Pesquisa Operacional (2023) 43(spe1): e264750 p.1-25
doi: 10.1590/0101-7438.2023.043spe1.00264750

Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope
ARTICLES

SUPPORTING THE CHOICE OF THE BEST-FIT AGILE MODEL USING FITRADEOFF

Vanessa B. Schramm[1*], Adriana C. Damasceno[2] and Fernando Schramm [3]

Received June 6, 2022 / Accepted September 26, 2022

ABSTRACT. The goal of this study is to propose a decision model for supporting the choice of the best-agile model, based on the Flexible and Interactive Tradeoff (FITradeoff) method. This study provides the definition of an extensive set of 38 measurable criteria that were considered for evaluation of the most popular agile process models used in small and medium-scale enterprises. To illustrate the application of the model, we applied it to a set of alternatives that includes DSDM, SCRUM, XP2 and Crystal and an experienced project manager acted as our decision maker during the interactive elicitation process of the FITradeoff. The results show that the FITradeoff is very valuable for this class of problem because of its strong mathematical foundation and the possibility of combining two different ways for modeling the preferences of decision makers, which makes the process cognitively easier than other multi-criteria methods and increase the confidence on the results. Although our study focuses on the context of small and medium-scale companies for software development, the approach can be used to other types of environments, including distributed software development and large enterprises.

Keywords: project management, software engineering, software project management, Agile model, Multi-Criteria Decision Making/Aid, MCDM/A.

1 INTRODUCTION

Software engineering uses models to represent processes and their relations in order to provide an overview of software development and supporting the decision making that occurs throughout the project life cycle (Pressman & Maxim, 2019; Winters et al., 2020). A software process model is a simplified version of the work being performed during software development, including how activities are organized, the degree of interaction among them and working products (Hurtado

*Corresponding author
[1]Development of Systems for Supporting Sustainable Decisions (DeSiDeS), Federal University of Campina Grande (UFCG), Rua Aprígio Veloso, 882, Bloco CB (REENGE), Bairro Universitário, 58429-900 Campina Grande, PB, Brazil – E-mail: vanessa@labdesides.ufcg.edu.br https://orcid.org/0000-0001-9276-5251
[2]Development of Systems for Supporting Sustainable Decisions (DeSiDeS), Federal University of Campina Grande (UFCG), Rua Aprígio Veloso, 882, Bloco CB (REENGE), Bairro Universitário, 58429-900 Campina Grande, PB, Brazil – E-mail: adriana.damasceno@gmail.com https://orcid.org/0000-0002-5252-2826
[3]Development of Systems for Supporting Sustainable Decisions (DeSiDeS), Federal University of Campina Grande (UFCG), Rua Aprígio Veloso, 882, Bloco CB (REENGE), Bairro Universitário, 58429-900 Campina Grande, PB, Brazil – E-mail: fernando@labdesides.ufcg.edu.br https://orcid.org/0000-0003-3303-615X

Alegría et al., 2014). Since a model captures one of the many views of the software development process, it corresponds to a part of its whole information. In this way, it offers a guidance on how processes can be used, instantiated or executed.

Agile software process models, also called agile models, help managing a software project by letting the team respond to changes in a rapid and creative way through iterative and incremental cycles that can be guided by customers' feedback (Al-Zewairi et al., 2017). Thus, agile models prioritize change of requirements (adaptability) and project simplicity by adopting four core values depicted in the "Agile Manifesto" (Beck et al., 2001): (i) individuals and interactions over processes and tools; (ii) working software over comprehensive documentation; (iii) customer collaboration over contract negotiation; and (iv) responding to changes over following a plan. Agile models address modern business pressures and technology advances (Kettunen & Laanti, 2005; Kalenda et al., 2018) by facilitating activities, such as management of change, faster time to market, visibility and transparency (Diebold et al., 2019). As a consequence, technical community, government, private sector, and academia have used agile models in their projects, making agile a mainstream software process model in use currently (Stavru, 2014). The increasing number of publications in the specialized literature and the number of special issues devoted to agile development also confirms the importance of this topic (Dingsøyr et al., 2012; Hoda et al., 2017; Alaidaros et al., 2019).

Extreme Programming (XP) and SCRUM are the most used agile models for different environments (Dingsøyr et al., 2012; Vallon et al., 2018). Recently, Alaidaros et al. (2019) performed a systematic literature review on this topic, with 48 papers that were published within 2008-2018 and confirmed that Extreme Programming (XP) and SCRUM are the most used, with occurrence of 24%. Other agile models that appeared in the reviewed papers are: Feature-Driven Development (FDD), with 13%; Rational Unified Process (RUP) and Kanban with 9%; Crystal models (7%); Dynamic Systems Development Method (DSDM) (5%); Adaptive Software Development (ASD) (4%); Lean (3%); and Test Driven Development (TDD) (2%).

According to the context in which the software will be developed, a model is more suitable to a specific situation than others (Silva et al., 2016). For example, implementation of agile models in medium enterprises provokes a big impact in the daily business because of lack time (Diebold et al., 2019). El Beggar (2018) adds that, although agile models share common features, none of them could be adopted for any type of Information Technology projects. As a consequence, software companies face difficulties regarding the selection of a software model that match their needs (Vavpotic & Vasilecas, 2011; Hazzan & Dubinsky, 2005; Harb et al., 2015). According to Vavpotic & Vasilecas (2011), organizations often lack knowledge and experience necessary to evaluate and to choose the most suitable model.

