Research Article

Design of an Agile Methodology oriented to the development of software in dissertation projects

Gustavo Rivera[1], Fredys Simanca[2], Fabian Blanco[2], Alexandra Abuchar[3], Miguel Hernandez[4]

[1]Universidad Autónoma de Colombia, Bogotá, BOG 110931 Colombia
[2]Universidad Cooperativa de Colombia, Bogotá, BOG 110931 Colombia
[3]Universidad Distrital Francisco José de Caldas, Bogotá, BOG 110931 Colombia
[4]Fundación Universitaria Los Libertadores, Bogotá, BOG 110931 Colombia

Corresponding author: Fredys Simanca (e-mail: fredys.simanca@campusucc.edu.co).

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

ABSTRACT:
The referred methodology is the result of a consistent research with the problems evidenced in the students' dissertations, who, when approaching software development projects, agree on the selection of agile methodologies inspired by the benefits evidenced by the market trend; finding themselves with the disjunctive of traversing academic spaces with substantial differences in relation to the business environment where the methodologies are generally aimed at. The conceptual framework of the proposed methodology is defined within the Software Engineering discipline, also related to the precepts attentive to agile methodologies and the academic approach to applicability. It has been experimented through case studies in the development of curricula of Higher Education in Engineering programs, where successful results have been observed. The most relevant characteristics of the methodology are based on the systemic approach of context, simplicity and ease of learning, and methodical components such as: iterative, incremental, collaborative work and adaptability to academic environments.

KEYWORDS: Agile methodology; software project; dissertation.

1. INTRODUCTION

The Higher Education Institutions - HEIs -, protected by university autonomy, accept and regulate the option of dissertation as a component of the academic program to be fulfilled as a requirement to qualify for the respective professional degree. Understanding the dissertation, as a demonstrative academic exercise of appropriation, application and prospection of knowledge, to address and ideally solve problems related to the disciplinary performance, in order to assess the skills consistent with the comprehensive professional training, in a scenario of compliance with institutional regulations [1].

A modality exposed to the fulfillment of the dissertation in the undergraduate programs in Engineering is covered by the software projects, where the need to go through Software Engineering (SE) is demanded, and from it the precepts related to the models and methodologies for software development. This is the moment when the dilemma of deciding between the two dominant currents of the market arises: to accept, by characterization, the agile methodologies or the so-called traditional ones [2].

It is advisable not to become isolated from the generalized recognition of agile methodologies due to their preferential use in the global market [3], for its benefits in synthesis presumed by: promoting incremental and iterative development, available to small work teams, of unobjectionable flexibility in the arrangement and evolution of processes, and proposal of short time units to obtain resulting products; where the participation of the client in the process is noticed.

Traditional methodologies fundamentally oriented to products, proclaim sequential methodical rigor, process control aligned to exhaustive planning and documentary support, in compliance with prerequisite connotation stages, typically understood as: requirements, design, development and testing [4], without ignoring what has been said by [5], regarding the emergence of a third hybrid alternative, made up of the best practices and artifacts of the two methodical paradigms, where flexibility is combined with the property of robustness.

The academic space where software projects are enacted, warns relevant particularities determined by the lack of expertise of students in the approach of the engineering assignment coupled with the magnitude of the academic responsibility. Difficulties are evidenced concerning the selection of the methodology and regarding the designation of roles, the acquisition and administration of resources, and the management of time in compliance with the acquired commitments; a problem not available in business environments due to the ductility and orientation of the methodologies.

The selection of the methodology in the dissertations, shows an evident degree of uncertainty in the procedural compliance, participating and reflecting in the indicators of university desertion or the dilation of the process with academic consequences focused on the students [6] [7]. This situation supports and justifies the constitution of a methodology that, in compliance with the vision of the dissertations, allows the adoption of an engineering practice participating in the support of disciplinary competences, without losing the context of the professional activity.

The Agile Methodology oriented to disertations (MaTraGra – for Método del Trabajo de Grado), which is being proposed in this document is conceived to be applied in software projects immersed in dissertations, attending the peculiarities of the academic spaces and the normative designs regulated by the HEIs, considering the spectrum promulgated by the agile methodologies, with the disciplinary inspiration sponsored by the SI and the contextual scope promoted by the systemic approach.

