En este curso, además de explorar la inmersión se quería explorar la libertad y su gestión sobre todo cuando la colaboración pone en conflicto los intereses individuales y los intereses colectivos. De manera que se les otorgó amplia libertad a los estudiantes (treinta y dos) para que se organizaran, coordinaran, y entre todos realizaran el proyecto. Resumiendo, había dos macro variables simultáneas, era una inmersión (1) en un contexto de libertad (2). Otra vez la asignatura asumió un gran riesgo, pero esta vez respecto al impacto de la libertad. Nuestro objetivo fue conocer qué sucede en la clase cuando se dispone de la libertad de colaborar para alcanzar un fin común complejo y de gran tamaño. Había dos razones para afrontar este riesgo. Una razón era la convicción de los autores sobre el valor universal de la libertad. "Como el hueso al cuerpo humano, y el eje a una rueda, y el ala a un pájaro, y el aire al ala, así es la libertad la esencia de la vida. Cuanto sin ella se hace es imperfecto, mientras en mayor grado se la goce, con más flor y más fruto se vive. Es la condición ineludible de toda obra útil." (Martí, 1977). La otra razón para realzar la libertad era metodológica y se acota al campo de la ingeniería de software: "The best architectures, requirements, and designs emerge from self-organizing teams." (Beck et al., 2001).

Respecto a las soft skills había varios objetivos docentes: 1) extender la competencia de trabajar en equipo a la competencia de trabajar con varios equipos; 2) reforzar la competencia de trabajar con poca información a causa de las indefiniciones asociadas con la magnitud del proyecto; 3) estimular las competencias de autoaprendizaje y confianza en sí mismos porque los estudiantes deben desarrollar un producto software complejo sin la participación de docentes.

Como en cursos anteriores la calificación se asoció implícitamente con el

proyecto, pero no se definió cómo, ni se usaron exámenes; se vería al final de la asignatura. El rol de observador del profesor se acentuó al aumentar la libertad global de la clase. Este rol es difícil de llevar porque habitualmente los profesores desempeñan roles muy activos en las clases.

Resultados de la primera inmersión. El proyecto alcanzó resultados mínimamente aceptables por partes separadas, pero no se logró la interconexión entre las partes. Respecto a la participación, numerosos estudiantes trabajaron poco o nada; algunos por dejadez y otros porque no les dejaron. La proporción general podría ser de un tercio de los estudiantes que trabajaron, otro tercio de estudiantes con poco trabajo y un tercio de estudiantes que no trabajaron. En fin, se apreció un nivel medio bajo de compromiso con el proyecto. Solo en el equipo más pequeño hubo cohesión alrededor de un líder dominante que conocía la tarea asignada, pero posiblemente dos de los seis miembros de ese grupo hicieron la mayor parte del trabajo. En el resto de los grupos hubo desorganización y carencia de liderazgo, nadie se quería responsabilizar. En uno de los grupos se produjeron serios enfrentamientos personales entre unos pocos miembros mientras los otros mantenían una actitud pasiva.

La libertad exige ponerse de acuerdo en el caso del trabajo colaborativo. Cuando el proyecto es pequeño, también es pequeña la cantidad de personas que deben ponerse de acuerdo. En este ensayo de inmersión, el tamaño del proyecto exigía el acuerdo de la totalidad de la clase. Los estudiantes se autoorganizaron según sus afinidades creando grupos de diferentes tamaños (6,8,8,10) y, se repartieron a sí mismos las tareas de los grupos y de los miembros de cada grupo. Pero era una estructura endeble por el bajo nivel de compromiso. Posiblemente en virtud de la libertad, los estudiantes siguieron la pauta habitual del trabajo en equipo en sus estudios universitarios, donde hay diversos objetivos e intereses y poco compromiso.

Los hábitos académicos, el tiempo disponible y algún factor más estuvieron en contra de lograr el acople adecuado. La libertad, quizás excesiva para las condiciones del curso, produjo un poco de movimiento browniano, que fue motivo de frustración sobre todo para los estudiantes. No obstante, el profesor mantuvo el nivel de libertad hasta el final. Los problemas de colaboración humana repercutieron en las interconexiones (colaboración) de las partes del software que fueron prácticamente inexistentes.

La calificación obligatoria de la asignatura, que impone el premio y la penalización, también se dejó en manos de los estudiantes, pero se individualizó. Estuvo basada en las autoevaluaciones personales y en las opiniones de los líderes de grupos; muchas de ellas sobrevaloradas, proteccionistas y a la defensiva. Esto provocó abundantes calificaciones elevadas no ajustadas al desempeño. Como consecuencia, se produjo un malestar en los estudiantes porque las calificaciones no discriminaban el que había trabajado más del que había trabajado menos o nada. Sorprende este malestar porque los estudiantes fueron las fuentes de información para esas calificaciones.

La inmersión no produjo el esperado efecto de colaboración de todos con todos, ni tampoco produjo la atención que se esperaba sobre la conveniencia de reducir las dependencias entre los módulos del sistema. A pesar de todo, el balance fue positivo a juicio del profesor por diversas razones. 1) Los estudiantes habían vivido una experiencia de aproximación a la IS donde se enfrentaron a condiciones de incertidumbre respecto al producto y a las técnicas para desarrollar el producto. 2) En estas condiciones los estudiantes fueron capaces de obtener resultados aceptables por sí solos, sin la dirección del profesor. Por tanto, consolidaron la confianza en sí mismos. 3) Los estudiantes experimentaron la influencia de los aspectos humanos del desarrollo de software. Se había dado un paso adelante en la aproximación a la IS actual.

Varios estudiantes confirmaron la utilidad de la asignatura, cuando comenzaron a trabajar en empresas. Habían aprendido, pero no sabían qué habían aprendido exactamente hasta que se enfrentaron a situaciones laborales reales. Fue un primer ensayo que valió la pena y convenía repetirlo, pero reajustando la libertad.

2.5. Segunda inmersión.

La segunda inmersión se realizó el curso siguiente con un total de 37 estudiantes y su diseño reconsideró dos ejes de la primera inmersión: la libertad y la organización, ambos entrelazados.

La libertad. La libertad y el tamaño del proyecto se mostraron contradictorios, dadas las condiciones de los cursos. Entonces, con la finalidad de mantener el tamaño del proyecto se redistribuyó la libertad para reducir el movimiento browniano. Esta imposición solo restringió la libertad de los estudiantes para la autoorganización de la "empresa", pero dejó libertad para las actividades de los equipos, incluyendo las decisiones y las imprescindibles coordinaciones entre ellos.

Una medida fue establecer obligatorio el rol de líder de equipo y hacerlo rotativo porque forma parte de las competencias de aprendizaje, al menos, de iniciación en este desempeño. Los equipos fueron formados por el profesor siguiendo el criterio de balancear personalidades utilizando un test simple de personalidad (John Wiley & Sons, 2022). También se introdujeron dos estudiantes de niveles superiores, que habían cursado la asignatura en el curso anterior, como apoyo técnico y organizativo con capacidad de participar en la evaluación. Además, del aprendizaje de estos estudiantes también aprende la asignatura. Los estudiantes avanzan notablemente en su formación de IS apoyando el desarrollo del producto software. El resultado formal son Trabajos de Fin de Grado o de

Máster según sea el nivel que cursen los estudiantes. Del otro lado, la asignatura se enriquece de forma significativa con las ideas y acciones que aportan estos estudiantes a partir de sus vivencias previas en la asignatura. Tanto los líderes de los equipos como los estudiantes externos desempeñan roles destinados a priorizar el proyecto y facilitan hacer parcelas de libertad que reducen (dividen) el problema de ponerse de acuerdo. La libertad queda redistribuida y desconcierta menos.

Otra de las medidas del reajuste fue restablecer y mejorar la eficacia de las relaciones entre el stakeholder externo y la "empresa" de la clase, mediante un control periódico que facilitase la retroalimentación temprana. Se diseñó un sistema de evaluación coincidente con los ciclos de evaluación del proyecto por el cliente externo.

La organización. El ecosistema. Aunque fue invisible en la primera inmersión, el proyecto, los estudiantes y el stakeholder formaban un ecosistema como también se forma un ecosistema particular cada vez que se desarrolla un proyecto en el universo software real. La visión de ecosistema es una forma de interpretar la organización del de los organismos (sistemas) y sus relaciones como un todo vivo. Las soft skills son el pilar de estas relaciones. En nuestro caso docente se podría decir que la inmersión se produce dentro de un simulado ecosistema empresarial.

La evaluación, omitida en la primera inmersión, se incluyó en el ecosistema de la segunda inmersión que quedó formado por:

- El sistema software que se desarrolla (el Proyecto).
- El sistema de los desarrolladores (los estudiantes).
- El sistema cliente (cuya interfaz es el stakeholder externo).
- El sistema de evaluación (retroalimentación).

Ecosistema. El sistema software que se desarrolla (el Proyecto)

El proyecto es la razón de ser del ecosistema en el universo software real y el eje del aprendizaje en la inmersión. Por tanto, se ha procurado que el proyecto tenga dos cualidades. Primero, que sea especialmente atractivo, que provoque curiosidad y reto. El proyecto de la segunda inmersión, y también el proyecto de la primera, han sido temas de interés empresarial que involucran técnicas nuevas para los estudiantes de la asignatura, por ejemplo: desarrollo web, programación en la nube, programación responsive y aplicaciones móviles. Segundo, se ha procurado que el proyecto estimule el desarrollo de las competencias propuestas sobre todo haciendo que abarque a toda la clase tal que se resalten las relaciones en el ecosistema. En la dimensión humana, ha importado desarrollar la colaboración como factor de éxito. Mientras que en la dimensión técnica, ha importado que se aprenda a reducir la dependencia entre las partes software como otro factor de éxito. Concretamente, que se estimule el uso del Principio de Ocultación de información (Parnas, 1972) enfrentando a los estudiantes con un proyecto de muchas interconexiones. El proyecto de la segunda inmersión se ocupó de desarrollar un sistema integral de gestión de trenes para sustituir a otro sistema concebido en términos de silos.

Ecosistema. El sistema de los desarrolladores (los estudiantes).

Los estudiantes forman el sistema que desarrolla el sistema software (proyecto). La interacción es en ambos sentidos. Los estudiantes construyen el software y el software les "exige" soluciones. Este conjunto de sistemas interactúa a su vez con el sistema (empresa) "cliente" mediante su interfaz, el stakeholder.

La interacción entre ambas "empresas" es en los dos sentidos. El stakeholder externo "recibe" el producto software y "remunera" con puntos de la evaluación de los estudiantes. El criterio de evaluación del producto se resume en una condición satisfaciente (satisfactorio y suficiente). El objetivo principal

de esta interacción es promover una estrategia de sentido común a menudo olvidada: "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software." (Beck et al., 2001).