Taromirad & Ramsin (2008b) point out that as agile models gain widespread popularity, it is becoming increasingly important that an adequate evaluation framework should be developed for the selection of the most suitable model for a given circumstance. According to Hesari et al. (2010), the selection of software models has become a critical task in software development projects, and they argue for a deep evaluation of models, mainly due to their increasing number

and variety. Hamed & Abushama (2013) say that this selection is a vital activity in any software project, with a great impact on customer satisfaction. Harb et al. (2015) adds that the nature of this decision is complex, involving a hundred of criteria. Finally, according to El Beggar (2018) various studies that were published in the specialized literature are concerned with the comparison of agile methods, but do not provide a deep evaluation of them.

In this paper, we present a multicriteria model for supporting the choice of the agile model to be adopted in software development projects in the context of small and medium-scale enterprises. This model includes a set of measurable criteria that should be considered to ensure the choice of the best-fit agile model according to the characteristics of project, team and company. It also provides the evaluation of main agile models used in small and medium-scale enterprises in relation to the set of criteria. The evaluation will be performed using the FITradeoff multicriteria method. The model allows to incorporate other agile models to be considered in the decision-making process, as well as, it allows project manager degree of freedom to assign criteria's weights to criteria according to the characteristics of the project in concern.

Various multicriteria methods have been drawn up and a way to classify them is considering the rationality the decision maker uses to analyze criteria: using a compensatory rationality or a non-compensatory rationality. We believe that a compensatory rationality is more appropriate in this case, that is, decision maker is able to perform tradeoffs among criteria, allowing a low performance in a given criterion can be compensated for by a high performance in another criterion. With this, we have reduced our set of possibilities to the group of compensatory methods, which includes unique criterion of synthesis methods (Multi-Attribute Value Theory (MAVT)-based methods and Multi-Attribute Utility Theory (MAUT)-based methods). Among the compensatory methods, we chose FITradeoff (De Almeida et al., 2016) that is a MAVT-based method. FITradeoff allows to consider non-linear functions to represent the preference of decision maker in each criterion, improving the quality of the model, but increasing the complexity of the elicitation process. However, unlike other methods that consider non-linear functions, FITradeoff deals with this complexity, reducing possible inconsistencies (the so-called elicitation error). Moreover, in the new version of FITradeoff (de Almeida et al., 2021) allows to combine two approaches for preference modeling: holistic evaluation and elicitation by decomposition. Then, the decision maker can chose either one or other form to have his/her preferences modeled or combine these two approaches. The FITradeoff is implemented into a Decision Support System (DSS) that can be obtained for free (www.cdsid.org.br/fitradeoff).

Regarding the context of decision, it was observed that small and medium-scale enterprises play an important role in software development: studies show that more than 70% of software products were developed in small and medium-scale enterprises (Drury-Grogan et al., 2017). In 2015, 92,4% of software companies in Germany were small and medium-sized (Picot et al., 2015); similarly, in Brazil, this number was 99,5% (Gentile, 2018) in 2019. Not all agile models are well suited for small and medium-scale enterprises due to rapid requirements change, budget, resources, and strict time restrictions. As observed by Hamed & Abushama (2013), normally the selection of a model to support software development in small and medium-scale enterprises

is based on experience. For this reason, the proposed model is intended to support this type of organization, but it can also be applied in the large-scale enterprises and for both practitioners and researchers.

This paper is organized as follows: Section 2 presents a literature review on Multi-Criteria Decision Making/Aid (MCDM/A)-based approaches for supporting the evaluation and selection of agile models. Section 3 presents the structuring of the multi-criteria decision problem, which includes the definition of the set of alternatives (a literature review on main agile models used in commercial projects) and criteria with their description and evaluation scale; Section 4 presents a numerical application of the FITradeoff method; Section 5 presents the discussions; and, finally, Section 6 presents the conclusions of the study.

2 RELATED WORKS

To the best of our knowledge, there are only few studies that are concerned with providing a structured approach for supporting the choice of the best-fit software process model, for which a MCDM/A-based method is used.

Vavpotic & Vasilecas (2011) proposed an index, named Methodology-Project-Value, that is given by the aggregation of alternatives performance in 11 evaluation criteria that were classified into two groups: project proprieties and method proprieties. They calculated the index of 6 models: SCRUM, TDD, XP, RUP, RUP for small projects, and Oracle Custom Development Method (CDM).

Hicdurmaz (2012) presented an approach that used a fuzzy version of AHP (Analytic Hierarchy Process) method with a fuzzy version of TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) method to evaluate 4 alternatives considering 12 criteria. Fuzzy-AHP is used for assigning the criteria weights and Fuzzy-TOPSIS is used to determine the rank of alternatives, based on their overall performance. The approach is not intended for agile models. Demirtas et al. (2014) proposed an approach that combines the qualitative technique SWOT, which was used for formulation of criteria, with a fuzzy version of AHP to compare agile with waterfall models, taking into consideration 13 evaluation criteria.