Consistent with MaTraGra's guiding profile, it is potentially supported by the constituent and universally accepted principles of agile methodologies [8], subscribing to the following characteristics of relevance: incremental product development in iterative processes; determination of roles consistent with the participants in the research; sponsoring collaborative work; flexibility willing to rethink; and proposing moments of evaluation and feedback [9]. Likewise, the methodical disposition biased to learning precepts referring to the ease of appropriation, comprehension and application.

The purpose of the study is emphasized in the presentation of the MaTraGra methodology from the constitutive conceptual foundation, the methodical components that participate in the functional and operative structure, the precepts that are involved in the moments or phases enacted and the approaches resulting from the research experience around the methodology by way of conclusions. Finally, the referential disposition of the conceptualization that supports the theoretical and thematic foundation.

1.1 Theoretical foundations supporting the MaTraGra Methodology

The guiding profile of the MaTraGra methodology is preferably directed to the development of software projects contemplated in engineering dissertations, nourished by the disciplinary postulates of the SE, aligned to the precepts promoted and universally accepted on agile methodologies and contextualized by the systemic approach to address the problem from an integral perspective, recognizing the approach to human resources.

The HEIs welcome and present the dissertation as a terminal component in compliance with the syllabus attached to the academic programs, aimed at evidencing and valuing the professional disciplinary competences through an integral academic practice, for the recognition of knowledge and suitability in accordance with the curricular designs. It is referred to by [10] as the result of a creative and innovative, objective, reflective, controlled and critical process, consistent with a social problem or focused on transcending the frontiers of knowledge.

Promoted by the autonomy of the university, the power is granted to the HEIs to issue the regulatory guidelines for the approach of the dissertations. In accordance with the discretionary aptitude of the student assigned to the Faculty of Engineering, the student chooses the approach of software projects directed to the solution of problems from the engineering perspective, which comply with the institutional regulatory designations.

Software projects, referred to by [11] as the application of a systematic, disciplined and quantifiable approach to the software development process, framed by the postulates of Software Engineering, leading to the dilemma of adopting the models and methodologies of the spectrum contemplated in the market, presented in three categories: agile, traditional and hybrid methodologies.

A methodology for the development of software, designed to meet the commitments acquired by the software projects, is structured in appropriately defined stages [12]. According to [13], These stages evolve in accordance with the demands of Information Technologies (IT), determining higher levels of efficiency reflected in productivity, by adopting techniques, tools and other artifacts for the management and development of software.

Traditional or heavy methodologies are recognized by the preferential focus on products, the exhaustive control exercised over the processes, the postulation of stages or phases of sequence compliance that respond to a complex and strict planning [14], accompanied concurrently by supporting documentation; the phases of requirements identification and scope definition, product design and work planning, development process and testing with user participation are generalized in them [15].

Agile methodologies are based on the expertise of the participants, where the responsibility is assigned to meet the objectives, sponsoring flexibility in the process, iterative and incremental development, involving short-term results [16], coupled with the feasibility of coupling to different environments for the development of software projects. Reasons that warn the greater demand in the use, obtaining better results associated with productivity, quality and compliance; by the management to the change of priorities, project visibility, and alignment of IT and business according to the report of the agility state of [3].

Differential aspects between traditional and agile methodologies, presented by [17]: are highlighted: overlapping heuristics with standards without responding to a rigorous normative control, obtaining products in less time, continuous feedback exposed to frequent changes, role diversity, client participation in the work team, non-rigid software architecture and flexible contracting processes.

Agile methodologies respond, in their affinity, to the principles promoted in the Agile Manifesto [8], considered the main historical fact that promoted the emergence of this engineering current, postulating in its principles the prioritization of people over products, collaborative work and response to change supported by flexibility. Agile methodologies have evolved by following paths directed to project management where quick responses and predictive analysis are convenient to minimize uncertainty.

The systemic approach provides the integral and contextual vision of the system for its study, from the engineering perspective calls it mental software [18], welcoming it for the identification of the structure of the object of study, assimilated in the quality of the design and reflected in the development of the product, also, in the selection of appropriate tools for the analysis of the complexity and consequently the conception of systems methodologies; assertion complemented by [19], when stating that it can be used in the construction of administrative models of gradual and evolutionary construction.

