07-003

The review of software development methodologies and their evolution towards the user experience

Erik Aranburu Zabalo; Ganix Lasa Erle; Daniel Justel Lozano

Escuela Politécnica Superior de Mondragon Unibertsitatea;

Since the arrival of the digital systems and the computer in particular, the development of the software of such systems have been a topic of great concern and importance. From the time of its creation the ideologies or perspectives of its development have been changing, suffering an evolution caused significantly by the changes of the social and economic paradigm. That's way there is a great amount of methodologies that pose their methods and approaches of how to develop software applications.

Starting from this context, this communication presents a review of the methodologies that have been developed during the last years. As a result, this review shows a chronological analysis of the methodologies, comparing them by their perspective, the approach, the phase of the process that they are aiming and their application for real cases.

Keywords: Software Development; user experience; review; methodology

Revisión de metodologías de desarrollo de software y su evolución hacia la experiencia de usuario

Desde la llegada de los sistemas digitales y en concreto los ordenadores, el desarrollo del software de dichos sistemas ha sido un tema de gran interés e importancia. Desde su creación las ideologías o perspectivas de su desarrollo han ido variando, sufriendo una evolución que en gran medida está sujeta a los cambios del paradigma tanto social como económico. Es por ello que existe una gran cantidad de metodologías que plantean sus métodos y enfoques de cómo desarrollar aplicaciones de software.

Partiendo de este contexto, la comunicación presenta una revisión de las metodologías que se han ido desarrollando a lo largo de los últimos años. Como resultado se obtiene un análisis cronológico de las metodologías, comparándolas en base a la perspectiva, el enfoque, la fase del proceso a la que se dirigen y su aplicación para casos reales.

Palabras clave: *Desarrollo software; experiencia de usuario; revisión; metodología*

Correspondencia: Erik Aranburu Zabalo (e-mail: erik.aranburu@alumni.mondragon.edu)

Centro de Innovación en Diseño (DBZ). Escuela Politécnica Superior de Mondragon Unibertsitatea. C/ Loramendi 4, 20500 Arrasate – Mondragón (España).

Agradecimientos: Los autores agradecemos el apoyo recibido por el Diseinu Berrikuntza Zentroa (DBZ) de Mondragon Unibetsitatea y la Escuela Politécnica Superior de Mondragon Unibertsitatea.

1. Introducción

La imparable evolución tecnológica ha ido revolucionando el paradigma social y económico a lo largo de los años. Uno de los acontecimientos principales en dicha transformación fue la llegada de las tecnologías de información. La aparición de los ordenadores supuso un gran cambio socioeconómico, cambió la forma de vida de las personas y modificó la industria por completo, por lo que desde entonces el desarrollo software ha sido un tema transcendental.

Los sistemas digitales siguen evolucionando y el software debe seguir avanzando para poder cubrir las nuevas necesidades emergentes. Los dominios de la aplicación, las técnicas de análisis y diseño, lenguajes de programación y las estrategias de proyecto van variando conforme la tecnología avanza, por lo que el desarrollo software debe adaptarse a la variabilidad (Brinkkemper, 1996). En dicha evolución, los sistemas deben ser capaces de adaptarse a las nuevas necesidades del entorno y sus clientes, para ofrecer una aplicación que mejore la calidad y satisfacción de uso, pero manteniendo la familiaridad del producto e igualando o disminuyendo la dificultad de su uso (Lehman y Perry, 1997).

Para ello, es crucial entender que el cambio debe ser parte central del desarrollo software y que la ingeniería software deben estar predispuesta y preparada para una continua transformación. Con el objetivo de cubrir dicha necesidad de adaptación es importante crear soportes, los cuales se podrían clasificar en los siguientes niveles (Mens et al., 2005):

1. Investigación en formalismos y teorías para analizar, comprender, gestionar y controlar los cambios de software.

2. El desarrollo de modelos, lenguajes, herramientas, métodos, técnicas y heurísticos para ofrecer soporte a dichos cambios.