El aula es la "sede" de la "empresa" de desarrollo. Se utiliza para reunirse, analizar, trabajar y confrontar con el cliente. En ella se invierte alrededor de un cuarenta por ciento del tiempo (cuatro horas semanales) asignado a la asignatura por sus créditos ECTS (seis). La distribución del resto del tiempo corre por parte de los estudiantes. Esto último es una importante fuente de problemas en la academia porque los estudiantes deben atender varias asignaturas al unísono, y cada estudiante lo hace con sus costumbres e intereses.

Ecosistema. El sistema "cliente".

Del sistema cliente solo se percibe una interfaz: el stakeholder externo. Simula que representa a su empresa solicitando un proyecto software a la "empresa" formada por los estudiantes. El objetivo es establecer una supuesta relación empresa-empresa. Desde el primer hito, las personas que han actuado como clientes externos han tenido más de diez años de experiencia en el desarrollo de software.

El proyecto propuesto por el stakeholder externo está relacionado con su empresa (es un proyecto real) pero no responde a una solicitud de la empresa. De hecho, el producto software resultante no se entrega al stakeholder. El vínculo concreto de la empresa con los estudiantes facilita la posible incorporación de estudiantes a sus filas, en prácticas o ya graduados.

El rol del stakeholder externo es importante porque aporta una de las relaciones humanas esenciales en el desarrollo de software, sobre todo en el enfoque Ágil, y además otorga realismo al ecosistema. Es una persona ajena a la academia que representa a una empresa: propone. negocia y evalúa iterativamente el proyecto que desarrolla el sistema de los estudiantes. El stakeholder propone el proyecto

software de forma verbal, poco definido, cambiante, extenso en longitud y tiempo, sobre temas que sobrepasan el currículo y dirigido al conjunto de la clase. Es decir, propone un proyecto de gran tamaño como lo haría un cliente real. Este enfoque contrasta con la presentación de proyectos o ejercicios académicos donde los enunciados son escritos, definidos, estables y breves, sobre temas acotados en el currículo, dirigidos a un estudiante o a un grupo pequeño de estudiantes.

Ecosistema. El sistema de evaluación.

En la primera inmersión el modo de calificación de la asignatura se dejó abierto hasta el final para reducir su efecto de recompensa-coerción y ampliar el margen de libertad durante el curso. En la segunda inmersión se decidió crear un sistema de evaluación formativa numérica que apoyara la focalización de los objetivos y sirviese de retroalimentación. El mayor peso (60%) se destinó al trabajo colectivo. La evaluación se hizo cíclica, coincidiendo con los fines de ciclos o sprints, y consideró diversas perspectivas. Dado el corto período de los ciclos (tres semanas), la evaluación tiene un marcado carácter continuo, próximo a la visión Ágil y además, estimulante para el desarrollo de la competencia de trabajo bajo presión por las entregas al stakeholder. Las perspectivas de la evaluación son cuatro que se entremezclan: el desempeño colectivo, el desempeño individual, la visión externa (del stakeholder) y la visión desde el interior de la clase (Tabla 2). La evaluación es un sistema que se encuentra en un plano distinto del resto, pero cuya proyección incide sobre todo el ecosistema. La evaluación final es el promedio de las evaluaciones en cada ciclo. La calificación de la asignatura corresponde al profesor que considera la evaluación final de cada estudiante y otorga una nota según su apreciación.

Tabla 2. Esquema del sistema de evaluación

Tipo	Componente	Peso	Evaluador	Visión
Desempeño colectivo	**Evaluación de la Compañía por del cumplimiento global de los requisitos**	**20%**	**Stakeholder**	**externa**
	Evaluación del equipo por el cumplimiento de los requisitos	**20%**	**Stakeholder**	
	Evaluación del equipo por su desempeño interno	**20%**	**Líderes de sección**	**interna**
Desempeño individual	**Evaluación del desempeño individual**	**40%**	**Todos los líderes**	

Los evaluadores son los siguientes:

- El stakeholder evalúa el desarrollo del proyecto en su conjunto con un criterio satisfaciente (satisfactorio y suficiente) y también evalúa el conjunto de requisitos elaborado por cada equipo. La puntuación de la compañía se aplica a toda la clase y supone un 20% de la puntuación invididual. La puntuación del equipo se aplica a cada equipo y supone un 20% de la puntuación invididual.
- Los líderes de sección evalúan el desempeño integral de cada equipo (resultados, diseño, pruebas, productividad, coordinación, etc.). La puntuación se aplica a todo el equipo. Aporta el 20% de la evaluación individual.
- Los líderes de equipos evalúan el desempeño y actitud de cada miembro del equipo. Aporta el 40% de la evaluación. La evaluación propia y la de sus compañeros es un aspecto difícil del liderazgo. El desempeño y actitud de cada líder de equipo es evaluado por los lideres externos con el mismo peso de 40%.

El diseño de la evaluación consideró varios aspectos.

- Realzar el trabajo colectivo. El 60% de la evaluación de cada estudiante depende del rendimiento del equipo y de la compañía.
- Que se sienta la visión externa como una "relación económica" donde el stakeholder "recibe" y evalúa técnicamente el proyecto otorgando una puntuación a la totalidad de la clase (empresa).
- La "relación económica" simulada obliga, salvo excepciones, a que las evaluaciones interiores concuerden aproximadamente con la exterior del stakeholder para mantener un balance entre lo que "cobran" los desarrolladores y lo que recibe la empresa por los resultados del proyecto.
- La tendencia a sobrevalorar el trabajo realizado muy por encima del resultado (producto) se debe reducir a través de negociación.
- Distinguir el resultado técnico (requisitos cumplidos) del desempeño (individual y del equipo) porque expresan cualidades diferentes del trabajo.
- Disponer de evaluaciones interiores para la retroalimentación y corrección de rumbo. También son una herramienta de desarrollo de competencias de liderazgo, puesto que los líderes son los responsables de estas evaluaciones y de provocar las reflexiones sobre sus resultados.

Resultados de la segunda inmersión. El curso comenzó de forma similar a la primera inmersión. Sin embargo, al final del primer ciclo se produjo una importante protesta generalizada. Según los estudiantes, protestaron porque se sentían muy agobiados por la presión de la asignatura. En la primera inmersión no hubo protestas, pero tampoco hubo presión de la asignatura dada las condiciones de libertad: la evaluación se dejó abierta hasta terminar la asignatura. Pero en la segunda inmersión el sistema de evaluación ejerció presión desde el primer día. En el segundo ciclo se consiguieron aproximadamente los resultados del proyecto correspondientes al primer ciclo

y se tranquilizó la situación.

Al final del curso (último ciclo) el grado de desarrollo del proyecto superó notablemente los resultados de la primera inmersión, porque se consiguieron ofrecer imágenes parciales del sistema interconectado, aunque no se logró la imagen integral. Ni tampoco se logró que las interconexiones entre los módulos del sistema aplicasen el principio de Ocultación de Información. Este Principio ha tenido una historia difícil: ha sido rechazado, confundido e ignorado hasta nuestros días (Berard, 1993; Parnas, 2002). La segunda inmersión confirma que se mantiene la dificultad del Principio. Fracasó el intento por parte del profesor de que algunos estudiantes reinventaran algo similar al Principio inducidos por el trabajo con el proyecto.

Además de los resultados técnicos, en la segunda inmersión aumentó el compromiso de los estudiantes a un nivel medio. Fue un paso adelante en la dirección de transformar el modo de trabajo cooperativo al modo de trabajo colaborativo en el sentido de cohesionar objetivos e intereses. El trabajo en equipo es un pilar del enfoque Ágil. La comunicación humana entre los equipos de trabajo fue alta pero poco efectiva porque lastró los resultados del proyecto y además, la comunicación fue ineficiente porque dilapidó esfuerzos debido a su volubilidad. Se aspiraba a conseguir más disciplina de la alcanzada.

Se avanzó en el aprendizaje de la conceptualización y gestión de los requisitos, incluso en la negociación con el stakeholder. También mejoraron los siguientes aspectos: la distribución del trabajo fue más uniforme; hubo más estudiantes con experiencia de trabajo como líderes; el control del trabajo individual fue más preciso y la calificación fue más justa.

En la segunda inmersión se elevó la proporción de estudiantes involucrados con el proyecto. Por tanto, el desarrollo potencial de las competencias alcanzó a más estudiantes. Algunas competencias adquirieron mayor nivel de intensidad, por ejemplo, la confianza en sí mismos porque los resultados técnicos del

proyecto fueron mejores y también la competencia de trabajo bajo presión debido a la acción del sistema de evaluación. Sin embargo, la competencia de diseño de software no alcanzó el nivel de los cursos dados sin inmersión, donde era uno de los objetivos centrales. En su lugar se desarrollaron competencias de programación en las áreas desconocidas por los estudiantes, pero imprescindibles para la realización del proyecto. Metafóricamente hablando, han estado aprendiendo a levantar una catedral sin dibujarla, ni calcularla previamente, tal y como se construyeron muchas catedrales.

La segunda inmersión superó de manera notable los resultados de la primera inmersión. Valió la pena el segundo intento, aunque el coste de esfuerzo y tensión fue alto. No es posible asegurar que los resultados se deben a los cambios, pero se puede hacer la conjetura que los cambios influyeron positivamente en la mejora de los resultados. Fueron cambios severos sobre la libertad: la introducción de la evaluación sistemática, el liderazgo obligatorio en los equipos, la inserción de líderes externos a nivel de "empresa" para apoyar la coordinación y el trabajo de los equipos.

2.6. Tercera inmersión

En este apartado se describe el curso cuyos resultados se analizan con detalle en este artículo.

La tercera inmersión aumentó la cantidad de estudiantes (39 estudiantes) y se desarrolló un sistema para la gestión de la parte productiva de una empresa cervecera que consideraba desde los proveedores hasta los puntos de venta. El proyecto incluyó, una vez más, temas no contemplados en el currículo de los estudiantes: Web, IoT y Blockchain. Esta inmersión mantuvo las mismas condiciones de la segunda inmersión, excepto en dos aspectos. El primer aspecto fue redistribuir la libertad aumentando la autoridad del rol de los líderes externos: de asesores pasaron a ser jefes y esto varió el ecosistema haciéndolo

un poco más jerárquico. El segundo aspecto que se modificó fue la planificación del curso, que dedicó el primer mes a cortos seminarios para tratar de reducir el gap técnico de los estudiantes. Supuestamente este gap era el motivo de la protesta en la segunda inmersión.

3. Metodología

El propósito de este trabajo es compartir nuestra experiencia sobre la puesta en marcha del enfoque de inmersión con la comunidad educativa de IS. De forma coherente con este propósito, hemos llevado a cabo un estudio de caso cualitativo porque es una metodología de investigación adecuada para analizar una experiencia en particular, obtener una comprensión profunda de la misma y compartir sus resultados (Merrian, 1998).

A continuación, se describe la metodología usada en este estudio siguiendo los APA Style JARS (Journal Article Reporting Standards) para estudios cualitativos.