The Dimensional Analytical Tool 4-DAT is presented by [20], for the analytical measurement of the level of agility and adaptability of the methodologies in software development processes from four dimensions: characterization of the scope allowed by the methodology; degree of agility in the application; agile values in compliance with the Agile Manifesto, maintainability and profitability; and software processes aligned to the software life cycle.

The Technology Acceptance Model (TAM), presented by [21], proposes the acceptance of a technological system dependent on the user's attitude, directly affected by the functionality and capacity of the system, emphasizing the motivational factor from three elements: perceived usability, perceived usefulness and consequent attitude. Utility is conceived as being related to productivity and usability to the effort required for the use of the technology.

For the experimental development of the methodology, based on what was advocated by [22], the following six stages were carried out: Exploratory, planning, scenario entry, analytical phase, case study development and informative phase.

2. THE MATRAGRA METHODOLOGY

The orientation of the methodology adheres to the particularities revealed by undergraduate engineering dissertations, where software projects are presented as an academic space of integral engineering and disciplinary practice to demonstrate the competencies of professional training, applying knowledge of professional connotation in academic environments with differentiating particularities of the business or organizational world.

It is justified on the basis of the problem instituted by the difficulty of assimilating the methodologies accepted in the market and promulgated by the SE, in environments determined by the dissertations with specific differential aspects of the business world where the application profile of the methodologies is aimed.

The systemic approach supports the MaTraGra methodology covering contextual aspects from three perspectives: a holistic component, associated with the purpose of providing a methodical framework to address the object of study in its systemic dimension and reach feasible solutions to problems; secondly, the heuristic component, from the conception of the evolutionary methodology in a space of time outlining the applicability and functionality; finally, the ethical and aesthetic component, responds to the reason, intentionality and projection of the procedures and solutions.

Supporting the competitiveness processes of the organizations, based on the use of IT and the agile production of quality software, concurrently the need of having a human resource according to these movements is enhanced, the academy is involved in the responsibility of covering the spaces of professional training through the conception of relevant curricula [23] [24].

As stated by [25], the methodology is supported by two categories: the philosophical approach, which responds to the affinity to agile methodologies; and the guidelines for the development of software projects, participating in the processes of analysis, design, continuous communication and feedback.

In the methodology, four referential knowledge converge (See Fig. 1), which propose the sustainable conceptual disposition: the dissertation, which determines the application profile; Agile Manifesto, which provides the fundamentals of adherence to agile methodologies; Software Engineering, regarding the disciplinary methodological inputs for the development of software projects; and the Systemic Approach, which provides the contextual and integral spectrum.

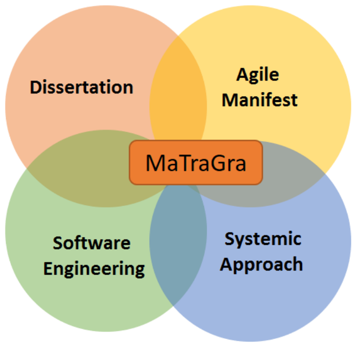


Fig. 1. MaTraGra referents. Source: Own elaboration

According to [26] [27], the methodology is made up of procedures, techniques, tools and documents, as well as phases and sub-phases that guide the process and enable planning, management, control and evaluation. Consequently, the components of the methodology are presented.

2.1 Roles

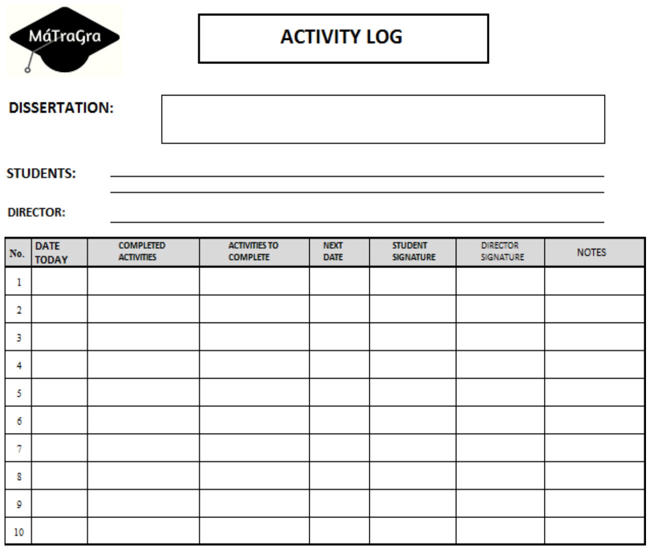
The human resource advocated by the methodology responds to the characterization of the dissertation, where collaborative and team work is promoted, where agile software development is focused [8]: in small and stable development groups [28]. Consequently, two types of roles are identified:

- *Student*: This role assumes the maximum academic responsibility for the process in compliance with the commitments acquired in the preliminaries of the project.
- *Director*: assumes the role of validator of the process and products, certifying their quality.