Harb et al. (2015) proposed an approach based on AHP to evaluate 4 agile models (Crystal, FDD, SCRUM and XP) based on a set of 10 criteria. Silva et al. (2016) provided a multicriteria evaluation of 4 agile models: Crystal, DSDM, SCRUM, and XP. They used SMARTER (Simple Multi Attribute Rating Technique Exploiting Ranking) method (Edwards & Barron, 1994) and considered a set of 13 criteria, whose weights were determined through the Rank Ordered Centroid (ROC) approach (Edwards & Barron, 1994; Barron & Barrett, 1996b,a).

El Beggar (2018) used the Goal/Question/Metric paradigm to construct the set of criteria and applied the PROMETHEE (Preference Ranking Organization Method for Enrichment Evaluation) to evaluate 8 agile models: SCRUM, XP, Crystal Clear, FDD, ASD, Agile Modelling Driven Development (AMDD), Kanban, and DSDM. They proposed 20 criteria that were clas-

sified into four groups: organization, technique, modelling, and quality assurance. These criteria were evaluated using linguistic terms that were converted into trapezoidal fuzzy numbers.

3 THE PROPOSED MODEL

As pointed out by Drury-Grogan et al. (2017), the satisfaction of a decision is related to both satisfaction with decision outcome itself and the decision-making process. In this section, we present the decision-making model to support the choice of the best-fit agile process model (Figure 1). The model is divided into two phases: (i) problem structuring; and (ii) FITradeoff analysis. The input for the first phase was a study in the specialized literature on the main agile models that have been used by in small and medium-scale enterprises' project as well as criteria that are considered by project managers to select agile model for software projects. In the second phase the FITradeoff was applied to perform the multicriteria evaluation, exploring the possibilities of combining different ways for preference elicitation, which can improve the modeling of preferences and consequently improve the confidence in the recommendation.

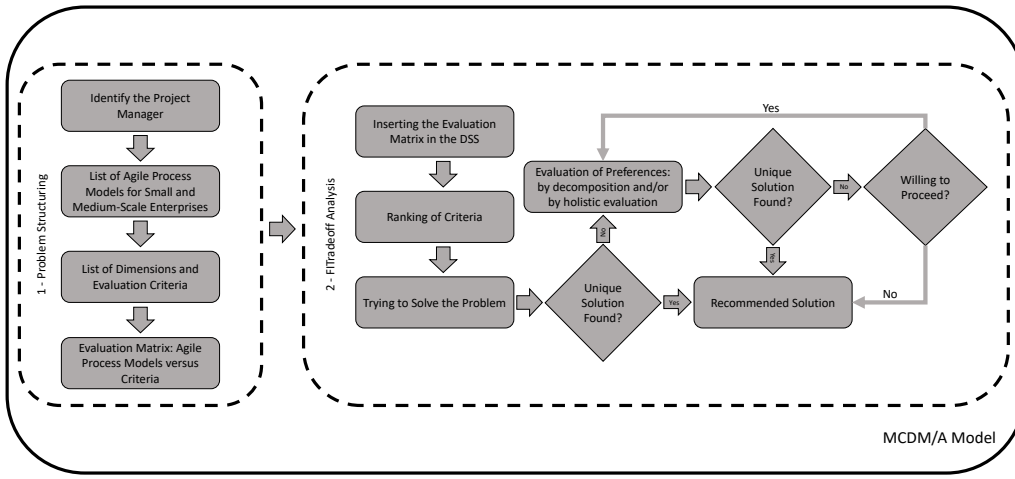

Figure 1 – The MCDM/A proposed model.

The decision regarding the choice of the agile model to be used should consider the preference of the actor who is responsible for the project and who has enough knowledge on the development environment; normally this actor is the respective project manager.

3.1 Agile Process Models

Currently, agile models have been used in different types of project and different contexts. According to Hamed & Abushama (2013), the most popular agile process models for small and medium-scale enterprises are: DSDM (Stapleton, 2003), Extreme Programming 2 (XP2) (Beck, 2000; Beck & Andres, 2005), Crystal models (Cockburn, 2004), and SCRUM (Schwaber & Beedle, 2001; Schwaber, 2004).

DSDM was the first proposed agile model (Larman & Basili, 2003). Its format is owned by the Agile Alliance and it intends to deliver software under restricted time and costs through the control of the requirements along the software development, adjusting the software functionalities accordingly (Moran, 2015). Among its main principles, DSDM allows the team to take decisions without superior approval (empowering the project team), every change can be reversible, the high-level scope must be defined before the project starts, and communication is encouraged. DSDM is divided into three main phases: the pre-project, responsible for setting the project candidates, budget and contract; the lifecycle, where the model is executed, being divided in five main phases; and the post project to guarantee the project efficiency, improving maintenance and improving the work of previous phases.