Diseño de la investigación

Se presenta un estudio de caso de la asignatura IS impartida con el enfoque descrito en el apartado anterior como tercera inmersión. El número de estudiantes del curso fue 39 y los estudiantes que participaron en el estudio fueron 37.

Las preguntas de investigación que guiaron este estudio son las siguientes:

- ¿Cómo se ha desarrollado este enfoque de aprendizaje a lo largo del semestre?
- ¿Qué resultados se han obtenido con este enfoque de aprendizaje al final del semestre?

La primera pregunta hace referencia al proceso educativo y la segunda a sus resultados en el momento de la finalización del curso. Hay que tener en cuenta que este estudio está realizado con una perspectiva de descubrimiento y no tiene como propósito la confirmación de ninguna hipótesis. El objetivo del estudio es descubrir hechos y procesos que probablemente pasarían inadvertidos con otros métodos de investigación menos profundos (Biddle & Anderson, 1986).

Recogida de datos

Para contestar a las preguntas de investigación planteadas hemos usado los siguientes instrumentos de recogida de datos:

- Cuestionario para estudiantes con preguntas abiertas al terminar el primer sprint.
- Cuestionario para estudiantes con preguntas abiertas al final del curso.
- Entrevistas a los estudiantes al terminar el segundo sprint.
- Entrevista al profesor de la asignatura al final del curso.
- Observaciones realizadas durante las clases y registradas en un diario de campo. Las observaciones han sido realizadas por el primer autor del trabajo y por uno de los estudiantes que tuvo el rol de líder de la compañía de desarrollo software.

Los cuestionarios, además de servir como fuente de información para esta investigación, se utilizaron con dos propósitos educativos. El primer propósito fue fomentar la reflexión de los estudiantes sobre su proceso de aprendizaje, y el segundo mejorar futuras ediciones del curso. Estos cuestionarios se pusieron a disposición de los estudiantes en dos momentos distintos del curso y su cumplimentación fue opcional.

- Cuestionario inicial. Fue puesto a disposición de los estudiantes después de la finalización del primer sprint y fue cumplimentado antes de la finalización del segundo sprint. Este cuestionario fue cumplimentado por 7 de los 37

estudiantes.

- Cuestionario final. Fue puesto a disposición de los estudiantes cuando el curso finalizó y las calificaciones estaban ya publicadas. Este cuestionario fue cumplimentado por 20 de los 37 estudiantes.

Entrevistas a estudiantes. Durante la clase dedicada a mostrar los resultados del segundo sprint se realizaron 4 entrevistas a estudiantes con distintos perfiles. Dichos estudiantes fueron seleccionados por el profesor de la asignatura y por los estudiantes que lideraban la compañía. El tiempo medio de las entrevistas fue de 15 minutos. Las entrevistas fueron grabadas en formato audio para su posterior análisis.

Descripción de los investigadores

El equipo de investigación de este trabajo está compuesto por dos investigadores y profesores de IS (segundo y tercer autor) de la universidad en la que se ha realizado el estudio y un investigador (primer autor) y profesor de didáctica STEM en una universidad diferente a la del estudio. El segundo y tercer autor tienen 30 años de experiencia como docentes universitarios y como investigadores. El primer autor tiene 17 años de experiencia en docencia universitaria e investigación.

Participantes

El número de participantes en el estudio fue 37. Han participado en el estudio todos los estudiantes que han firmado el consentimiento informado.

La media de la edad de los participantes del estudio es 21 años. Los estudiantes de esta asignatura han cursado con anterioridad 5 semestres del Grado en Matemáticas e Informática de la Universidad Politécnica de Madrid. La calificación de admisión a este grado en el curso que los estudiantes accedieron

a la universidad fue de 8,517 de los 14 puntos posibles. Esta titulación fue la más demandada por los estudiantes de las titulaciones ofrecidas por la Universidad Politécnica de Madrid sobre Computer Science. Las calificaciones de acceso a otras titulaciones relacionadas en ese mismo momento de acceso fueron 6,906 para Ingeniería Informática y 6,803 para IS.

Relación entre los investigadores y los participantes

El segundo autor del trabajo fue el profesor de la asignatura. El tercer autor del trabajo imparte la asignatura IS en otras titulaciones distintas a la del estudio y no ha estado en contacto con los participantes del estudio. La primera autora del trabajo ha estado en contacto con los estudiantes en momentos puntuales. Fue presentada a los estudiantes el segundo día de clase como una profesora ayudante a cargo de la investigación educativa. Además, asistió a las clases presenciales en varias ocasiones para realizar tareas de observación y para realizar entrevistas.

Análisis

El análisis se ha realizado con la herramienta ATLAS.ti (ATLAS.ti Scientific Software Development GmbH, 2020). En primer lugar, se anonimizaron los datos y se prepararon en un formato apropiado para la importación en ATLAS.ti. Cada participante en el estudio se identificó con un código del tipo E01, E02, E03, etc. Una vez finalizado el proceso de preparación de los datos, se importaron todos los datos del estudio dando lugar a un único documento ATLAS.ti.

El análisis se realizó tomando como guía las dos preguntas de investigación del estudio. La primera tiene como objetivo descubrir los eventos que han tenido lugar durante el curso y que son relevantes desde el punto de vista educativo.

El punto de vista es amplio e incluye tanto los hechos sucedidos como los conocimientos, sentimientos y actitudes de los estudiantes. La segunda pregunta de investigación tiene como propósito capturar una instantánea de los conocimientos, sentimientos y actitudes de los estudiantes inmediatamente después de finalizar el curso.

Para realizar el análisis se han realizado los tres pasos siguientes de forma iterativa.

- Paso 1. El primer autor del artículo analizó todos los datos del curso y seleccionó aquellas unidades de información que pudieran dar respuesta a las preguntas de investigación. La unidad mínima de información que consideró fue la frase. Estos fragmentos de texto seleccionados en esta fase de análisis reciben el nombre de citas en el software ATLAS.ti. Las citas seleccionadas incluyeron una o varias frases, o incluso un párrafo completo. Para cada cita se creo un código que capturara su esencia (Saldaña). Los códigos no estaban definidos de antemano, se crearon de forma específica para cada cita, emergieron de los datos.
- Paso 2. Posteriormente, el primer autor del artículo comparó los códigos obtenidos en el paso anterior y los agrupó en distintas categorías que permitieran dar una respuesta estructurada a las preguntas de investigación planteadas. Las categorías no estaban definidas de antemano, sino que emergieron de los códigos.
- Paso 3. Los tres autores del artículo discutieron los resultados obtenidos hasta el momento y llegaron a un consenso acerca de los mismos.

Una vez realizados estos tres pasos, que son en realidad actos interpretativos, se obtuvo una comprensión mayor de lo sucedido y se volvió al paso 1 para seguir obteniendo más información. En cada iteracción es posible descubrir información nueva, ya que el conocimiento de los investigadores varía con respecto a la iteracción anterior.

En la segunda y sucesivas iteracciones se realizaron las siguientes tareas:

- Se crearon nuevas citas que dieron lugar a nuevos códigos.
- El nuevo conjunto de códigos dio lugar a cambios en las categorías. Las categorías se modificaron, fusionaron, eliminaron o crearon para ofrecer una mejor interpretación de los resultados.

De esta forma, con cada iteracción cambió el conjunto de citas, códigos y categorías. El proceso de análisis terminó cuando una nueva iteracción ya no aportó ninguna información nueva. Se realizaron en total 4 iteracciones.

Integridad metodológica

El objetivo de nuestro estudio no es generalizar los resultados, sino realizar un análisis en profundidad de un contexto concreto con el propósito de aprender de él (Flyvbjerg, 2006). Por ello, utilizamos los criterios de rigor propuestos por Lincoln y Guba (1985): credibilidad, transferibilidad, fiabilidad y confirmabilidad.

Credibilidad

A continuación, se exponen las tres técnicas que hemos utilizado para cumplir con el criterio de credibility:. observación persistente, triangulación y verificación por parte de los participantes.

La técnica de observación persistente proporciona profundidad. Consiste en identificar las características y elementos de la situación que resultaron más relevantes para las preguntas de investigación y centrarse en ellos en detalle. Esta técnica se ha llevado a cabo durante el análisis de datos aplicando los pasos 1, 2 y 3 descritos anteriormente de forma iterativa.

Por otra parte, se han realizado dos tipos de triangulación: de instrumentos de recogida de datos y de investigadores. Es decir, para interpretar los resultados se ha tenido en cuenta la información proporcionada por distintos instrumentos

de recogida de datos. Además, los resultados han sido consensuados por los tres investigadores que han llevado a cabo el estudio, tal y como se ha descrito en el apartado de análisis.

Con respecto a la verificación por parte de los participantes, los resultados finales han sido respaldados por dos de los estudiantes del curso, aquellos que iban a actuar como líderes de la compañía en el próximo curso.

Transferibilidad

Incluimos una descripción detallada y comprehensiva del contexto, los resultados y las contribuciones, con el fin de que los potenciales usuarios del enfoque inmersivo dispongan de elementos suficientes para valorar su transferibilidad.

Confirmabilidad

Los siguientes hechos respaldan estos criterios:

• Los métodos y procedimientos del estudio se describen de manera explícita y detallada.

• Es posible seguir la secuencia de cómo se recopilaron y procesaron los datos hasta llegar a las conclusiones.

• Los resultados se relacionan explícitamente con los datos originales.

4. Resultados

A continuación, se da respuesta a las preguntas de investigación planteadas presentando los aspectos educativos relevantes que han surgido durante el desarrollo del curso y los resultados del mismo. Los siguientes apartados se estructuran de acuerdo con el esquema presentado en la Figura 1.

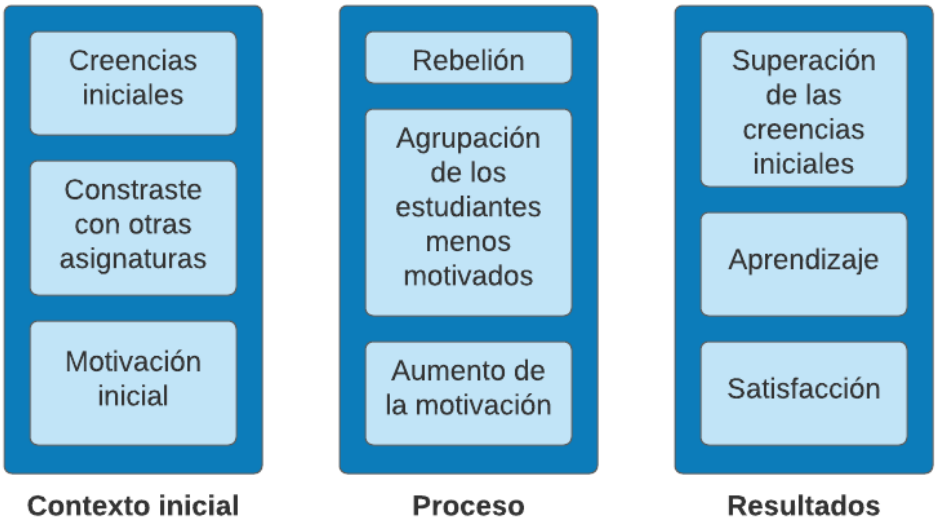

Figura 1. Resultados del estudio de caso

A) Contexto inicial

Con referencia al contexto inicial del curso se describen tres aspectos. El primero son las creencias iniciales de los estudiantes sobre la IS. El segundo es el contraste entre el enfoque adoptado en esta asignatura y el enfoque adoptado en el resto de las asignaturas del semestre. Y el tercero se centra la diferente motivación inicial de los estudiantes hacia la asignatura IS. La tabla 1 muestra una síntesis de este apartado.