2.2 Documentation

For [25], documentation is important for review and correction purposes, in support of system usability, as well as for diary management and activity logging, cited by [29]. In agile software development by iterations or phases, according to [28], Formal documents participate in the communication processes in compliance with requirements. The methodology proposes two categories of documents to be elaborated concurrently with the development of the dissertation:

- Activity log: is constituted as an agenda for the registration of commitments and their fulfillment (See Figure 2), with the responsibility of the director.
- Engineering document: sustainable support of the academic engineering practice in compliance with the regulations of the institution. The student is responsible for its development and with the validation by the director.



ACTIVITY LOG

DISSERTATION:

STUDENTS:

DIRECTOR:

No.	DATE TODAY	COMPLETED ACTIVITIES	ACTIVITIES TO COMPLETE	NEXT DATE	STUDENT SIGNATURE	DIRECTOR SIGNATURE	NOTES
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

Fig. 2. Activity log. Source: Own elaboration

2.3 MaTraGra structural processes

The methodology is composed of phases that allow iteration and flexibility, determined by the moments of validations, see Figure 3.

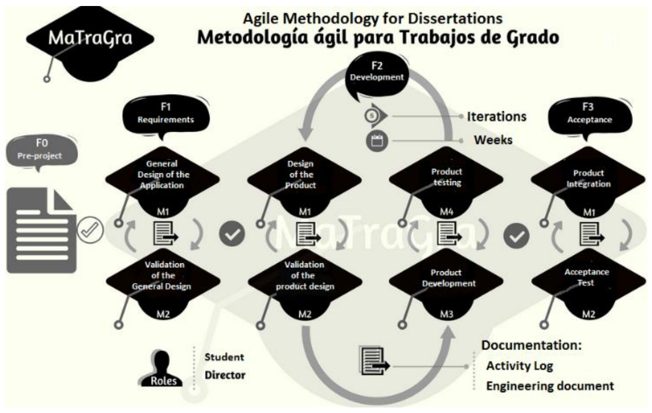


Fig. 3. MaTraGra Methodology. Source: Own elaboration

***Phase 0. Pre-project**:* fundamental prerequisite to approach the development of the methodology, where the project is conceived. Exposed by [30], as the document in which the idea that constitutes the core of the problem of the dissertation is identified and specified.

Phase 1. Requirements: where the identification of requirements that characterize the quality of the products is promulgated [31]; to determine the behavior, properties and restrictions of the software to be built [32]. It consists of two moments:

- *Moment 1. General design of the application*: agile design of the overall system architecture [28], platform specifications, development tools, data and testing. Also, the functional and operational planning of the work. Moment documented concurrently in student responsibility.
- *Moment 2. Validation of the general design*: refers to the engineering evaluation of the feasibility and technological consistency assumed in the design, producing feedback iteration of the moments. Moment under the responsibility of the director with the participation of the student, pertinently documented.

Phase 2. Development: proposes iterative and incremental software development, consistent with the previous phase, composed of work cycles, where the redefinition of requirements at incremental level or by components is contemplated [28]; estimating lower costs than when performed at the end [16]. It is composed of four moments:

- *Moment 1. Product design*: is fed by the analysis of specific product requirements, proceeding to design, scoping, functional specification and test planning. Documented process in student's responsibility.
- *Moment 2. Validation of product design*: for the assurance of compliance with the specifications, in application of the planned tests. Validation and feedback can produce iteration with the previous moment. Responsibility of the director and the student's contest; duly documented.
- *Moment 3. Product development*: construction of software components, related to the conceived design, participate of the moment the component or unit tests performed by the student. Responsible for the moment and documentation, the student.
- *Moment 4. Product testing*: by taking into account the design specifications, in order to verify compliance with the requirements and quality assurance defined in phase 1. It may cause feedback iterations on the moments or phase. Under the responsibility of the director with the participation of the student. Documented moment.