XP was the second proposed agile model in 1996, with an improved version in 2004, named XP2 (Beck, 2000; Beck & Andres, 2005). XP2 is the second most used agile model (Yang et al., 2016) to be used with software architecture or in a general context (Dikert et al., 2016). It is focused on the software engineering process, particularly on the analysis, development and the testing process in order to increase the quality of the product. As main practices, XP2 requires the whole team to sit together in a single room to improve communication and respect among its members. Although most information about the software behavior is communicated verbally, software requirements also are registered in user stories, which are small cards that contain a specific software behavior. Besides, it uses small releases, design improvement, customer tests and focuses on simple design to deliver value to the project, letting the software to be open, tangible and understandable to the customer. In addition, pair programming and test-driven development are outstanding practices of XP2: the first one requires programmers to work in pairs with a single computer to let the code be thoroughly reviewed; and the second practice requires the code to be tested while it is being developed, allowing rapid feedback about the code quality. In the sequence, XP2 adopts a single coding standard so that the overall system seems to be developed by a single member. Finally, this model adopts a sustainable pace, which allows the project to be developed indefinitely in a given rate or time.

The Crystal family of models (Cockburn, 2004) includes: Clear, Yellow, Orange, Red, and Blue. These set of strategies are based on size and criticality. The Crystal family focuses on people, how they communicate to each other, their talents, skills, and how they interact during the project execution. It requires close communication that restricts the project team to be in a single space, avoids interruptions during tasks, and promotes easy access to expert members in order to clarify questions. Moreover, it demands frequent deliveries and continuous improvement as a way to discover early problems in the project.

SCRUM is the most used model worldwide, being developed for working in a volatile environment (Dikert et al., 2016). It is rapid to implement and addresses management issues that are faced by Information Technology projects. It concentrates on the management of tasks in a team-based environment. Besides that, it emphasizes empirical feedback, team self-management and product deliveries in short iterations. The main roles of this model are the product owner (the leader over the product that prioritize software features – also called stakeholder), SCRUM mas-

ter (responsible for helping the team and disseminate SCRUM practices) and the SCRUM team. The project is guided by sprints, which is a set of tasks to be executed and divided into iterative cycles. The product backlog stores the features to be implemented along the project. The allocated tasks are transferred to the sprint backlog and they are defined and distributed to the team in meetings defined by the SCRUM model.

Besides these options, other agile process models can be added in the list of alternatives.

3.2 Dimensions and Criteria

Taromirad & Ramsin (2008a,b) proposed a comprehensive set of evaluation criteria in order to support the selection of the most appropriate agile model for software projects. They presented ninety criteria organized into a hierarchical structure of five groups that have been divided into subgroups. The authors also present an overview of existing studies that bring frameworks/methods for evaluation and selection of agile models with focus on criteria that are considered in the evaluation process. This work was revised considering the main aspects of agile models and applied in the context of XP agile model. According to them, the drawback of these studies is an evaluation based on a limited set of criteria, which does not cover aspects and characteristics regarded as important in the context of the selection of an agile software development model, or the lack of quantitative metrics. Another weak point of their work is the lack of experimentation or case studies with companies, letting implications for practical application unknown.

We analyzed Taromirad and Ramsin's work and removed those criteria that were interdependent and redundant when taking into account small and medium enterprises. Also, we replaced them by those ones for which it was provided an evaluation scale with a well-defined direction of preference. As a consequence, this set was reduced from ninety to thirty-eight criteria that were organized into four groups: (i) Process; (ii) Agility; (iii) Usage; and (iv) Cross-Context.

The group "Process" is concerned with the methodological evaluation of the model and contains 27 criteria. It comprises six subgroups: (i) Definition; (ii) Phases; (iii) Artifacts; (iv) Requirements; (v) Documentation; and (vi) General Features. The subcategory "Definition" contains criteria that evaluate the definition of the process development for each model; considering only this dimension of analysis, a model that provides more details about its process is preferable to a model that neglects its process. The subcategory "Phases" contains criteria that evaluate how the models cover the phases of a generic development cycle; a model that covers a higher number of phases, considering techniques to assure a smooth transition between phases without gaps in the process, is preferable to another. The subcategory "Artifacts" contains criteria that evaluate the quality of the artifacts that are produced during the process development of each model; for this study, the term quality is determined by the quantity of artifacts, the existence of models and standards, the consistency among them, and, finally, how tangible, understandable, and testable they are. The subcategory "Requirements" includes criteria that evaluate requirements; considering this single dimension, it will be always preferable a model that addresses functional and

non-functional requirements, allowing changes on it along the process development, and whose products can be traced by the requirements. “Documentation” focuses on documentation that is a desirable aspect in a software development process. Finally, “General Features” includes criteria that evaluate general features of the model, such as complexity of the model, completeness in terms of lifecycle coverage and suitable products, practicality and practicability. Table 1 presents criteria, description, evaluation scale and direction of preference for the Process group.

The group “Agility” is concerned with how the software model adheres to the agile principles; a model that is more fit to agile principle is preferable. It represents a single group with 6 criteria: speed, sustainability, flexibility, learning, responsiveness, and leanness. Table 2 shows these criteria, description, evaluation scale and direction of preference.

The group “Usage” focuses on practical issues of the model, especially aspects of the project management, documentation and adaptation to different projects. It is preferable a model with project management activities (project and team management, quality assurance, risk management and so on), tutorials and trainings and portability to different projects (flexibility, scalability, extensibility, tutorials and training and empirical evidence). Table 3 shows further details for the Usage Group.