Tabla 3. Contexto inicial

Creencias iniciales	• La tecnología es lo más importante en la IS (Se ignora la dimensión humana) • La incertidumbre es erradicable en la IS.
Contraste con otras asignaturas	• Mayor libertad de la habitual. No hay pasos a seguir. • Llevar a cabo un producto software que funcione versus estudiar para superar una prueba de evaluación.
Motivación inicial	• Diferentes niveles de motivación hacia la asignatura (Sprint 1). • Hay estudiantes que perdieron su motivación inicial debido a la falta de implicación de algunos compañeros/as (Sprint 2).

A.1) Creencias iniciales de los estudiantes sobre la Ingeniería del

Software

Durante el análisis realizado descubrimos dos creencias iniciales que pueden dificultar la sintonización de los estudiantes con el enfoque adoptado en la asignatura.

La primera creencia es que la tecnología es lo más importante en la IS. Esta creencia implica que se tiene poca conciencia de la importancia de la dimensión humana de la IS y de la holística entre las dimensiones tecnológica, metodológica y humana.

La segunda creencia es que la incertidumbre es erradicable en la IS, y que por tanto la aplicación de un método conducirá necesariamente a los resultados deseados. Esta creencia está en cierto modo relacionada con la primera, puesto que como se ha discutido en la introducción, la aplicación del método supone que cualquier desarrollador podría realizar con éxito cualquier software sólo aplicando el método de forma adecuada. Esto implica que no existen diferencias entre las personas y que por tanto no hay dimensión humana en la IS.

Para ilustrar estas dos creencias ofrecemos el estudio detallado de un estudiante, E21, en el que se observaron estas dos creencias al principio del curso.

> **Estudiante E21**
>
> Durante las primeras semanas de curso, este estudiante manifestó su insatisfacción con la asignatura. Sintió que todo se hacia de forma precipitada y con poco tiempo y que no podía realizar un buen trabajo como a él le gustaría. Añadió además que no aprendía nada nuevo, y que no había ningún aspecto de la asignatura que le pareciera interesante.
>
> En este primer momento del curso, se observaron dos creencias

fuertemente arraigadas en el estudiante. Por una parte, tenía la idea de que "para hacer software se necesita calma y planificación". Esta idea recuerda al Método, si se sigue un plan paso por paso y se hace bien, se llega necesariamente a los resultados deseados.

De forma coherente con la mencionada idea "para hacer software se necesita calma y planificación", el estudiante manifestó que en el desarrollo de software "en lugar de sprints deberían ser carreras de fondo". Los sprints son periodos de tiempo pequeños en los que se deben conseguir determinados resultados y por tanto no están relacionados con la idea de calma. Los sprints tampoco son compatibles con una planificación exhaustiva del proyecto de principio a fin, puesto que después de cada sprint puede haber cambios importantes. Por tanto, la creencia de que para hacer software se necesita calma y planificación, pudo ser una de las causas de que al principio el estudiante se sintiera incómodo con el ritmo rápido y el horizonte corto de los sprints.

La segunda creencia que se observó en el estudiante al inicio del curso es que pensaba que lo más importante en el proceso de desarrollo de software era la tecnología, que las interacciones sociales tenían poca importancia. El estudiante opinaba que de poco servía mucha comunicación y que lo que hacía falta era hacer bien la parte técnica. Por tanto, podríamos inferir que, al principio del curso, el estudiante era poco consciente de la dimensión humana de la ingeniería de software, la dimensión tecnológica era la única importante para él.

Dado que la inmersión se diseñó de acuerdo con las ideas del desarrollo de software ágil y con el objetivo de desarrollar las soft skills (estrechamente relacionadas con la dimensión humana), la reacción inicial del estudiante es comprensible. El enfoque del curso situó al estudiante en un escenario incómodo poco acorde con sus creencias

acerca de la naturaleza de la IS: (1) necesidad de calma y planificación y (2) predominio de la dimensión tecnológica sobre la dimensión humana.

Tenemos evidencias de estas dos creencias iniciales en un estudiante de los 39 que había en el curso. Sin embargo, el descubrimiento de estas creencias es valioso por dos motivos. El primero es que este descubrimiento advierte de la existencia de dichas creencias y permite comprender la reacción del estudiante y ofrecerle una ayuda personalizada. El segundo es que aunque sólo tengamos evidencias de un estudiante, podría haber otros estudiantes que también tuvieran esas creencias, pero o bien no las han exteriorizado o bien no las hemos recogido a través de los instrumentos de recogida de datos de esta investigación

A.2) Contraste con otras asignaturas

Una buena parte de los estudiantes manifestó de algún modo el fuerte contraste entre el enfoque de la asignatura IS y el resto de asignaturas que cursaban en ese semestre.

Las asignaturas que los estudiantes cursaron en el mismo semestre que Ingeniería del Software fueron las siguientes: Topología, Modelización, Programación Lógica, Programación Funcional y Sistemas Operativos. Todas ellas están muy centradas en la adquisición de hard skills y muy poco en la adquisición de soft skills. Por tanto, funcionan bien con enfoques más tradicionales.

Las características diferenciadoras de la asignatura mencionadas por los estudiantes son dos:

La libertad otorgada a los estudiantes es mucho mayor de lo habitual. El reto de crear un producto software por parte de toda la clase se plantea como una actividad poco estructurada en la que no se comunica a los estudiantes los pasos que deben seguir de forma detallada.

- La asignatura consiste en llevar a cabo un producto software. No se trata de "estudiar" para después realizar una prueba de evaluación como en otras asignaturas. El objetivo que se persigue durante el curso es que el producto software funcione, en lugar de la superación de una prueba de evaluación específica.

Estas características de la asignatura han facilitado o dificultado la adopción del nuevo enfoque en función de las características y situación personal de cada estudiante. Hay estudiantes que sintonizaron de forma rápida con el enfoque de la asignatura. Es el caso del estudiante E36 que se describe a continuación.

Estudiante E36

E36 va por la mañana a la universidad y por la tarde trabaja en una empresa que desarrolla software. Dice que es un poco duro compatibilizar el trabajo con sus estudios. Lleva casi un año en esta situación y dice que está cansado y está faltando a algunas clases. En su opinión aprende más trabajando que en clase, va contento a trabajar.

La asignatura Ingeniería del Software proporciona una experiencia más cercana al trabajo que a una clase tradicional. Por tanto, el contexto de este estudiante parece propicio para el enfoque de la asignatura. Es muy probable que las condiciones iniciales del estudiante le hayan permitido sintonizar más rápido con el enfoque.

En cuanto a las diferencias entre esta asignatura y el resto de las asignaturas en las que está matriculado, E36 manifiesta que ha dedicado más tiempo a esta asignatura que al resto. Además, E36 indica una diferencia importante entre Ingeniería del Software y el resto de las asignaturas, dice que durante el curso sólo dedica tiempo a Ingeniería del Software. Para el resto de las asignaturas estudia sólo para el examen. Este comportamiento podría estar relacionado con la experiencia del estudiante, que cree que aprende más trabajando que en clase. La

asignatura Ingeniería del Software, está planteada desde una perspectiva similar a la de un trabajo, y por tanto motiva más a este estudiante en particular.

En relación con el tiempo dedicado a las asignaturas, E36 dice: "no estudio nada y sólo hago Ingeniería del Software". En esta afirmación, utiliza el verbo hacer para la asignatura Ingeniería del Software y el verbo estudiar para el resto de las asignaturas. También hemos encontrado esta misma terminología cuando E36 mencionó que otros estudiantes le habían dicho que en esta asignatura (Ingeniería del Software) "no había que estudiar".

Esta distinción entre "estudiar" y "hacer" hace referencia a dos modos diferentes de aprender desde el punto de vista de E36. El primero, estudiar, podría interpretarse como una actividad generalmente individual, en la que el objetivo final es superar una prueba de evaluación. El segundo, hacer, está directamente relacionado con el enfoque de inmersión adoptado en el curso en el que los estudiantes crean un producto software.

Por otra parte, las fuertes diferencias del enfoque de la asignatura ocasionaron un choque cultural para otros estudiantes como se puede comprobar en las siguientes declaraciones:

E23 (cuestionario inicial): "Creo que ha sido demasiada libertad dada de repente". "Esta asignatura se lleva de forma muy distinta a las asignaturas que hemos dado hasta ahora"

E39 (cuestionario final): "Me ha parecido complicada en general, porque es muy distinta a lo que estamos acostumbrados."

No obstante, los resultados sugieren que este choque cultural fue superado por los estudiantes durante las primeras semanas del curso al vivir la experiencia y

sentirse motivados por el reto propuesto. A continuación, se muestran las declaraciones del estudiante E02 que prefiere el enfoque de la asignatura en lugar de "estudiarse unos temarios".

> E02 (cuestionario final): "La verdad que no tenia mucha idea de como iba a ser hasta que se nos puso en contacto con el cliente y pensé que seria una asignatura mas en la que estudiarse unos temarios y examinarse, pero me ha sorprendido para bien, ya que me parece mucho más dinámico y considero que he aprendido mucho mas que algunas otras asignaturas que he tenido que estudiar 1000 veces mas."

En cuanto a la libertad otorgada los estudiantes, esta produjo incomodidad y rechazo al principio en parte de los estudiantes. No obstante, fue valorada positivamente al final del curso debido al aprendizaje obtenido a través de ella. En las siguientes declaraciones del estudiante E39 se puede comprobar este sentimiento.

> E39 (cuestionario final): "Lo que más me ha gustado ha sido que nos habéis dejado bastante libertad. A veces no me gustaba porque sentía que si cuando estábamos haciendo el trabajo muy individual sin pensar en que luego había que acoplarlo al resto alguien nos hubiera llamado la atención hubiera sido todo más fácil. Pero ahora viéndolo con más perspectiva entiendo que nos ha venido muy bien, sobre todo porque en la mayor parte de asignaturas siempre nos dicen qué pasos debemos seguir y estamos poco acostumbrados a tomar decisiones y organizarnos solos."