3. Validaciones reales en casos de estudio de sistemas de software complejos de la industria.

Acorde con el segundo nivel de soportes planteados, desde los inicios de los sistemas digitales se ha creado una gran variedad de metodologías y herramientas para la adaptación de los sistemas software a la continua evolución. Los métodos y herramientas tienen como objetivo servir como guía de trabajo para su implementación en casos reales, y aportan una serie de ventajas. Ayudan a dividir el proceso en pasos estructurados para así reducir la complejidad, facilitan la gestión de proyecto, la división de tareas y el control del proceso reduciendo el riesgo. Además pueden aportar técnicas y herramientas que ayuden la aplicación en cada paso. Como consecuencia, se consigue que los proyectos elaborados bajo una metodología concreta adquiera mayor calidad y suponga una mayor satisfacción del usuario (Fitzgerald, 1998)

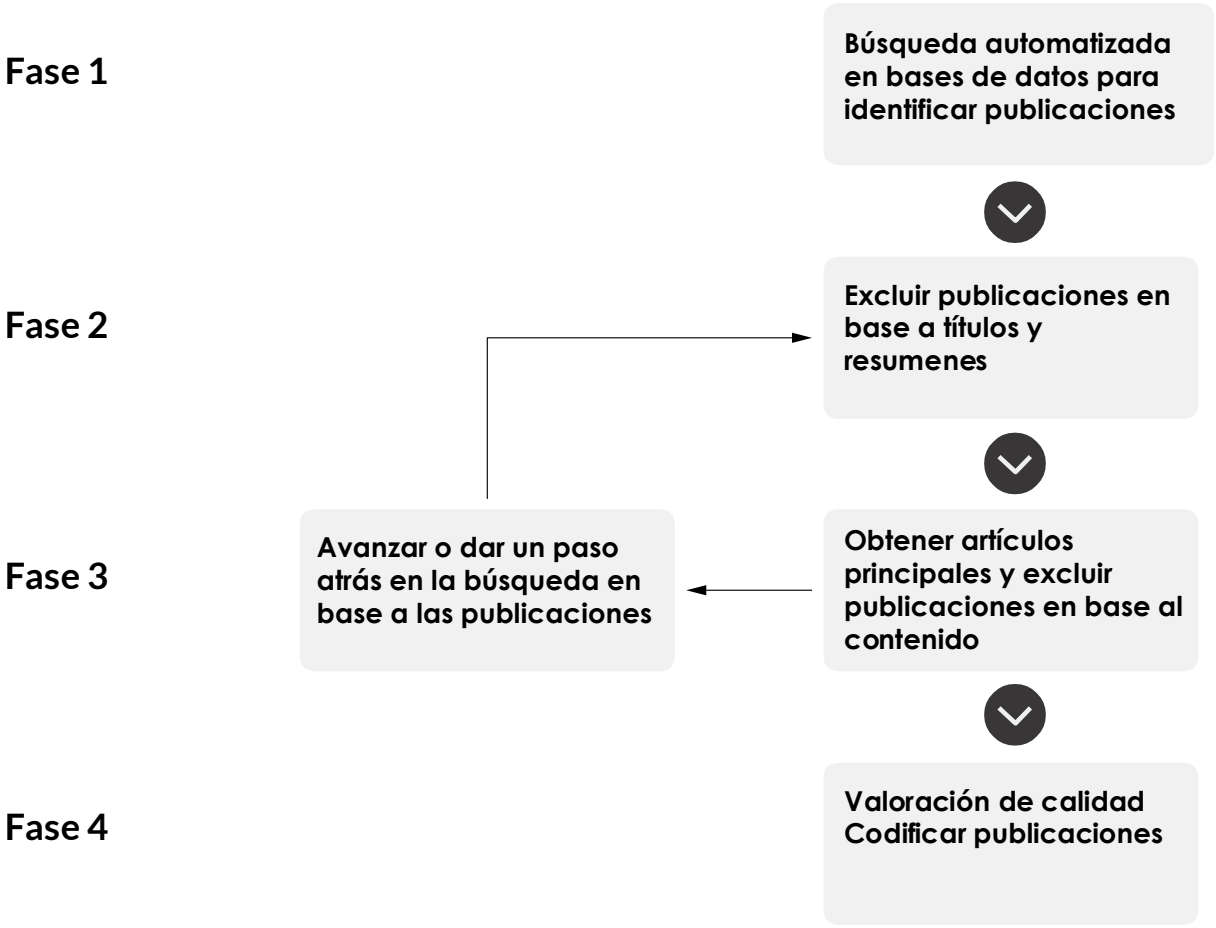
Con el objetivo de mostrar la evolución de las metodologías, en este artículo se presenta una revisión de métodos propuestos en los últimos años. El análisis se basa en una comparación cronológica y una clasificación en base a los criterios de perspectiva, enfoque, fase del proceso a la que se dirigen y su aplicación para casos reales.