The Cross-Context group is comprised of a single criterion named “Constraints” which is devoted for evaluation of aspects that were not directly covered by the remaining criteria. Table 4 presents description, evaluation scale and direction preference of the Constraints criteria.

Table 1 – Criteria for the Process group.

c_i	Criterion	Description	Evaluation scale	Direction of preference
c_1	Explicitness and Unambiguity	Is the development process defined explicitly and unambiguously?	Yes or No	"Yes" is preferable to "No" (Max)
c_2	Rationale	Has the process been rationalized through providing extensive and precise explanations?	Yes or No	"Yes" is preferable to "No" (Max)
c_3	Completeness	A complete process definition includes definitions for: development cycle, lifecycle, roles, activities, modeling language, artifacts, practices/ techniques, rules, and umbrella activities.	Ratio of the number of existing definitions to the total.	A higher value is preferable to a smaller value (Max)
c_4	Generic development lifecycle coverage	Which phases of the generic development cycle are covered by the development process? Generic phases include: inception, analysis, design, implementation, test, deployment, maintenance, support, and postmortem.	Ratio of the number of covered phases to the total number of generic phases.	A higher value is preferable to a smaller value (Max)
c_5	Smooth transition	Is the transition between the phases smooth?	Yes or No	"Yes" is preferable to "No" (Max)
c_6	Seamless transition	Is the transition between the phases seamlessness?	Yes or No	"Yes" is preferable to "No" (Max)
c_7	Adequate products	Does the development process produce the products typically associated with the generic development activities (feasibility analysis, requirement specification, design, modeling, documentation, test, training and deployment)?	Ratio of product types supported to the ideal number of product types.	A higher value is preferable to a smaller value (Max)
c_8	Modeling coverage	Do the products include models (analysis and design)?	Yes or No	"Yes" is preferable to "No" (Max)
c_9	Consistency	This criterion evaluates the level at products complement each other.	High, Medium (overlapping exist that can result in inconsistencies) or Low	"High" is preferable to "Low" (Max)
c_{10}	Tangibility/Visibility	This criterion evaluates the level at the products are tangible, understandable, and testable to end-users.	High, Medium or Low	"High" is preferable to "Low" (Max)
c_{11}	Standards	Are there any specific standards for the products?	Yes or No	"Yes" is preferable to "No" (Max)
c_{12}	Process based on functional/non-functional requirements	Is the development process based on requirements?	Yes or No	"Yes" is preferable to "No" (Max)
c_{13}	Non-functional requirements verification	Are the non-functional requirements addressed?	Yes or No	"Yes" is preferable to "No" (Max)
c_{14}	Traceability	Can the products be traced to the requirements?	Yes or No	"Yes" is preferable to "No" (Max)
c_{15}	Requirements change	Does the development process let changes in requirements?	Yes or No	"Yes" is preferable to "No" (Max)
c_{16}	Available and published documents	Is the development process published and available to users?	Published and available (2), Published but not available (1); No published and not available (0)	A higher value is preferable to a smaller value (Max)
c_{17}	Process enactment documentation	Is the running process documented?	Yes or No	"Yes" is preferable to "No" (Max)
c_{18}	Size/Complexity	Size/complexity is defined as a function of building blocks of the development process.	Total number of practices, roles, products, and phases/ stages.	A lower value is preferable to a higher value (Min)
c_{19}	Practicality	This criterion evaluates the level at the process is practical, based on issues affecting practicality, such as support for umbrella activities, independence from specific tools, pragmatic techniques, concrete rules, and non-overlapping activities.	High, Medium or Low	"High" is preferable to "Low" (Max)
c_{20}	Practicability	This criterion evaluates the level at process development is practicable, through providing effective activities or techniques such as suitability filters and instantiation/ adaptation methods.	High, Medium or Low	"High" is preferable to "Low" (Max)

Table 2 – Criteria for the Agility group.

c_i	Criterion	Description	Evaluation scale	Direction of preference
c_{21}	Speed	How quickly does the methodology produce results?	1/[iteration length (in days) + deployment interval (in days)]	A higher value is preferable to a smaller value (Max)
c_{22}	Sustainability	Are speed and quality maintained until the end? Are they controlled or monitored?	Yes or No	"Yes" is preferable to "No" (Max)
c_{23}	Flexibility	Are expected/ unexpected changes captured and handled in the project?	Ratio of the number of supporting activities and practices to the total	A higher value is preferable to a smaller value (Max)
c_{24}	Learning	Does the process "learn" from past projects and previous iterations?	Ratio of the number of supporting activities and practices to the total	A higher value is preferable to a smaller value (Max)
c_{25}	Responsiveness	Does the method provide feedback?	Ratio of the number of supporting activities and practices to the total	A higher value is preferable to a smaller value (Max)
c_{26}	Leanness	Does the method value shorter time spans, using economical and simple quality-assured means for production?	Ratio of the number of supporting activities and practices to the total	A higher value is preferable to a smaller value (Max)

Table 3 – Criteria for the Usage group.