Hemos encontrado evidencias del contraste que ha supuesto esta asignatura en 11 estudiantes. Sin embargo, como se ha argumentado anteriormente, esto no quiere decir que el resto de los estudiantes no hayan percibido este contraste. Los resultados obtenidos sugieren que ha existido un fuerte contraste entre el

enfoque adoptado por esta asignatura y el enfoque adoptado por el resto de las asignaturas del semestre y que esto ha podido suponer una barrera en las primeras semanas del curso para algunos estudiantes.

A.3) Motivación inicial de los estudiantes

Durante el curso surgieron dos cuestiones importantes alrededor de la motivación. La primera de ellas es que durante el inicio del curso los estudiantes mostraron diferentes niveles de motivación hacia la asignatura. Esta situación no es una novedad, cualquier profesor se encuentra con ella con mucha frecuencia. No obstante, el enfoque adoptado requiere necesariamente de colaboración. Por tanto, los estudiantes con escasa motivación supusieron una dificultad porque la falta de rendimiento individual de un estudiante incide en el rendimiento de su equipo y por tanto en el rendimiento de la compañía.

La segunda cuestión es que muchos estudiantes perdieron motivación al comprobar que había estudiantes que no se tomaban la asignatura tan en serio como ellos. Esto ocurrió sobre todo en el segundo sprint.

Estas cuestiones acerca de la motivación inicial se detectaron en las observaciones realizadas durante las clases de la asignatura y también fueron expuestas por los estudiantes en los cuestionarios y en las entrevistas. Dos de los cuatro estudiantes a los que se entrevistó estaban muy motivados al principio del curso y perdieron motivación durante el segundo sprint.

A continuación, se muestran dos evidencias de las entrevistas, en las que se preguntó directamente por la motivación:

> E36: "Es bastante entretenido en comparación a asignaturas más teóricas, pero es verdad que como no toda la gente se lo toma igual al final te desmotiva y te hace pensar bueno pues si nadie me responde voy a pasar"
>
> E03: "Yo me las apaño para entregar las cosas a tiempo. Hay gente que

no. Cierta parte de la gente no se toma la asignatura en serio"

B) Proceso

En esta sección se proporciona información acerca de tres situaciones relevantes que tuvieron lugar durante el proceso de aprendizaje. En primer lugar, se describe cómo un grupo de estudiantes se opuso al enfoque adoptado en la asignatura. En segundo lugar, se explica la solución adoptada para integrar en la asignatura a los estudiantes menos motivados. Por último, se muestra el aumento de motivación que tuvo lugar en los dos últimos sprints.

Tabla 4. Proceso

Rebelión	• Las condiciones del contexto inicial (creencias, contexto y motivación) pudieron contribuir a la rebelión.
Agrupación de los estudiantes menos motivados	• Creación de un nuevo grupo que integró a los estudiantes con bajo rendimiento.
Aumento de la motivación	• Motivación mayor a partir del tercer sprint. • Motivación por crear un software realizado por ellos mismos por el grupo de estudiantes. • Motivación porque la asignatura sirve como preparación para el futuro profesional.

B.1) Rebelión

Después del segundo sprint, un grupo de estudiantes manifestó su desacuerdo con el enfoque adoptado en el curso y solicitó la realización de un examen final en lugar de "trabajar" en la compañía de software. Es decir, los estudiantes solicitaron volver a lo conocido. Se rebelaron contra el enfoque de la asignatura, que como se ha mencionado anteriormente, suponía un fuerte contraste con el resto de las asignaturas cursadas hasta el momento.

La forma de evaluación solicitada por los estudiantes, el examen final, era posible en la asignatura debido a normas de la universidad, pero debía solicitarse en los primeros quince días del curso. Como la solicitud de los estudiantes se

realizó más tarde de ese plazo no se atendió de forma inmediata. Al cabo de pocos días este conflicto desapareció y los estudiantes siguieron con la asignatura tal y como estaba establecida desde el comienzo. Desconocemos las causas precisas de este cambio de parecer. Una de ellas podría ser el aumento de la motivación de los estudiantes como se detalla en un apartado posterior.

Las causas de la rebelión pudieron ser variadas y diferentes para cada estudiante. Las condiciones descritas en el apartado anterior en cuanto al contexto inicial podrían haber contribuido en alguna medida: creencias iniciales no acordes con el enfoque de esta asignatura, fuerte contraste del enfoque de esta asignatura y el enfoque del resto de asignaturas, y falta de motivación hacia la asignatura Ingeniería de Software.

Esta rebelión se presenció durante las observaciones realizadas durante las clases. A continuación, se puede encontrar un testimonio en el que se muestra la falta de motivación inicial, la solicitud del cambio de enfoque y el aumento de motivación que proporcionó al final el enfoque.

> E25 (cuestionario final): "Antes de comenzar el curso, he de reconocer que la asignatura no me interesaba demasiado. Me gusta más la parte de matemáticas que la de informática. Mi motivación durante el curso ha ido aumentando. La asignatura (aunque en algunos casos te den ganas de irte a julio, hacer un examen y olvidarte de todo el proyecto) te empieza a gustar poco a poco y cuando ves que tu trabajo sale adelante es una gran satisfacción."

B.2) Agrupación de los estudiantes menos motivados

Después del segundo sprint, se detectó que en cada equipo había estudiantes con un bajo rendimiento. Para solucionar este problema, se agrupó a dichos estudiantes en un nuevo equipo para que estuvieran obligados a trabajar para conseguir los objetivos asignados, en lugar de apoyarse en los miembros más

activos del equipo. La medida adoptada tuvo éxito y motivó a estos estudiantes a trabajar en el proyecto a partir del tercer sprint.

A continuación, se muestra un testimonio en el que se aprecia el resultado de la medida adoptada.

> E01 (cuestionario final): "Como aspecto negativo hubo miembros del equipo que no mostraron el interés esperado por la asignatura y esto conllevó que no hiciesen absolutamente nada. Cuando eso se solucionó todo fue sobre ruedas."

B.3) Aumento de la motivación (sprint 3 y sprint 4)

Los resultados del estudio muestran que la motivación de los estudiantes fue mucho mayor en el tercer y cuarto sprint. Hemos encontrado dos cuestiones que han constituido una motivación importante para los estudiantes.

La primera de ellas es la creación de un producto software, realizado por el grupo de estudiantes, sin necesidad de ninguna ayuda externa. La segunda cuestión es la percepción de que la experiencia vivida en la asignatura iba a servir como preparación para un futuro profesional cercano.

El cuestionario final del curso fue cumplimentado por 20 estudiantes, de los cuales 17 manifestaron haber aumentando su motivación al final del curso. A continuación, se muestra un conjunto de citas que muestran este aumento de motivación.

> E37: "Tras la presentación de la asignatura, la motivación era muy alta. Luego fue cayendo poco a poco hasta el final, pues cuando todo empezaba por fin a funcionar, la asignatura se veía con otros ojos."
>
> E26: "He de decir que fue la motivación de menos a más, al final la motivación venía de ver que teníamos software "funcional" hecho por

nosotros."

E34: "Al igual que con la mayoría de las asignaturas, mi motivación al principio era aprobar. Ha variado durante el curso porque he aprendido mucho más de lo que esperaba. Una de las motivaciones era que perfectamente podía ser un trabajo y me iba a valer de cara al futuro."

C) Resultados obtenidos

Esta sección expone los resultados principales del enfoque aplicado en la asignatura: la superación de creencias obstaculizadoras, el aprendizaje logrado por los estudiantes y su grado de satisfacción.

Tabla 5. Resultados obtenidos

Superacion de las creencias	• Superación de la creencia de que lo tecnología es lo más importante en la Ingeniería del Software. Toma de conciencia de la importancia de la dimensión humana.
Aprendizaje	• Menor aprendizaje en hard skills que en los cursos sin inmersión • Destaca el aprendizaje en soft skills o Habilidades de comunicación o Comportamiento y rol en una compañía o Capacidad para trabajar en situaciones carentes de información y bajo presión o Organización y planificación del trabajo. o Habilidades de negociación. o Trabajo en equipo. Colaboración versus cooperación. o Liderazgo o Confianza en sí mismos
Satisfacción	• 95% de estudiantes satisfechos o muy satisfechos.

C.1) Superación de las creencias

En la sección dedicada al contexto, se han discutido dos creencias iniciales que han supuesto una dificultad en el curso. En este apartado se analiza la superación de estas creencias durante el curso. Las creencias fueron detectadas en el estudiante E21. A continuación se ofrece un análisis de la evolución de este estudiante con respecto a sus creencias iniciales.

Estudiante E21

Con respecto a la creencia de que la tecnología es lo más importante en el desarrollo de software, el estudiante reconoció al final del curso que la comunicación es crucial. Por tanto, para este estudiante, la inmersión consiguió uno de los objetivos para los que fue diseñada: aflorar la dimensión humana de la ingeniería del software. El estudiante, que había obviado al principio las soft skills, tomó conciencia de su importancia a través de la vivencia, y se interesó por el comportamiento humano, por el juego de roles propiciado por el entorno empresarial simulado en la asignatura.

"Me ha resultado muy interesante el juego de roles en el trabajo, el ver cómo actuaba cada uno de mis compañeros en un escenario no tan académico. Conocer más de ellos, lo bueno y lo malo, así como de las lideresas. Te das cuenta de por qué hacen lo que hacen y en ese sentido entiendes su rol y forma de actuar."

En cuanto a la creencia de que para desarrollar software es necesario calma y planificación (anclada en El Método), hubo indicios que sugieren que la creencia seguía arraigada en el estudiante al finalizar el curso. En particular, el estudiante declaró en el cuestionario final que la educción de requisitos es "una labor muy complicada que debe involucrar a mucha gente y toda muy atenta y concentrada". Esta afirmación sugiere que la educción de requisitos es una tarea complicada, pero que se puede realizar de forma precisa si se pone toda la atención y todos los recursos necesarios.

Por tanto, la confianza que el estudiante tenía en el método al principio del curso parece seguir intacta, cree que a través de El Método es posible llegar a es especificación de requisitos precisa y que la incertidumbre es erradicable.

En conclusión, la superación de la primera creencia es también el aprendizaje más importante que ha llevado a cabo este estudiante. E21 descubrió durante el curso la dimensión humana de la ingeniería del software, que había pasado inadvertida para él hasta ese momento.

En cuanto a la dimensión metodológica de la Ingeniería del Software, el estudiante tenía al principio del curso unas creencias basadas en la idea cartesiana de la infabilidad del método, que se resumen en la idea de que la incertidumbre es erradicable en la Ingeniería del Software. A pesar de que el estudiante ha convivido con la incertidumbre durante un semestre, esta experiencia no sido suficiente para abandonar esta creencia. Este cambio de paradigma puede requerir de mucho tiempo.