A continuación, se expone el método empleado para la selección de las metodologías, después en la siguiente sección se hace una breve explicación de cada metodología, se explican los criterios de comparación, se muestra la tabla comparativa y por último se indican las conclusiones obtenidas tras la revisión.

2. Método

La revisión de metodologías para su posterior análisis comparativo se ha basado principalmente en buscar metodologías referentes en los últimos años en el ámbito del desarrollo de software. El criterio en el que se ha fundamentado la selección ha seguido un proceso definido por Brhel et al. (2015) en el que se definen cuatro fases (Fig. 1).

Fig. 1: Fases del método de búsqueda



En la primera fase, se ha realizado una búsqueda en la base de datos Google Scholar (2017), para ello se ha empleado el término de búsqueda “software development methods”. En la segunda fase, basándose en los títulos y resúmenes se han identificado posibles publicaciones de interés y se han excluido las consideradas como no relevantes para que en una tercera fase, se analizara el contenido. En este paso, se ha decidido si el contenido era adecuado para su inclusión en la revisión y si disponía de carácter relevante en el ámbito del desarrollo de software. En el caso de no considerarlo apto para la inclusión, se ha retrocedido al paso anterior. Finalmente, con las metodologías seleccionadas se ha procedido a realizar la revisión, el cual se ha mostrado en la tabla comparativa. A continuación, se listan las metodologías seleccionadas con un breve resumen de sus contenidos.

3. Metodologías

Metodologías analizadas y listadas en orden cronológico y breve explicación de sus contenidos:

01. Jackson System Development (JSD) method. (Jackson, M.A., 1982)

La metodología JSD se basa en un proceso secuencial y no iterativa, definida con pasos fijos en una estructura de árbol. Se fundamenta en los siguientes pasos: análisis de la entidad y la acción, análisis de la estructura de la entidad, modelo inicial, funciones del sistema, momentos del sistema y la implementación.

02. Structured Systems Analysis and Design Method (SSADM). (Ashworth, C. M., 1988.)

Es una metodología "Waterfall" basada en un proceso secuencial (no iterativa), en el que no se avanza a siguientes fases antes de finalizar las previas. Se divide en las siguientes fases: análisis de viabilidad, investigación del entorno, análisis del sistema de la empresa, definición de requerimientos, opciones técnicas del sistema, diseño de la lógica y diseño físico.

03. Agile software development: the business of innovation. (Highsmith, J. ; Cockburn, A., 2001)

Se basa en proveer un proceso dinámico caracterizado por ciclos iterativos y la participación puntual de agentes externos, de modo que, el Producto Mínimo Viable (PMV) se diseña lo antes posible y se va mejorando con las continuas evaluaciones.

04. Key principles for user-centred systems design. (Gulliksen, J., 2003)

User-Centered Systems Design (UCSD) es un proceso enfocado en la usabilidad a lo largo de todo el proceso de desarrollo y en todo el ciclo de vida del sistema. Se basa en los siguientes principios: centrado en las necesidades del usuario y sus acciones, participación activa de los usuarios, desarrollo iterativo e incremental del sistema, prototipos desde fases iniciales, evaluar el uso en el contexto real y un equipo multidisciplinar con una visión holística del proceso.

05. Usability engineering methods for software developers. (Holzinger, A., 2005)

El método se enfoca en trabajar la usabilidad del producto durante todo el proceso. Define la usabilidad como la facilidad de uso y aceptabilidad del sistema para ciertos usuarios llevando a cabo tareas específicas en un entorno específico, enfocándose tanto en la calidad del producto como la satisfacción del usuario. Además, propone diferentes técnicas de usabilidad para cada fase del proyecto.

06. Towards a Framework for Integrating Agile Development and User-Centred Design.

(Chamberlain, S.; Sharp, H., 2006)

Es un marco teórico donde propone la unión de las metodologías Desarrollo de Software Ágil (DSA) y Diseño Centrado en el Usuario (DCU). Una vez analizadas sus similitudes y diferencias, define los siguientes principios como base para una adecuada unión: integrar al usuario en el proceso de desarrollo, colaboración cercana entre los diseñadores y desarrolladores, crear prototipos cuanto antes para tangibilizar las ideas, ofrecer tiempo necesario a fases previas de exploración y gestionar el proyecto de forma cohesionada.