Phase 3. Acceptance: in compliance with what was committed in the preliminary project, the integral consolidation of the software system and the supports, in search of the engineering recognition, including the integration tests [33]. Composed of two moments:

- *Moment 1. Product integration*: refinement process for the integration of products in the software system, allowing the integral vision of the solution provided in compliance with the commitments. The student assumes the responsibility with the support of the documentation.
- *Moment 2. Acceptance tests*: participate in the activities of comprehensive review of compliance with requirements, comprehensive testing of the application and documentary analysis. The director is responsible for issuing compliance certification, with the assistance of the student. It allows the documentary refinement.

3. WORK STRATEGIES

Consistent with what was stated by [25], when describing a good software development process as one that is oriented to the people who participate in the work, the MaTraGra methodology proposes the following work strategies:

- Collaborative work
- Assignment of responsibilities
- Fulfillment of responsibilities
- Frequent feedback
- Concurrent documentation
- Process and activity logs
- Programmatic testing

In order to support the systematic development of the software project through the adoption of the MaTraGra methodology, the TraGraAPP application is presented (see Figure 4), developed for mobile device platforms, with the purpose of managing and monitoring the processes and activities contemplated in the phases and moments.

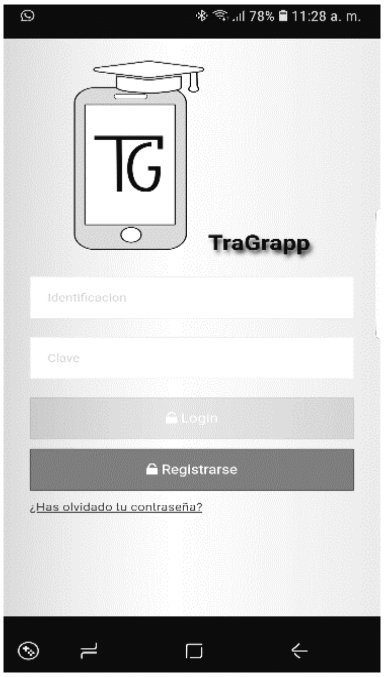


Fig. 4. TraGraapp Application

3.1. Validation of the MaTraGra methodology

For the testing spectrum of the methodology, the 4-DAT tool was used, in order to undertake the evaluative analysis of agility compliance. Additionally, case studies were used as part of the research process aimed at the qualitative analysis of usability, functionality, efficiency and methodical effectiveness. Implicitly in the case study, the Technology Acceptance Model (TAM) was used in compliance with the usability measurement.

The substantiation of the applicability of the methodology in spaces determined by the Engineering degree works that deal with software projects is based on the experimentation of the same in case studies, with the consequent obtaining of information by means of a survey, instrument subjected in its validity to the Content Validity Coefficient (CVC), proposed by [34] for the determination of the theoretical validity in educational research, sponsored, in its measurement, by the consequent Likert scale.

The case studies showed successful results (See Figure 5) related to the application of the methodology, supported by the qualification of the experience of the students and managers participating in the software projects exposed to the work with the MaTraGra methodology and other agile methodologies of recognition.

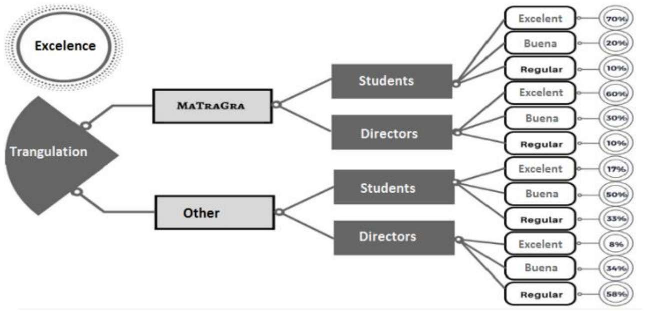


Fig. 5. MaTraGra vs. Other Methods. Source: Own elaboration

CONCLUSIONS

The orientation of the MaTraGra methodology is available to software projects immersed in undergraduate engineering dissertations, without losing the projection of the same to other areas of applicability. It is aligned to the precepts of agile methodologies universally proclaimed, with the disciplinary theoretical support determined by Software Engineering and the contextualization of the Systemic Approach in its conception and application exercise.