c_i	Criterion	Description	Evaluation scale	Direction of preference
c_{27}	Computational complexity	This criterion evaluates the level at computational complexity.	High (scientific and complex), Medium (business-oriented) or Low (simple and personal usage).	"Low" is preferable to "High" (Min)
c_{28}	Project management	Support for development process management; including planning, scheduling, controlling and monitoring, and process review.	Ratio of the number of covered activities to the total.	A higher value is preferable to a smaller value (Max)
c_{29}	Team management	Does the methodology provide any process for team and people management?	Yes or No	"Yes" is preferable to "No" (Max)
c_{30}	Quality assurance	Does the methodology support for quality assurance techniques such as technical review, continuous verification and validation, and strategies/ techniques enhancing requirements traceability?	Yes or No	"Yes" is preferable to "No" (Max)
c_{31}	Risk management	Does the methodology support for risk management techniques such as feasibility analysis, risk-based planning, active user involvement, continuous verification and validation, iterative process/ product/ plan reviews, and continuous integration?	Yes or No	"Yes" is preferable to "No" (Max)
c_{32}	Adaptation and customization	Does the methodology provide methods for customizing it based on the parameters of the project at hand?	Yes or No	"Yes" is preferable to "No" (Max)
c_{33}	Flexibility	Does the methodology allow the process and modeling language to be changed during its execution?	Yes or No	"Yes" is preferable to "No" (Max)
c_{34}	Scalability	Is the methodology suitable for projects with different sizes, criticalities and complexities?	Yes or No	"Yes" is preferable to "No" (Max)
c_{35}	Extensibility	Does the methodology provide any extension points?	Yes or No	"Yes" is preferable to "No" (Max)
c_{36}	Tutorials and training	Are tutorials and training documents available?	Yes or No	"Yes" is preferable to "No" (Max)
c_{37}	Empirical evidence	Does empirical evidence exist?	Yes or No	"Yes" is preferable to "No" (Max)

Table 4 – Criteria for the Constraints group.

c_i	Criterion	Description	Evaluation scale	Direction of preference
c_{38}	Constraints	Any general constraints in the methodology that influences practicality?	Yes or No	"No" is preferable to "Yes" (Min)

3.3 FITradeoff Analysis

The first step of the FITradeoff DSS (http://fitradeoff.org/) is inserting the evaluation matrix and inform the direction of preference of each criterion. Then, the evaluation matrix is converted into a decision matrix using linear functions $v_i(x)$ associated to each criterion $i\,(i = 1, 2, \cdots, n)$ that will assign a performance value to each single alternative x in each criterion.

To rank the criteria, decision makers can choose between two interactive procedures, namely: pairwise comparison or overall evaluation. The first procedure corresponds to an interactive process between analyst and decision maker, in which the former asks to the project manager to choose consequences A or B. As can be seen in Figure 2, these consequences represent the best outcomes in two different criteria. If the decision maker choose consequence A instead of B, it represents that he/she prefers the criterion C_1 instead of C_2. The second procedure also corresponds to an interactive process, in which analyst asks to the project manager to consider a hypothetical alternative, whose performance is the worst considering all the criteria. Then, the decision analyst proceeds with the following assumption: "supposing that you can improve the performance of this alternative in only one of the criteria to the maximum value, which criteria would you chose?". The chosen criterion is put in the first position of the ranking of criteria priorities. As can be seen in Figure 3, the same question is repeated until all criteria are ranked, being possible to have more than one criterion occupying the same position in case of indifference among them. The output of this step is the ranking of criteria (from the best to the worst). Based on this ranking, a weight space $(k_1, k_2, \cdots, k_n, \text{where} \sum_{i=1}^{n} k_i = 1; k_i \geq 0)$ is determined.

The next step is trying to solve the problem by seeking an alternative from the set that has the maximum overall performance considering the weight space. For this, it calculates the overall performance of each alternative x by aggregating its respective performance value $v(x)$, according to Equation 1.

$$v(x) = \sum_{i=1}^{n} k_i v_i(x) \tag{1}$$

where: k_i is the weight of criteria i and v_i is a value function that measures the performance of the alternative in relation to the criteria i. And usually assuming that:

$$\sum_{k=1}^{n} k_i = 1\,(k_i \geq 0)$$

Then, it identifies the potentially optimal alternatives, that is, alternatives, the overall performance of which is greater than or equal to the value of any other alternative from the whole