07. U-SCRUM: An agile methodology for promoting usability. (Singh, M., 2008)

Propone una metodología basada en la integración de los principios de usabilidad y de SCRUM. La denominada metodología SCRUM se fundamenta en los marcos de desarrollo ágiles, con procesos incrementales e iterativos. Sin embargo, no plantea los términos de la usabilidad, por ello, Singh propone una metodología que une las dos partes e incorpora los conceptos de usabilidad durante todo el proceso.

08. Aesthetics and experience-centered design. (Wright, P. ; Mccarthy, J., 2008)

Una metodología centrada en diseñar la experiencia de uso del usuario al interactuar con el producto. Propone un marco teórico donde hace especial hincapié en la fase inicial del proceso, donde se identifica mediante la participación de los usuarios cuáles son sus necesidades y motivaciones, para que mediante historias de usuario se diseñe la experiencia que tendrán en su uso.

09. User eXperience Design and Agile Development: From Theory to Practice. (Silva, T., 2012)

Propone la unión de las metodologías Diseño de Experiencia de Usuario y Desarrollo de Software Ágil. Para ello, incluye en el proceso iterativo ágil un paso previo, llamado iteración 0, donde se aplican los conceptos de experiencia de usuario y diseño centrado en el usuario para identificar las necesidades de los usuarios. Además de eso, presenta diferentes técnicas para la integración de conceptos de usabilidad durante todo el proceso.

10. Context-aware systems. (Fischer, G., 2012)

Es una metodología que dicta como factor transcendental el contexto que rodea al sistema, y tiene como objetivo ofrecer un producto que asista a las personas para aumentar su conocimiento, productividad y creatividad. De este modo, se consigue proveer un sistema que da la información correcta, en el momento correcto, en el lugar correcto, en el modo correcto y a la persona correcta.

11. Agile Usability Patterns for UCD early stages. (Bertholdo, A., 2014)

Una metodología centrada en las fases iniciales del proyecto, basado en un proceso que combina las características del DSA y DCU e introduce los conceptos de la usabilidad durante todas las fases. Define en concreto tres técnicas para las fases iniciales: identificar las necesidades de los usuarios mediante las herramientas del DCU, especificar y analizar el contexto de uso del sistema y definir los requerimientos técnicos.

12. Metodologías ágiles centradas en personas para desarrollar software educativo. (González, C., 2015)

Es un modo de trabajo que engloba las metodologías de DSA, DCU, Experiencia de Usuario y Lean UX. Reúne conceptos de todos ellos, por lo que plantea un proceso basado en el “design thinking”, donde propone herramientas para analizar el usuario, el contexto y las acciones, para que mediante un desarrollo ágil se consiga diseñar un sistema que aporta una experiencia positiva en el usuario.

13. Creating people-aware IoT applications by combining design thinking and user-centered design methods. (Fauquex, M., 2015)

Una metodología basada en el design thinking y experiencia de usuario para crear aplicaciones centradas en el usuario. Alega que el desarrollo se conduce en base a las necesidades del usuario y no en las oportunidades técnicas. Esta metodología engloba todas las fases del proceso y propone herramientas para cada una de ellas. Define las siguientes fases: exploración, identificación de requisitos, análisis, diseño, prototipo, evaluación y redefinición.

14. Empowering user interfaces for industrie 4.0. (Pfeiffer, T., 2016)

Expone un modo de trabajo centrado en crear sistemas que mediante conceptos extraídos del DCU, DSA y la usabilidad, sean capaces de facultar y fortalecer las capacidades del usuario para conseguir aumentar el control y productividad de las máquinas de la industria 4.0. Expone tres herramientas concretas para ello: Personas, Storyboards y el uso del eye-tracking para observaciones del comportamiento del usuario.

15. A conceptual UX-aware Model of Requirements. (Kashfi, P., 2016)

Expone un modo de trabajo que une los conceptos de la experiencia de usuario y la ingeniería del software. Distingue la experiencia de usuario de los conceptos únicamente de usabilidad, ya que, engloba también los aspectos emocionales de los usuarios. Define tres niveles principales en el proceso de desarrollo: requerimientos de experiencia de usuario (necesidades emocionales del usuario), objetivos QR (requerimientos de calidad) y objetivos FR (requerimientos funcionales).

16. User experience methodology: from the physical to the emotional. (Priestley, O., 2016)

Una metodología de diseño para desarrollar sistemas que crean experiencias. Propone que el centro de un proceso de desarrollo software debe ser la persona y trabaja dos fases principales: la generación y la evaluación. En el primero de ellos, se identifica cuáles son las motivaciones del usuario y sus necesidades tanto funcionales como emocionales. En el segundo, plantea una evaluación basada en la usabilidad, pero con carácter cualitativo y no cuantitativo.