The dissertation is provided by the HEI, as an academic exercise of integral connotation, for the diagnosis of the fulfillment of professional and disciplinary knowledge competences, subject to the fulfillment of an academic and administrative regulation of institutional order. It is an academic space that proposes to address engineering problems of all kinds, where to raise software projects, there is an academic environment where particularities associated with the resources and differential responsibilities with the business environment are typified.

The orientation of the dissertation is given towards the demonstration and valuation of professional disciplinary competencies through the monitoring and evaluation of the integral behavior and engineering connotation when approaching and providing solutions to problems by obtaining quality solutions.

The following are presumed to be determining characteristics of the MaTraGra methodology: iterative processes composed of short work cycles; incremental development of products; disposition to small and cohesive work teams; collaborative stages; promotes evaluative feedback; concurrent documentation; and responds to spaces regulated by Higher Education Institutions on research.

The methodology is adapted to the availability of human, physical, technical and financial resources, generally available in specific academic environments for the development of projects within the framework of the research.

In the testing spectrum of the methodology, it was subjected to sustained experimentation, obtaining successful results in terms of agility, consistency, applicability, functionality and willingness to obtain quality products.

The MaTraGra Methodology is based on the immersed projection of the engineering products, to participate in the globalized technological evolution and the consequent conjunction with humanistic precepts in the order of the educational system.

References

[1] J. Sanchez Sierra, S. Ormaechea Liberal y B. Ramos Luceño, «Analysis of the subject Final Degree Project (FDP) in Spanish Communication Sciences degrees,» Revista Española de Documentación Científica, vol. 41, nº 4, pp. 1-13, 2018.

[2] A. Navarro Cadavid, J. D. Fernandez Martínez y J. Morales Velez, «A review of agile methodologies for software development,» Prospectiva, vol. 11, nº 2, pp. 30-39, 2013.

[3] COLLABNET, «13th Annual State of Agile Report,» 8 5 2020. [En línea]. Available: https://stateofagile.com/#ufh-i-613553418-13th-annual-state-of-agile-report/7027494. [Último acceso: 15 03 2021].

[4] E. Orejuela, T. Restrepo Rios, D. Rojas Lara y A. Abuchar Porras, «El traje nuevo del Emperador: la metodología de desarrollo SCRUM. ¿Por qué falla en proyectos de software?,» Avenir, vol. 1, nº 1, pp. 28-31, 2019.

[5] D. Alves Da Silva, E. Costa de Oliveira, E. Dias Canedo y H. Ferreira Martins, «Application of a hybrid process software requirements management,» de 2016 11th Iberian Conference on Information Systems and Technologies (CISTI), Gran Canaria, 2016.

[6] S. Chalela Naffah, A. Valencia Arias, G. Ruiz Rojas y M. Cadavid Orrego, «Psycho-social and familial factors influencing drop-out rates among university students in the context of developing countries,» Revista Lasallista de Investigacion, vol. 17, nº 1, pp. 103-115,

2020.

[7] M. Rojas Betancur y D. C. González, «Deserción estudiantil en la Universidad de Ibagué, Colombia: una lectura histórica en perspectiva cuantitativa,» Zona Próxima, vol. 9, nº 1, pp. 70-83, 2008.

[8] K. Beck y e. al, «Manifiesto por el Desarrollo Ágil de Software,» 25 10 2016. [En línea]. Available: http://agilemanifesto.org/iso/es/manifesto.html. [Último acceso: 15 3 2021].

[9] N. Lalband y D. Kavitha, «Estimation of software quality parameters for hybrid agile process model,» SN Applied Sciences, vol. 3, nº 3, pp. 1-11, 2021.

[10] L. C. Torres Soler, Tesis de Maestría. Qué Hacer, Bogotá: Universidad Autónoma de Colombia, 2013.

[11] IEEE, 1044-1993 - IEEE Standard Classification for Software Anomalies, Estados Unidos: IEEE, 1994.

[12] V. Chandra, «Comparison between Various Software Development Methodologies,» International Journal of Computer Applications, vol. 131, nº 9, pp. 7-10, 2015.

[13] O. Tinoco Gómez, P. P. Rosales López y J. Salas Bacalla, «Criterios de selección de metodologías de desarrollo de software,» Industrial Data, vol. 13, nº 2, pp. 70-74, 2010.