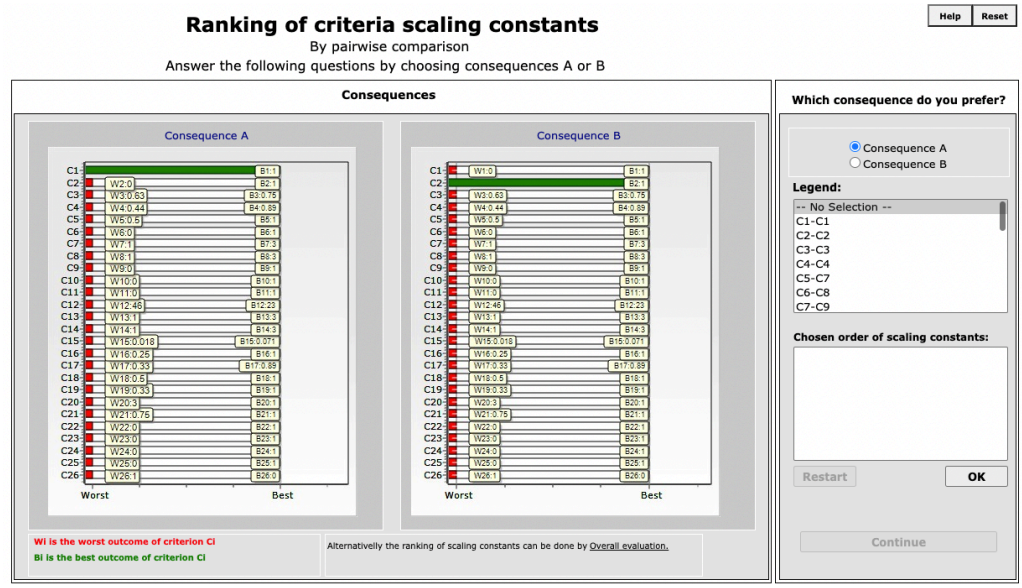

Figure 2 – Ranking of criteria using pairwise comparison.

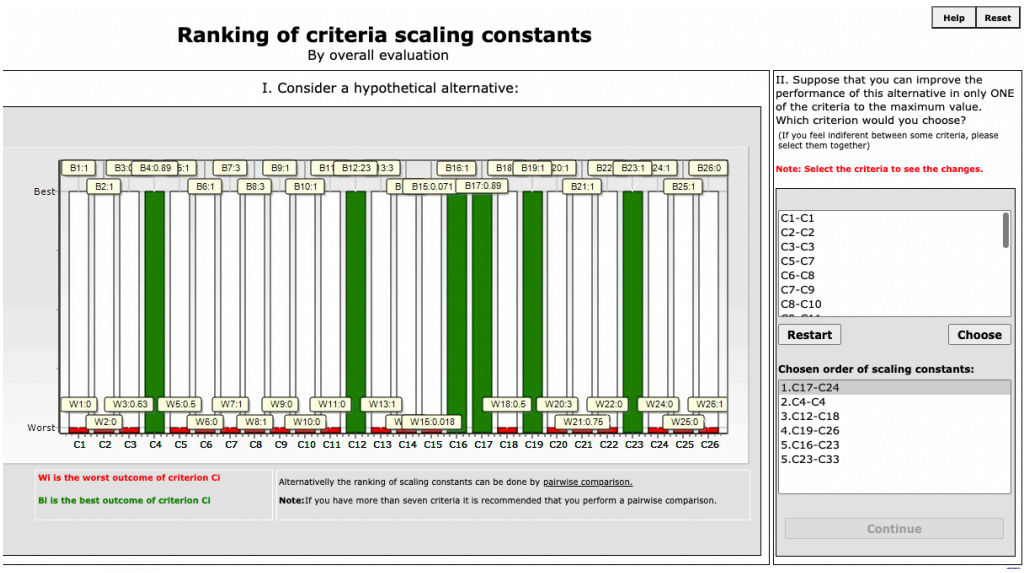

Figure 3 – Ranking of criteria using overall evaluation.

set, for at least one vector of weights in the whole weight space. After that, the DSS eliminates alternatives for which the overall performances are less than the value of at least one of the alternatives from the subset of potentially optimal alternatives. The problem is solved if a unique alternative remains in the subset after eliminating the inferior ones. Otherwise, the de-

cision maker can choose between continuing the elicitation by decomposition or switching to holistic evaluation. Therefore, the steps for evaluation of preferences is started.

The evaluation of preferences is the main part of FITradeoff. If the decision maker chooses a holistic evaluation, thus, the DSS shows current results by four visualization ways: (i) bar graph; (ii) tabular; (iii) radar graph; and (iv) bubble graph. With this information, the decision maker can choose to continue the holistic evaluation or to go back for elicitation by decomposition. If he/she is confident to perform a holistic evaluation, he/she may select the best alternative or exclude the worst one. Otherwise, he/she may go back to elicitation by decomposition. When he/she will be confident, he/she may return for the holistic evaluation again.

In the elicitation by decomposition, firstly, the decision maker is asked to choose between two consequences and depending on his/her response, the distribution of weights is inferred. Based on the new distribution of weights, a value for x_i is set, which represents an outcome between the best and worst performance in the criterion c_i. This new value will be used for comparison of consequences for eliciting weights. For each pair of consequences, the decision maker should answer your preference, being possible to present no answer or even ask to see partial results. In most cases, the decision maker is able to indicate a preference whose available options are threefold: prefer consequence A, which means set $x_i = x_i^{''}$, where $x_i^{''}$ is an outcome between the best (b_i) and the worst (w_i) performances in relation to the criteria i, thus $1 = v_i(b_i) > v_i\left(x_i^{''}\right) > v_i\left(w_i^{''}\right) = 0$; prefer consequence B, which means set $x_i = x_i^{'}$ where $x_i^{'}$ is also an outcome between b_i and w_i, with $v_i\left(x_i^{'}\right) > v_i\left(x_i^{''}\right)$, thus $1 = v_i(b_i) > v_i\left(x_i^{'}\right) > v_i\left(x_i^{''}\right) > v_i(w_i) = 0$; and indifference between A and B, which means set $x_i = x_i^I$, where x_i^I is an outcome between $x_i^{'}$ and $x_i^{''}$, thus $1 = v_i(b_i) > v_i\left(x_i^{'}\right) > v_i\left(x_i^I\right) > v_i\left(x_i^{''}\right) > v_i(w_i) = 0$. Based on the answer, the DSS solves a Linear Programing Problem (LPP) (Equation 2) whose intent is to reduce the weight space and, consequently, the set of potentially optimal alternatives by eliminating the dominated ones.