17. Spiral UX Design Model. (Guo, H., 2016)

Es la adaptación de la experiencia de usuario al modelo espiral de desarrollo de software. Define un proceso iterativo que comienza con la identificación de las necesidades de usuario. Después para las siguientes fases propone un modelo espiral donde cada paso de avance supone un aumento del grado de fidelidad del concepto, partiendo desde bocetos de diseño hasta el producto final.

4. Criterios de clasificación

Una vez descritas las metodologías revisadas, en esta fase se detallan los 4 criterios empleados para la posterior comparativa: enfoque, perspectiva, fase del proceso y aplicación.

4.1. Enfoque

Se refiere al modo en el que el autor cree que se debería abordar un proceso de desarrollo de software. Se han dividido cuatro enfoques diferentes:

- Producto: se centra únicamente en el propio software, propone una metodología para crear sistemas más eficientes y efectivos.

- Producto-usuario: plantea que el proceso debe partir de las necesidades del usuario, y que su cumplimiento determina el éxito del producto. Además de conceptos de diseño centrado en el usuario propone técnicas de desarrollo software.

- Producto-usuario-contexto: propone un modo de trabajo en el que es necesario contemplar todo el entorno del sistema, analizando el producto, el usuario que lo utiliza y el contexto en el que se realiza la interacción.

- Experiencia: defiende que al desarrollar un nuevo software se debe diseñar su experiencia de uso, cumpliendo con las motivaciones y necesidades funcionales y emocionales de los usuarios.

4.2. Perspectiva

El criterio de la perspectiva se refiere al punto de vista de los miembros del equipo de trabajo que plantea el autor, la cual se determina por la(s) disciplina(s) de los miembros del equipo. Se han diferenciado tres perspectivas:

- Software Development Process (SDP): el equipo de trabajo compuesto por desarrolladores de software.

- SDP-Diseño: equipo multidisciplinar compuesto por desarrolladores y diseñadores.

- Diseño: el equipo formado por diseñadores.

4.3. Fase de proceso

En este criterio se han definido las fases del proceso en las que hace referencia el método de trabajo. Las fases están definidas en base a la metodología de DBZ de Mondragon Unibertsitatea (DBZ, 2014):

- Estratégico: identificar oportunidades para nuevos productos y servicios a través del análisis del contexto interno y externo de la empresa.

- Exploración: en base a la oportunidad detectada, definir las especificaciones que el producto/servicio tiene que considerar para satisfacer las necesidades del cliente/usuario.

- Diseño: generar nuevos conceptos de producto/servicio acordes con las especificaciones recogidas en el Brief de Diseño.

- Desarrollo: Diseñar y desarrollar en detalle el concepto de producto/servicio seleccionado para la obtención de un prototipo funcional.

- Lanzamiento: insertar y promocionar el nuevo producto/servicio en el mercado y recoger información de su rendimiento para la identificación de mejoras.

4.4. Aplicación

Trata de clasificar las metodologías en base a la aplicabilidad de las mismas. Se han definido tres aplicaciones:

- Framework: marco teórico donde expone conceptos de un modo de trabajo.