[14] S. Benneth, S. McRobb y R. Farmer, Análisis y Diseño en Sistemas Orientados a Objetos con UML, Madrid: McGraw-Hill, 2007.

[15] H. Janampa Patilla, K. Sotomayor Peralta y M. Prado Vasquez, «Extreme programming software development model on scrum for agile software management,» RISTI - Revista Iberica de Sistemas e Tecnologias de Informacao, vol. 2021, nº E39, pp. 611-626, 2021.

[16] K. Kendall y J. E. Kendall, Análisis y diseño de sistemas, Naucalpan de Juárez: PEARSON EDUCACIÓN, 2011.

[17] M. J. García Rodriguez, Estudio comparativo entre las metodologías ágiles y las metodologías tradicionales para la gestión de proyectos de software [Trabajo de fin de master], Oviedo: Universidad de Oviedo, 2015.

[18] F. Sáez Vacas, Complejidad y tecnologías de la información, Madrid: Universidad Politécnica de Madrid, 2009.

[19] C. Petrella, «Aportes del Enfoque Sistémico a la Comprensión de la Realidad. Avances del proyecto de investigación,» https://es.slideshare.net/junexelkutor/teoriadesistemasaplicadoaorganizaciones, Bogotá, 2017.

[20] A. Qumer y B. Henderson-Sellers, «An evaluation of the degree of agility in six agile methods and its applicability for method engineering,» Information and Software Technology, vol. 50, nº 4, pp. 280-295, 2008.

[21] F. Davis, «Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology,» MIS Quarterly, vol. 13, nº 3, pp. 319-340, 1989.

[22] A. Latorre, D. Rincón y J. Arnal, Bases metodológicas de la investigación educativa, Barcelona: GR92, 2006.

[23] F. Niño Rojas, M. Jimenez Valderrama, S. Gómez Ardila y D. Lancheros Cuesta, «Mobile Learning Based on the Frame Model and Applied to the Solution of Differential Calculus Optimization Problems,» de International Conference on Information Technology & Systems, Santa Elena, 2021.

[24] R. Valero Vargas, J. Palacios Rozo y R. González Silva, «Tecnologías de la Información y la Comunicación y los Objetos Virtuales de Aprendizaje: un apoyo a la presencialidad,» Vínculos, vol. 16, nº 1, pp. 82-91, 2019.

[25] R. Pressman y B. Maxin, Software Engineering: A Practitioner's Approach, Madrid: McGraw-Hill, 2020.

[26] D. Avison y G. Fitzgerald, INFORMATION SYSTEMS DEVELOPMENT, Madrid: McGraw-Hill, 2006.

[27] D. Avison y V. Taylor, «Information Systems Development Methodologies: A Classification According to Problem Situation,» Journal of Information Technology, vol. 12, nº 1, pp. 73-81, 1997.

[28] I. Sommerville, Ingeniería del Software, Madrid: Pearson Educación, 2005.

[29] M. Piattini, F. Ó. García Rubio, F. Pino y I. G. Rodríguez de Guzman, Calidad de Sistemas de Información, Madrid: Ra-Ma, 2019.

[30] ICONTECT, «Normas Icontect,» ICONTEC, 01 01 2020. [En línea]. Available: https://normasicontec.co/. [Último acceso: 15 03 2021].

[31] MADEJA, «Marco de Desarrollo de Software de la Junta de Andalucía,» Madeja, 01 03 2013. [En línea]. Available: http://www.juntadeandalucia.es/servicios/madeja/. [Último acceso: 15 03 2021].

[32] T. Wiegers, More About Software Requirements, Estados Unidos: Microsoft Press, 2006.

[33] D. Sánchez Hernández, F. Lizano Madriz y M. Sandoval Carvajal, «Integration of remote usability tests in extreme programming: A literature review integração de testes de usabilidade remotos em programação extrema: Revisão de documentos,» Uniciencia, vol. 34, nº 1, pp. 20-30, 2020.

[34] G. Mousalli-Kayat, «https://www.researchgate.net/,» 01 01 2017. [En línea]. Available: https://www.researchgate.net/publication/321397866_Los_Instrumentos_de_Evaluacion_en_la_Investigacion_Educativa. [Último acceso: 15 03 2021].