C.2) Aprendizaje

El aprendizaje se ha estudiado desde el punto de vista del aprendizaje percibido por los estudiantes. Además de los hard skills propios de la Ingeniería del Software, relacionados con la tecnología y con la metodología de desarrollo, los estudiantes manifestaron haber aprendido distintos soft skills que se pueden resumir en los siguientes:

- Habilidades de comunicación
- Comportamiento y rol en una compañía
- Capacidad para trabajar en situaciones carentes de información y bajo presión
- Organización y planificación del trabajo.
- Habilidades de negociación.
- Trabajo en equipo. Colaboración versus cooperación.
- Liderazgo
- Confianza en sí mismos

A continuación, se exponen los aprendizajes más importantes percibidos por tres estudiantes.

Estudiante E21

En cuanto a los aprendizajes más importantes realizados por el estudiante, el primero de ellos fue darse cuenta de la importancia de la comunicación en el desarrollo de software. Este aprendizaje se produjo a través de la experiencia. En particular parece que el aprendizaje se produjo después de comprobar las consecuencias de una comunicación deficiente, ya que el estudiante manifestó tanto haber realizado código que no se conectaba bien con el de otros equipos como no haberse comunicado bien con el resto de equipos cuando fue líder.

El segundo aprendizaje fue, en palabras del estudiante, "aprender cómo debe comportarse y cuál es su rol idóneo en una compañía". El estudiante dijo que se encontraba preparado para trabajar en una empresa porque se podía hacer una idea de cómo sería. Además, dijo que pensaba que no volvería a repetir los errores cometidos durante el curso, y que mejoraría en los aspectos en los que creía que flaqueaba. Del mismo modo que el aprendizaje anterior, el estudiante manifestó que había adquirido este aprendizaje mediante ensayo y error. Es decir, ambos aprendizajes se han propiciado en la inmersión diseñada. El estudiante reconoció este hecho afirmando "es algo que no podría haber aprendido con ninguna otra asignatura" y que el aprendizaje en la empresa hubiera sido más duro.

Además, el estudiante manifestó haber mejorado de forma considerable su capacidad para trabajar en situaciones carentes de información y bajo presión. Esta mejora se realizó durante todo el curso, pero especialmente cuando desempeñó el papel de líder, ya que sintió mucha más presión.

El estudiante también percibió que había mejorado sus competencias de organización y planificación tanto del trabajo individual como del trabajo en equipo a lo largo del curso. La misma percepción de mejora ocurrió con el aprendizaje autónomo al que ya estaba acostumbrado durante sus estudios, y que siguió mejorando con esta asignatura.

Por último, el estudiante manifestó haber consolidado conocimientos previos de bases de datos, sobre todo a través de la consulta en Internet.

En resumen, el estudiante consideró que se había producido un salgo cualitativo en el aprendizaje de Ingeniería del Software, dijo "hemos tenido que dar un salto considerable de nivel en cada uno de nuestros aspectos para estar al nivel del proyecto y de sus exigencias".

Esta afirmación del estudiante sugiere que los aspectos a los que se refiere son de distinta índole. Parece estar haciendo referencia a las hard y soft skills que están presentes en la Ingeniería del Software y podría ser la toma de conciencia de que la Ingeniería del Software es mucho más que tecnología.

Si analizamos todos los aprendizajes mencionados de este estudiante, podemos observar que todos excepto el último (consolidación de conocimientos sobre bases de datos), son soft skills (importancia de la comunicación, comportamiento en una compañía, trabajo en situaciones de presión, organización y planificación, aprendizaje autónomo). Por tanto, los aprendizajes percibidos por este estudiante muestran el logro de uno de los principales objetivos del enfoque.

Estudiante E36

Con referencia a los requisitos, E36 manifiesta después del segundo sprint que "la toma de requisitos ha servido de ayuda para aprender a ser detallista y dejar todo por escrito". Esta afirmación hace referencia a que

E36 ha aprendido ha realizar un documento de requisitos. En el cuestionario final, resume la utilidad de la experiencia que ha tenido con los requisitos con la siguiente frase: "Me ha sido de gran ayuda para que a la hora de trabajar fuera de la burbuja estudiantil el mercado no te coma y se aprovechen de ti, el tiempo es oro". Las palabras de E36 sugieren un aprendizaje acerca de habilidades de negociación, una soft skill esencial en la comunicación con el cliente. Es decir, el aprendizaje acerca de requisitos ha trascendido lo meramente técnico (escribir un documento de requisitos), E36 ha aprendido a comunicarse con el cliente y a llegar a un acuerdo con él.

Otra competencia que ha desarrollado este estudiante es la organización del trabajo en equipo. Después de su experiencia con los primeros sprint, manifestó que podrían haber tenido menos reuniones y repartido mejor el trabajo y hubieran llegado al mismo punto. Además, el estudiante declaró "que debería haber acaparado menos trabajo confiando en los demás miembros". Ambas afirmaciones sugieren que E36 ha mejorado su competencia para organizar el trabajo dentro un equipo.

Estudiante E01

E01 es un estudiante que ha tenido un compromiso muy alto con la asignatura. Según sus propias palabras es la asignatura en la que más ha aprendido con mucha diferencia. En cuanto a su aprendizaje de soft skills llama la atención la visión clara de este estudiante con respecto al trabajo en equipo. Dice cuando se trabaja en equipo todos los miembros trabajan en interés del equipo. En contraste, cuando se trabaja en grupo se trabaja con un interés personal.

Consideramos que este ha sido un aprendizaje muy importante, puesto que los estudiantes están muy acostumbrados a trabajar en grupo, pero

menos a trabajar en equipo. Es decir, los estudiantes suelen dividir los trabajos que hacen en grupo y trabajan de forma cooperativa, pero poniendo el foco en su calificación individual. Sin embargo, el trabajo en equipo requiere colaboración para realizar un producto común, el crédito es de todos. Este espíritu colaborativo se ha fomentado en la asignatura considerando la calificación de la compañía como el 40% de la calificación individual de cada estudiante.

E01 fue líder de su equipo y opinó que su experiencia como líder fue muy positiva porque le permitió involucrarse mucho en el proyecto. Al principio tuvo dificultades con la gestión del equipo, que ser resolvieron más adelante. Con respecto a la gestión de equipos, su aprendizaje más valioso es que lo más importante es saber motivar a los miembros del equipo. También añadió que la comunicación en los equipos es muy difícil cuando no hay motivación.

En relación con la mejora de la comunicación entre los distintos equipos que forman la compañía, el estudiante planteó una nueva forma de actuar basada en su experiencia adquirida. Comprobó que las reuniones celebradas durante las últimas semanas del proyecto funcionaron y propuso esta solución como técnica para la comunicación entre equipos: "que todos los líderes comenten sus avances y debatan sus problemas un par de veces por semana".

Como resumen de su aprendizaje, el estudiante menciona que ha aprendido principalmente a trabajar en equipo, resolver problemas y encontrar alternativas bajo presión. Manifiesta que no puede explicar cómo lo ha aprendido, que no ha habido un proceso formal ni consciente, dice haber aprendido sin darse cuenta.

Por otra parte, el estudiante piensa que uno de los aspectos más interesantes de la asignatura es la necesidad de resolver los problemas

> que iban apareciendo, cree que durante la resolución de problemas se produce mucho aprendizaje.

El objetivo principal del curso ha sido que los estudiantes vivan la Ingeniería del Software y que descubran todas sus dimensiones, incluida la dimensión humana que ha podido pasar inadvertida hasta el momento. Consideramos que los aprendizajes más importantes para los estudiantes han sido el trabajo en equipo, la comunicación en la compañía y el aumento de confianza en sí mismos, lo que se refleja en las siguientes citas:

> E26: "En cuanto al trabajo en equipo y comunicación entre equipos se ha aprendido mucho yo creo, lo cual es fundamental para el sector en el que aspiramos a trabajar."
>
> E39: "Aunque no he aprendido un temario como en el resto de asignaturas siento que he aprendido a trabajar en equipo, a tener una visión global de los requisitos y de las tareas en una empresa y creo que son valores que no se me van a olvidar y que me serán muy útiles en el futuro"
>
> E37: "Estoy satisfecha con mi aprendizaje, he aprendido la importancia de la comunicación, y a nivel de programación también he aprendido cosas."
>
> E25: "Estoy bastante satisfecho con mi aprendizaje, me he demostrado a mi mismo que puedo ser capaz de llevar a un equipo y que las cosas funcionen medianamente bien y además aprender cosas nuevas sin ayuda de una persona que esté encima de mi."

No obstante, también es relevante tomar en consideración las palabras del estudiante E10, al que le hubiera gustado profundizar más en los aspectos

técnicos.

> E10: "Estoy satisfecho con el aprendizaje desde el punto de vista social y de comunicación. En el resto de aspectos, creo que hemos aprendido más bien poco, como comenté en anteriores preguntas."

C.3) Satisfacción con la asignatura

En primer lugar, presentamos los resultados de la pregunta 24 del cuestionario final, en la que se preguntaba a los estudiantes si había merecido la pena cursar la asignatura. Los estudiantes debían contestar con la siguiente escala: Strongly Disagree, Disagree, Neutral, Agree y Strongly Agree.

De los 20 estudiantes que contestaron el cuestionario final, catorce contestaron Strongly Agree, 5 contestaron Agree y 1 contestó Disagree. Por tanto 19 de los 20 estudiantes estuvieron satisfechos con la asignatura. Se consiguió un alto porcentaje de satisfacción, incluso en los estudiantes que habían comenzado la asignatura con una baja satisfacción como el estudiante E21 cuyo caso presentamos a continuación.

> **Estudiante E21**
>
> La satisfacción de este estudiante se transformo por completo durante el curso. Comenzó el curso con una gran insatisfacción y terminó muy satisfecho. En el cuestionario final del curso el estudiante manifestó que estaba completamente de acuerdo en que había merecido la pena cursar la asignatura y que se sentía preparado para incorporarse a una empresa
> Declaró además que había aprendido mucho en la asignatura, que lo que había aprendido le había gustado y que el grado de satisfacción era muy alto porque había aprendido sin que nadie se lo hubiese enseñado. Además, se mostró muy satisfecho porque se logró el reto, el trabajo salió

adelante y la compañía de software logró desarrollar el producto deseado.

5. Discusión

A continuación, se discuten los principales hallazgos del estudio. En primer lugar, se expone la tríada motivación, reto y libertad que ha constituido el eje principal de enfoque. Después, se analizan las competencias adquiridas por los estudiantes agrupados por las tres dimensiones de la IS: humana, tecnológica y metodológica. Para finalizar, se realiza una reflexión sobre las implicaciones educativas del enfoque.

5.1 Motivación, reto y modulación de la libertad

Una de las cuestiones más relevantes de los resultados obtenidos es que muchos estudiantes se sintieron muy motivados en los dos últimos sprints del curso. Como hemos descrito anteriormente, su motivación era lograr el funcionamiento del sistema software creado y la comunicación de todas sus partes. Los estudiantes se han involucrado en esta actividad por el interés que tenía la actividad en sí misma y no como medio para conseguir un fin, lo que indica que su motivación fue intrínseca (Ryan & Deci, 2000).