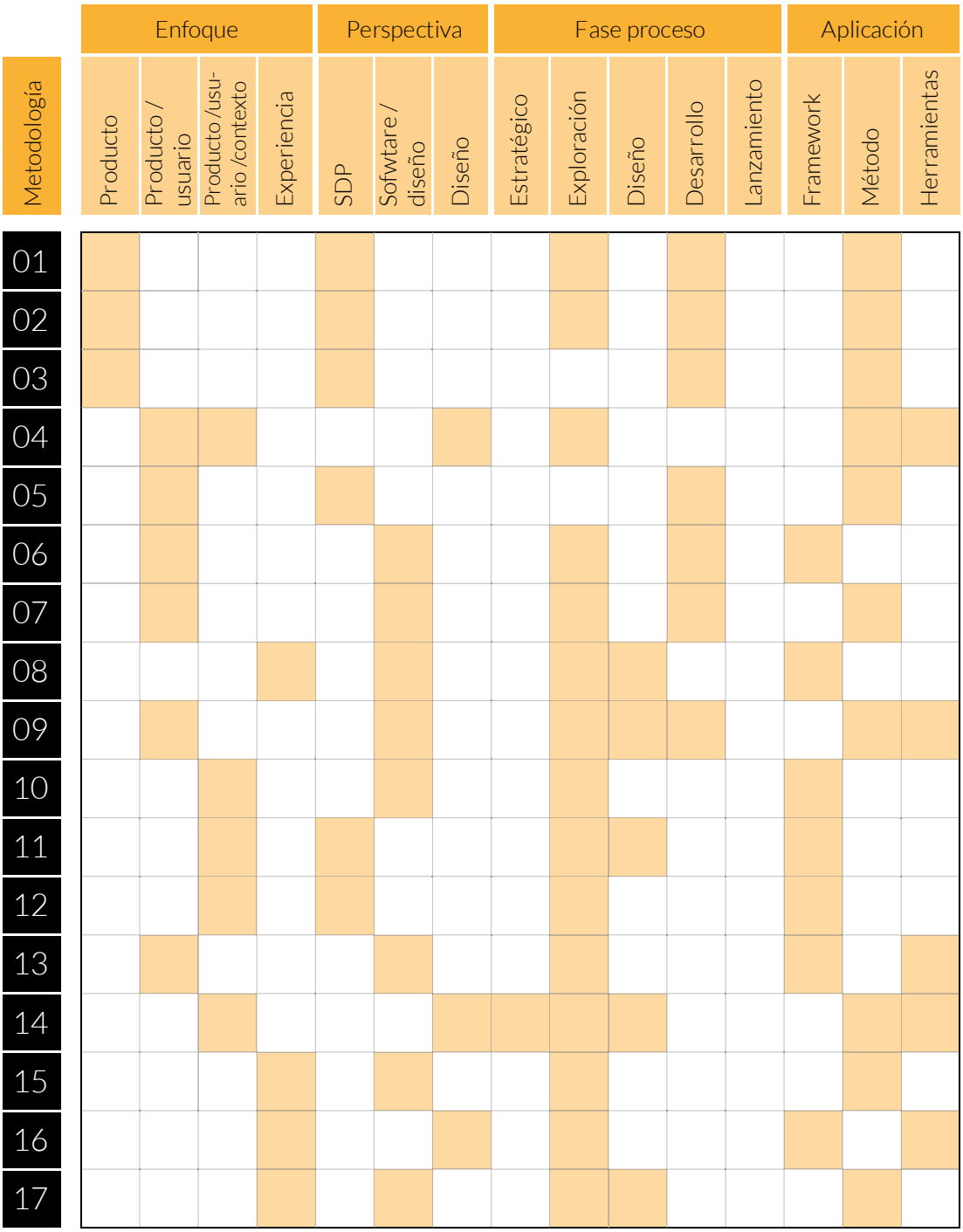
- Método: propone un proceso concreto con pasos y fases a realizar para poder aplicar lo expuesto de forma teórica.

- Herramientas: plantea herramientas concretas de forma detallada para su aplicación directa en casos reales.

5. Nueva clasificación de metodologías de desarrollo software

En esta nueva clasificación se han reunido en una tabla las metodologías mencionadas para compararlas con los criterios definidos (Fig. 2). En el eje vertical se han situado las metodologías, cronológicamente en el orden en el que se han expuesto en el apartado 3. En el eje horizontal, se han ubicado los criterios, en el mismo orden definido en el apartado 4.

Fig. 2: Nueva clasificación de metodologías de desarrollo software



Mediante el análisis de las metodologías en base a los criterios definidos se han obtenido diferentes resultados. Por un lado, se observa que hay una evolución de los desarrollos de software hacía la experiencia de usuario. En un principio, se centraban únicamente en el desarrollo del propio software, por lo que el objetivo era lograr un producto óptimo en la funcionalidad. Sin embargo, con los años se fue introduciendo la idea de considerar al usuario en el proceso, principalmente identificando sus necesidades con el producto y diseñándolos partiendo de ellos. Como consecuencia, se crearon muchas metodologías basadas en la unión de métodos propios de desarrollos de software, como el Agile Software Development (ASD) y las de diseño como el User Centered Design (UCD). Posteriormente,

además de centrarse en el usuario, se empezó a introducir el término del contexto, analizando así la interacción entre máquina y usuario de una forma más completa.

Estas metodologías centradas en el usuario, han tomado gran importancia y hoy en día muchas empresas lo han integrado para sus desarrollos. A pesar de ello, el diseño de interfaces sigue evolucionando y en los últimos años ha empezado a surgir el término experiencia de usuario. Trata de diseñar la experiencia que tendrá el usuario a la hora de utilizar el producto y se considera el eslabón que mayor vínculo genera entre el producto y el ser humano, el lugar donde mayor presencia tienen las emociones humanas. Por lo tanto, no solo se centra en las necesidades pragmáticas que pueda tener el usuario, como en el UCD, sino que también se centra en las necesidades emocionales y en las motivaciones que quiere cumplir en el uso del producto, para así conseguir un sistema que aporte emociones positivas que aumentan la implicación de las personas (Isen, 2001), facilitando los procesos de aprendizaje (Kort y Reilly, 2002). Además, toma en cuenta todo lo que pueda influir en la experiencia del usuario, como el contexto, el momento de uso, las acciones a realizar y los medios que utilizará para ello.

Por otro lado, se puede observar que la perspectiva del diseño está cada vez más presente en las metodologías de desarrollo de software. Lo que demuestra que el ámbito de la informática y el diseño están cada vez más cerca y que los proyectos de equipos multidisciplinares son cada vez más habituales.

Asimismo, al analizar en qué fases de desarrollo se centra cada metodología, se puede observar que las más antiguas, las que se centran únicamente en la funcionalidad del producto, hablan sobre todo de la fase de desarrollo. En cuanto a las metodologías centradas en el usuario y la experiencia, en su mayoría se focalizan solamente en la fase de exploración, donde analizan al usuario y demás factores que puedan influir para después realizar tanto el diseño como el desarrollo. De modo que se observa una carencia de metodologías que tomen en cuenta todo el proceso.

Para finalizar, el último criterio indica que estas metodologías tienen aplicaciones diferentes, pero sí se observa que en los últimos años hay más metodologías enfocadas en el usuario que muestran fases definidas de trabajo y herramientas que permitan realizar lo que proponen.

6. Conclusiones y líneas futuras

Tras la revisión realizada y como conclusión final, se ha procedido a la propuesta de las características idóneas que debería cumplir hoy en día una metodología de diseño y desarrollo de interfaces. En primer lugar, la metodología debería tener el enfoque de la experiencia de usuario, ya que trabaja aspectos que favorecen el desarrollo de usuarios más activos, críticos, participativos, motivados y con un mayor vínculo afectivo respecto a lo que les rodea. En segundo lugar, debería visualizar el proyecto desde una perspectiva de unión entre software y diseño, formando un equipo multidisciplinar que aporte opiniones de experto en ambos campos de conocimiento y puntos de vista diferentes y constructivos que evitan grandes rediseños y optimizan la duración final del proyecto. En tercer lugar, sería adecuado que contemplara todas las fases del proceso, desde una fase de exploración de oportunidades hasta la fase final de lanzamiento, para así conseguir un producto más coherente y completo. Por último, la metodología a desarrollar debería ser capaz de mostrar su aplicación para casos reales, ofreciendo una guía estructurada que defina el método, las fases del proceso y los pasos a dar acompañado con las herramientas a emplear en cada una de ellas.

7. Bibliografia

Ashworth, C. M. (1988). Structured systems analysis and design method (SSADM). *Information and Software Technology*, *30*(3), 153-163. doi: 10.1016/0950-5849(88)90062-6

Bertholdo, A. P. O., da Silva, T. S., Melo, C. D. O., Kon, F., & Silveira, M. S. (2014, June). Agile usability patterns for UCD early stages. In *International Conference of Design, User Experience, and Usability* (pp. 33-44). Springer International Publishing. doi: 10.1007/978-3-319-07668-3_4

Brhel, M., Meth, H., Maedche, A., & Werder, K. (2015). Exploring principles of user-centered agile software development: A literature review. *Information and Software Technology*, *61*, 163-181. doi: 10.1016/j.infsof.2015.01.004

Brinkkemper, S. (1996). Method engineering: engineering of information systems development methods and tools. *Information and software technology*, *38*(4), 275-280. doi: 10.1016/0950-5849(95)01059-9

Chamberlain, S., Sharp, H., & Maiden, N. (2006, June). Towards a framework for integrating agile development and user-centred design. In *International Conference on Extreme Programming and Agile Processes in Software Engineering* (pp. 143-153). Springer Berlin Heidelberg. doi: 10.1007/11774129_15

Da Silva, T. S., Silveira, M. S., Maurer, F., & Hellmann, T. (2012). User experience design and agile development: From theory to practice. *Journal of Software Engineering and Applications*, *5*(10), 743. doi: 10.4236/jsea.2012.510087