Esta motivación intrínseca fue una diferencia importante con los dos cursos anteriores (primera y segunda inmersión), en los que no se observó este fenómeno. Una explicación plausible para la motivación intrínseca es la mejora de la adecuación del reto a conseguir. El reto fue percibido por los estudiantes con una dificultad adecuada.

A su vez, esta mejora de la adecuación del reto pudo estar modulada por la gradual restricción de la libertad que se proporcionó en la segunda y tercera inmersión. En la primera inmersión hubo mucha libertad. Los equipos de trabajo fueron formados por los estudiantes sin ninguna limitación por parte del profesor. En la segunda inmersión se redujo algo más la libertad. Los equipos fueron formados por el profesor y se incluyeron dos estudiantes como

apoyo técnico y organizativo con capacidad para participar en la evaluación. En la tercera inmersión, el curso estudiado en este trabajo, se restringió todavía más la libertad. Los dos estudiantes de apoyo actuaron como jefes en lugar de como asesores, y además proporcionaron un apoyo técnico mucho mayor que en el curso anterior.

Los resultados de la investigación sugieren que el aumento de motivación intrínseca pudo deberse a que la restricción de la libertad facilitó el camino de los estudiantes para conseguir el reto propuesto. Esta restricción de libertad es sutil porque los estudiantes siguieron sintiendo el proyecto como una obra suya, sin intervención del profesorado, y sin que nadie les obligara a realizar las tareas de una forma determinada. Las sucesivas aproximaciones al enfoque educativo nos han permitido encontrar un balance en la guía proporcionada a los estudiantes para que el reto resulte motivador.

Un indicio adicional de la motivación existente en esta tercera inmersión es el número de estudiantes que han cursado esta asignatura y se han postulado para participar como voluntarios para el próximo curso. En la primera inmersión surgió un voluntario que participó en el siguiente curso junto con un estudiante de otra titulación (Ingeniería Informática). En la segunda inmersión surgió también un único voluntario que participó en el siguiente curso junto con el estudiante voluntario de la primera inmersión (que repitió la experiencia). En la tercera inmersión surgieron dos voluntarios para el siguiente curso.

En resumen, el curso se ha apoyado en tres elementos clave: el reto de construir un sistema software, la libertad guiada que se otorgó al alumnado y la motivación generada por ese propio reto. Estos elementos se reforzaron mutuamente: ajustar el grado de libertad convirtió la tarea en un verdadero desafío, y ese desafío despertó una motivación intrínseca en los estudiantes. Presuntamente, esta motivación intrínseca ha sido el motor del curso que ha

permitido que los estudiantes superaran el choque cultural producido por la asignatura y la falta de motivación inicial. Es decir, la motivación ha permitido la participación en el proyecto software y la consiguiente vivencia de la Ingeniería del Software.

A continuación, se analizan los resultados obtenidos desde el punto de vista de las tres dimensiones de la Ingeniería del Software: humana, metodológica y técnica.

5.2 Dimensión humana de la Ingeniería del Software

Una de las cuestiones más relevantes de los resultados obtenidos es que los estudiantes tomaron conciencia de la dimensión humana de la Ingeniería del Software. La situación de partida de los estudiantes con respecto a esta cuestión era diversa. Al principio del curso algunos estudiantes tenían la creencia de que lo más importante en la Ingeniería del Software es la tecnología a diferencia de otros estudiantes que eran ya más conscientes de la importancia de lo humano. Independientemente de sus creencias iniciales, todos los estudiantes tuvieron la ocasión de vivir la Ingeniería del Software de forma holística durante el desarrollo del curso. Por tanto, todos ellos pudieron comprobar la influencia de las soft skills en los resultados del proyecto y tomar conciencia de su importancia.

Además de la toma de conciencia, los estudiantes han desarrollado soft skills entre los que destacan las habilidades de comunicación, el trabajo en equipo, el liderazgo y la confianza en sí mismos. Los tres primeros son soft skills considerados relevantes para la práctica de la IS (Garousi et al., 2020; Matturro et al., 2019). Los dos primeros están incluidos de forma explícita en la última versión del Software Engineering Body of Knowledge (Bourque & Fairley, 2014). El último es un soft skill esencial para el desempeño de cualquier puesto de trabajo.

Pensamos que este enfoque ha sido efectivo porque los estudiantes han tenido la oportunidad de vivir en primera persona la influencia del factor humano en la Ingeniería del Software. Aunque la Ingeniería del Software se basa en metodologías que establecen procesos para desarrollar un producto, el éxito final reside en cómo las personas ejecutan esos procesos (Capretz, 2014; Capretz & Ahmed, 2018).

Cabe señalar que este curso tiene entre sus objetivos principales potenciar las soft skills. Sin embargo, no se trata de un curso centrado en soft skills de forma aislada y descontextualizada como las planteadas en (Dell'Aquila et al., 2017). Por el contrario, las soft skills están incluidas de forma implícita y holística, junto con el resto de las competencias necesarias para la práctica de la Ingeniería del Software. En un curso específico sobre soft skills, es posible que estas se desarrollen, pero no se tomaría conciencia de la importancia de la dimensión humana de la Ingeniería del Software porque no se habría vivido la experiencia. Además, el enfoque usado elude la necesidad planteada por Maturro (2019) de conocer y definir de antemano los soft skills para poder enseñarlos y aprenderlos. Dado el contexto establecido para el curso, las soft skills se desarrollan sin haberlas planificado una a una de antemano.

Consideramos que la toma de conciencia de la dimensión humana de la IS es tan importante como el desarrollo de soft skills. La razón de esta afirmación es que esta toma de conciencia puede influir profundamente en la toma de decisiones y en el comportamiento de las personas en un proyecto de Ingeniería de Software, tanto desde el punto de vista del liderazgo como desde el punto de vista de miembro de un equipo.

5.3 Dimensión tecnológica de la Ingeniería del Software

Los estudiantes han desarrollado las competencias tecnológicas necesarias para poder llevar a cabo el proyecto propuesto. Por tanto, los estudiantes han

aprendido lo esencial para que el software funcione y se comunique. No obstante, los resultados del proyecto son mejorables en cuanto a la calidad del diseño software, la realización de pruebas y la usabilidad.

Es preciso señalar que en los cursos anteriores a la inmersión se desarrollaban en mayor medida competencias tales como diseño software y pruebas. La inmersión diseñada ha puesto el foco en la holística de la Ingeniería del Software y como consecuencia se ha podido profundizar menos en algunas competencias dado que se ha mantenido el número créditos de la asignatura.

Sin embargo, aunque no hayan adquirido las competencias al más alto nivel, los estudiantes han vivido la necesidad de profundizar en el desarrollo de las competencias diseño software, pruebas y usabilidad y esto constituye un aprendizaje en sí mismo. Esta necesidad, el conocimiento acerca de su importancia y su utilidad, podría constituir una fuente de motivación intrínseca para aprender estas competencias una vez finalizado este curso.

Hay además otra cuestión relativa a las competencias técnicas. En este curso se dedicaron las primeras semanas del curso, antes de comenzar con el proyecto, a impartir clases sobre diseño software, pruebas y usabilidad. Se esperaba que estas clases facilitaran el reto a conseguir y evitaran la protesta que tuvo lugar en la segunda inmersión. Sin embargo, este objetivo no se consiguió, ya que la protesta se repitió en esta tercera inmersión. Los resultados de este estudio sugieren que la rebelión no se produjo por falta de conocimientos técnicos, sino por el choque cultural producido por el uso de un enfoque docente al que no estaban acostumbrados.

Para terminar, se describen a continuación las competencias técnicas presentes en el enfoque. Al principio del curso, los estudiantes disponían de conocimientos técnicos aprendidos en asignaturas anteriores (programación y bases de datos). Durante el curso, los estudiantes desarrollaron competencias

(desarrollo web) y consiguieron que el proyecto software funcionara. Al finalizar el curso, los estudiantes tomaron conciencia de la importancia del diseño software, las pruebas y la usabilidad porque vivieron la necesidad de las mismas.

5.4 Dimensión metodológica de la Ingeniería del Software

La metodología usada en el curso ha seguido los principios del desarrollo ágil. Se llevaron a cabo un total de cuatro sprints y a la finalización de cada uno, el cliente se reunión con la compañía para proporcionar retroalimentación.

La mayoría de los estudiantes no ha manifestado de forma explícita un aprendizaje sobre la dimensión metodológica. Una de las causas posibles es que la metodología seguida fue muy relajada, no se impuso una disciplina de trabajo. Por ejemplo, se establecieron dos tipos de reunión diaria (los días que había clase de esta asignatura): una reunión diaria de equipos y una reunión diaria de líderes de equipos. Las dos reuniones tuvieron dificultades. La mayoría de los equipos no celebró su reunión diaria de forma regular y las reuniones de líderes se realizaron de forma diaria, pero poco coordinada y eficiente. En la última etapa del curso, la calidad de las reuniones de los líderes mejoró notablemente debido a la intervención de las lideresas de la compañía. La experiencia adquirida en estas reuniones ha sido percibida como un aprendizaje para el estudiante E01.

Por otra parte, hemos detectado una creencia que dificulta la comprensión y la adopción de la metodología ágil. Los que la poseen sostienen que la incertidumbre es erradicable en la Ingeniería del Software y que la aplicación de un método de forma correcta asegurará los resultados deseados. Esta creencia la detectamos al inicio del curso y hemos comprobado que al final del curso todavía existía. Este hecho llama la atención sobre la dificultad de erradicar esta creencia en algunos estudiantes y la posible permanencia de la misma durante

la vida profesional.

5.5 Implicaciones educativas

Cuando comenzamos a poner en marcha el enfoque de inmersión en la asignatura Ingeniería del Software, pensamos en que esta asignatura podría ser un colofón final sobre el aprendizaje en Ingeniería del Software. Sin embargo, después de estos tres años de experiencia pensamos en este curso como un primer contacto con la Ingeniería del Software en el que se viven todas sus dimensiones. El curso presentado resalta la necesidad de todos los elementos importantes de la IS, especialmente las soft-skills que pueden haber sido invisibles hasta ese momento.

El enfoque adoptado tiene como objetivo desarrollar las soft skills necesarias para la ingeniería del software de una forma holística, en un contexto de inmersión. De este modo, el curso diseñado constituye una introducción a la Ingeniería del Software en la que se vive la necesidad de cada uno de los diferentes elementos que la componen. Así pues, la asignatura puede proporcionar la motivación para el estudio de las otras asignaturas más específicas como diseño software, gestión de proyectos, usabilidad, etc, puesto que los estudiantes pueden enlazar los conceptos y técnicas estudiados con las experiencias previas en las que sintieron la necesidad de los mismos.