Fauquex, M., Goyal, S., Evequoz, F., & Bocchi, Y. (2015, December). Creating people-aware IoT applications by combining design thinking and user-centered design methods. In *Internet of Things (WF-IoT), 2015 IEEE 2nd World Forum on* (pp. 57-62). IEEE. doi: 10.1109/WF-IoT.2015.7389027

Fischer, G. (2012, May). Context-aware systems: the'right'information, at the'right'time, in the'right'place, in the'right'way, to the'right'person. In *Proceedings of the international working conference on advanced visual interfaces* (pp. 287-294). ACM. doi: https://doi.org/10.1145/2254556.2254611

Fitzgerald, B. (1998). An empirical investigation into the adoption of systems development methodologies. *Information & Management*, *34*(6), 317-328. doi: 10.1016/S0378-7206(98)00072-X

González-González, C. S., Toledo-Delgado, P., & Muñoz-Cruz, V. (2015). Agile human centered methodologies to develop educational software. *Dyna*, *82*(193), 187-194. doi: 10.15446/dyna.v82n193.53495

Gulliksen, J., Göransson, B., Boivie, I., Blomkvist, S., Persson, J., & Cajander, Å. (2003). Key principles for user-centred systems design. *Behaviour and Information Technology*, *22*(6), 397-409. doi: 10.1080/01449290310001624329

Guo, H. (2016, July). Lean but not Mean UX: Towards a Spiral UX Design Model. In *International Conference of Design, User Experience, and Usability*(pp. 25-33). Springer International Publishing. doi: 10.1007/978-3-319-40409-7_3

Highsmith, J., & Cockburn, A. (2001). Agile software development: The business of innovation. *Computer*, *34*(9), 120-127. doi: 10.1109/2.947100

Holzinger, A. (2005). Usability engineering methods for software developers. *Communications of the ACM*, *48*(1), 71-74. doi: 10.1145/1039539.1039541

Isen, A. M. (2001). An influence of positive affect on decision making in complex situations: Theoretical issues with practical implications. *Journal of consumer psychology*, *11*(2), 75-85. doi: 10.1207/S15327663JCP1102_01

Jackson, M. A. (1982) *Tools and notions for program construction: An advanced course*; pages 1-25; Cambridge University Press.

Kashfi, P., Feldt, R., Nilsson, A., & Svensson, R. B. (2016, August). A Conceptual UX-Aware Model of Requirements. In *International Conference on Human-Centred Software Engineering* (pp. 234-245). Springer International Publishing. doi: 10.1007/978-3-319-44902-9_15

Kort, B., & Reilly, R. (2002). Theories for deep change in affect-sensitive cognitive machines: A constructivist model. *Educational Technology & Society*, *5*(4), 56-63. Obtenido de http://www.jstor.org/stable/jeductechsoci.5.4.56

Lehman, M. M., Ramil, J. F., Wernick, P. D., Perry, D. E., & Turski, W. M. (1997, November). Metrics and laws of software evolution-the nineties view. In *Software Metrics Symposium, 1997. Proceedings., Fourth International* (pp. 20-32). IEEE. doi: 10.1109/METRIC.1997.637156

Mens, T., Wermelinger, M., Ducasse, S., Demeyer, S., Hirschfeld, R., & Jazayeri, M. (2005, September). Challenges in software evolution. In *Principles of Software Evolution, Eighth International Workshop on* (pp. 13-22). IEEE. doi: 10.1109/IWPSE.2005.7

Pfeiffer, T., Hellmers, J., Schön, E. M., & Thomaschewski, J. (2016). Empowering User Interfaces for Industrie 4.0. *Proceedings of the IEEE*, *104*(5), 986-996. doi: 10.1109/JPROC.2015.2508640

Priestley, O. (2015). User experience methodology: from the physical to the emotional. *Learned Publishing*, *28*(4), 317-320. doi: 10.1087/20150412

Singh, M. (2008, August). U-SCRUM: An agile methodology for promoting usability. In *Agile, 2008. AGILE'08. Conference* (pp. 555-560). IEEE. doi: 10.1109/Agile.2008.33

Unibertsitatea, Mondragon. *Metodología de Innovación Centrada en las Personas*. 2014.

Wright, P., & McCarthy, J. (2010). Experience-centered design: designers, users, and communities in dialogue. *Synthesis Lectures on Human-Centered Informatics*, *3*(1), 1-123. doi: 10.2200/S00229ED1V01Y201003HCI009