Esta posición está de acuerdo con dos de los fundamentos propuestos por Comenius (Comenius, 1896), hay que conocer la enfermedad antes que los remedios y lo que naturalmente está unido se debe considerar conjuntamente y no por separado. Si aplicamos estos principios al aprendizaje de la Ingeniería del Software, el curso presentado permite conocer lo que pasa si no hay un diseño software, un nivel de usabilidad y un plan de pruebas apropiados. Por tanto, el enfoque adoptado permite conocer la enfermedad antes que los remedios. De este modo, cuando un estudiante se enfrente a cada una de estas

materias en el futuro, las abordará con mayor interés y claridad en cuanto a su uso. En cuanto al segundo principio mencionado, en la Ingeniería del Software están naturalmente unidas de forma holística todas sus dimensiones. Por tanto, conviene tenerlas presentes todas a la vez al menos en una asignatura para poder dotar de significado a cada una de las partes.

Por otra parte, el enfoque presentado está alineado con el modelo competencial (competency model) adoptado por el Computing Curricula 2020. Tal y como sugiere el curriculum, el curso presentado supone una forma de "refreshing the paradigm of teaching and educating, moving from knowledge or outcomes to proficiencies, and engaging graduates to exploit the benefits of workplace competencies" (Association for Computing Machinery & IEEE Computer Society, 2020).

6. Conclusiones

En este artículo hemos compartido nuestra experiencia acerca de la puesta en marcha de un enfoque educativo que tiene en cuenta la holística de la IS y está sintonizado con el nuevo paradigma de la IS en el que lo humano es protagonista.

Para analizar los resultados del curso hemos aplicado de forma rigurosa la metodología de investigación cualitativa. Dichos resultados muestran que los estudiantes han adquirido competencias de IS y en especial, han desarrollado *soft skills* imprescindibles en IS, entre las que se encuentran las habilidades de comunicación, el trabajo en equipo y el liderazgo. Además, los estudiantes han modificado sus actitudes a través de la toma de conciencia de la dimensión humana de la IS.

Las contribuciones para la comunidad educativa son dos: el diseño de nuestro enfoque y los resultados obtenidos con el mismo. El primero puede servir como referencia para poner en marcha enfoques similares y los segundos pueden

ayudar a comprender lo que sucede en el aula cuando se esté utilizando este enfoque.

La contribución para la comunidad de investigadores en Software Engineering Education es el diseño de la investigación cualitativa y en especial el procedimiento llevado a cabo durante el análisis de datos. Las investigaciones cualitativas no son frecuentes en IS Education. Por añadidura, el procedimiento concreto de análisis se diseña para cada investigación, no es siempre el mismo como en el caso de las investigaciones cualitativas. Por tanto, esta investigación puede servir como referencia para llevar a cabo investigaciones educativas con metodología cualitativa en IS education.

Referencias bibliograficas

Appelo, J. (2011). *Management 3.0. Leading Agile Developers. Developing Agile Leaders.* Addison-Wesley.

Assyne, N., Ghanbari, H., & Pulkkinen, M. (2022). The state of research on software engineering competencies: A systematic mapping study. *Journal of Systems and Software*, *185*, 111183. https://doi.org/10.1016/J.JSS.2021.111183

ATLAS.ti Scientific Software Development GmbH. (2020). *ATLAS.ti.*

Bacon, F. (1620). *Novum Organum*. St. Louis Publishing Company.

Beck, K., Beedle, M., Bennekum, A. van, Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., & Thomas, D. (2001). *Agile Manifesto. Principles.* https://agilemanifesto.org/principles.html

Berard, E. V. (1993). Abstraction, encapsulation, and information hiding. In E. V. Berard (Ed.), *Essays on Object-Oriented Software Engineering* (Vol. 1, pp. 63–73). Prentice-Hall.

Biddle, B. J., & Anderson, D. S. (1986). Theory, Methods, Knowledge and Research on Teaching. In M. C. Wittrock (Ed.), *Handbook of research on teaching: a project of the American Educational Research Association* (pp. 230–252). Macmillan; Collier-Macmillan.

Blum, B. I. (1992). *Software engineering : a holistic view*. Oxford University Press.

Boehm, B. W. (1988). A spiral model of software development and enhancement. *Computer*, *21*(5), 61–72.

Bourque, P., & Fairley, R. E. (Eds.). (2014). *SWEBOK V3.0 Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society.

Broman, D., Sandahl, K., & Abu Baker, M. (2012). The Company Approach to Software Engineering Project Courses. *IEEE Transactions on Education*, *55*(4), 445–452. https://doi.org/10.1109/TE.2012.2187208

Capretz, L. F. (2014). Bringing the human factor to software engineering. *IEEE Software*, *31*(2), 102–104. https://doi.org/10.1109/MS.2014.30

Capretz, L. F., & Ahmed, F. (2018). A Call to Promote Soft Skills in Software

Engineering. *Pyschology and Cognitive Sciences*, *4*(1), e1–e3. https://doi.org/10.17140/PCSOJ-4-e011

Cico, O., Jaccheri, L., Nguyen-Duc, A., & Zhang, H. (2021). Exploring the intersection between software industry and Software Engineering education - A systematic mapping of Software Engineering Trends. *Journal of Systems and Software*, *172*, 110736. https://doi.org/10.1016/J.JSS.2020.110736

Comenius, J. A. (1896). *The Great Didactic of John Amos Comenius*. Adam and Charles Black.

Dell'Aquila, E., Marocco, Davide, Ponticorvo, M., Di Ferdinando, A., Schembri, M., & Miglino, O. (2017). Educational Games for Soft-Skills Training in Digital Environments. In *Educational Games for Soft-Skills Training in Digital Environments*. Springer International Publishing. https://doi.org/10.1007/978-3-319-06311-9

DeMarco, T., & Lister, T. (1987). *Peopleware. Productive Projects and Teams*. Pearson Education.

Descartes, R. (1649). *A discourse of a method for the well-guiding of reason, and the discovery of truth in the sciences*. Thomas Newcombe.

Ellis, R. A., Han, F., & Pardo, A. (2019). When Does Collaboration Lead to Deeper Learning? Renewed Definitions of Collaboration for Engineering Students. *IEEE Transactions on Learning Technologies*, *12*(1). https://ieeexplore.ieee.org/document/8359494/

Flyvbjerg, B. (2006). Five Misunderstandings about Case-Study Research. *Qualitative Inquiry*, *12*(2), 219–245. https://doi.org/https://doi.org/10.1177/1077800405284363

Fromm, E. (1994). *Escape from freedom*. Henry Holt and Company.

Garousi, V., Giray, G., Tüzün, E., Catal, C., & Felderer, M. (2019). Aligning software engineering education with industrial needs: A meta-analysis. *Journal of Systems and Software*, *156*, 65–83. https://doi.org/10.1016/J.JSS.2019.06.044

Garousi, V., Giray, G., Tuzun, E., Catal, C., & Felderer, M. (2020). Closing the Gap between Software Engineering Education and Industrial Needs. *IEEE Software*,

37(2), 68–77. https://doi.org/10.1109/MS.2018.2880823

Hazzan, O. (2010). Putting Human Aspects of Software Engineering in University Curricula. *IEEE Software*, *27*(4), 90–91. https://doi.org/10.1109/MS.2010.104

John Wiley & Sons. (2022). *Everything DiSC*. https://www.everythingdisc.com/

Klir, G. J. (1988). *Fuzzy Sets, Uncertainty and Information*. Prentice Hall.

Liargkovas, G., Papadopoulou, A., Kotti, Z., & Spinellis, D. (2021). Software Engineering Education Knowledge Versus Industrial Needs. *IEEE Transactions on Education*, *PP*. https://doi.org/10.1109/TE.2021.3123889

Licorish, S. A., Galster, M., Kapitsaki, G. M., & Tahir, A. (2022). Understanding students' software development projects: Effort, performance, satisfaction, skills and their relation to the adequacy of outcomes developed. *Journal of Systems and Software*, *186*, 111156. https://doi.org/10.1016/J.JSS.2021.111156

Lincoln, Y. S., & Guba, E. G. (1985). *Naturalistic Inquiry*. SAGE Publications.

Martí, J. (1977). *Simple Verses*. Arte Público Press.

Maslow, A. H. (1987). *Motivation and Personality* (Third Edition). Harper & Row, Publishers.

Matturro, G., Raschetti, F., & Fontán, C. (2019). A systematic mapping study on soft skills in software engineering. *Journal of Universal Computer Science*, *25*(1), 16–41.

Merrian, S. B. (1998). *Qualitative Research and Case Study Applications in Education* (Second Edi). Jossey-Bass.

Oguz, D., & Oguz, K. (2019). Perspectives on the Gap between the Software Industry and the Software Engineering Education. *IEEE Access*, *7*, 117527–117543. https://doi.org/10.1109/ACCESS.2019.2936660

Paasivaara, M., Heikkilä, V. T., Vanhanen, J., & Lassenius, C. (2018). How Does Participating in a Capstone Project with Industrial Customers Affect Student Attitudes? *Proceedings of the 40th International Conference on Software Engineering: Software Engineering Education and Training*, 49–57. https://doi.org/10.1145/3183377

Papert, S. (1993). *The Children's Machine: Rethinking School in the Age of the Computer*. BasicBooks.

Parnas, D. L. (1972). On the Criteria To Be Used in Decomposing System into Modules. *Communications of the ACM*, *15*(12), 1053–1058.

Parnas, D. L. (2002). The secret history of information hiding. In M. Broy & E. Denert (Eds.), *Software Pioneers* (pp. 399–409). Springer.

Pressman, R. S. (2001). *Software Engineerning. A Practitioner's Approach*. McGraw-Hill Higher Education.

Prigogine, I. (1984). Only an Illusion. In S. M. McMurrin (Ed.), *The Tanner Lectures on Human Values. Volume V*. (pp. 35–64). University of Utah Press.

Raibulet, C., & Arcelli Fontana, F. (2018). Collaborative and teamwork software development in an undergraduate software engineering course. *Journal of Systems and Software*, *144*, 409–422. https://doi.org/10.1016/J.JSS.2018.07.010

Royce, W. W. (1970). Managing the Development of Large Software Systems. *Proceedings IEEE WESCON*, 1–9.

Ryan, R. M., & Deci, E. L. (2000). Intrinsic and Extrinsic Motivations: Classic Definitions and New Directions. *Contemporary Educational Psychology*, *25*(1), 54–67. https://doi.org/10.1006/CEPS.1999.1020

Tvedt, J. D., Tesoriero, R., & Gary, K. A. (2010). The Software Factory: An Undergraduate Computer Science Curriculum. *Http://Dx.Doi.Org/10.1076/Csed.12.1.91.8213*, *21*(1), 91–117. https://doi.org/10.1076/CSED.12.1.91.8213

Yourdon, E., & Constantine, L. L. (1979). *Structured Design: Fundamental of a Discipline of Computer Program and System Design*. Prentice-Hall.