$$
\begin{aligned}
&\underset{k_1,k_2,\cdots,k_i}{max} \sum_{i=1}^{n} k_i v_i(x_{ij}),\ j = 1,2,\cdots,m \\
&s.t. \\
&\sum_{i=1}^{n} k_i v_i(x_{ij}) \geq \sum_{i=1}^{n} k_i v_i(x_{iz}),\ z = 1,2,\cdots m, z \neq j \\
&k_{i+1} \leq k_i v_i\left(x_i^{'}\right) - \varepsilon \text{ for } i = 1 \text{ to } n-1 \\
&k_{i+1} \leq k_i v_i\left(x_i^{''}\right) - \varepsilon \text{ for } i = 1 \text{ to } n-1 \\
&\sum_{i=1}^{n} k_i = 1 \\
&k_i \geq 0,\ i = 1,2,\cdots,n
\end{aligned}
\tag{2}
$$

The resolution of this LPP may imply in the following situations: only one potentially optimal alternative is found (unique solution) and then the MCDM/A problem is solved; otherwise, a new cycle of analysis must be performed on this new subset of alternatives.

The steps of the DSS are executed until the optimal alternative is found or the decision maker decides to stop, and a recommendation is achieved.

4 EXAMPLE OF APPLICATION

To illustrate the application of the model (Fig. 1), we applied it to a set of alternatives that includes DSDM, SCRUM, XP2 and Crystal. To construct the evaluation matrix we used the evaluation of XP2 performed by Taromirad & Ramsin (2008b). In addition, we performed a new evaluation for the DSDM, SCRUM and Crystal models based on information depicted in Cockburn (2004), Schwaber & Beedle (2001), Schwaber (2004), and Moran (2015). This matrix is presented in Table 5.

As observed in the matrix (Table 5), the agile models have the same performance for the criteria c_5, c_6, c_{13}, c_{15}, c_{16}, c_{17}, c_{22}, c_{29}, c_{30}, c_{31}, c_{35} and c_{36}. Therefore, these criteria were not considered. As for the remaining criteria, the difference of performances exists despite been small, and they satisfy the monotonicity property that is a necessary condition for using additive aggregation models.

The application of the model requires the presence of a decision maker to determine the weights of criteria, using the interactive elicitation process of the FITradeoff, which included a series of question-answer rounds between analyst and decision maker. In our case, an experienced project manager of a medium-scale enterprise located in the Northeastern of Brazil acted as our decision maker. The ranking of criteria by overall evaluation (Figure 3) was chosen because during the elicitation process indifference among some criteria have appeared. Table 6 presents the ranking of criteria that was obtained for this application.

4.1 Elicitation by Decomposition

For this application, only one alternative was removed from the set and then, the procedure for evaluation of preferences by decomposition was performed for reducing the weight space. The steps of the method were executed until the optimal alternative is found. Figure 4 shows the DSS screen for the interactive procedure for evaluation of preferences by decomposition. The solution was found after the DSS runs eight cycles (Table 7).

The DSS recommends the SCRUM as the best-fit model according to decision maker preferences (Figure 5). Graph from Figure 6 illustrate the weight space and values of criteria weights for which the SCRUM is considered to be the optimal alternative.

Finally, the decision maker chooses to perform a sensitivity analysis. In this step, he decided to vary $\pm 10\%$ the consequence space of all criteria (Figure 7). The result presented in Figure 8 shows that the SCRUM agile model has been always included in the subset of Potentially

Table 5 – Evaluation matrix.

Criterion (c_i)	**Agile Models**			
	DSDM	**SCRUM**	**XP2**	**Crystal**
c_1	No	No	Yes	Yes
c_2	No	No	No	Yes
c_3	6/9	5/9	5/9	6/9
c_4	6/9	4/9	8/9	7/9
c_5	Yes	Yes	Yes	Yes
c_6	No	No	No	No
c_7	4/8	4/8	7/8	8/8
c_8	No	No	Yes	No
c_9	Medium	Medium	High	High
c_{10}	Medium	High	High	Medium
c_{11}	Yes	No	Yes	Yes
c_{12}	Yes	Yes	Yes	No
c_{13}	Yes	Yes	Yes	Yes
c_{14}	Yes	Yes	Yes	No
c_{15}	Yes	Yes	Yes	Yes
c_{16}	2	2	2	2
c_{17}	Yes	Yes	Yes	Yes
c_{18}	30	23	40	46
c_{19}	Medium	Medium	High	Medium
c_{20}	Low	Low	High	High
c_{21}	1/14	1/14	1/14	1/56
c_{22}	Yes	Yes	Yes	Yes
c_{23}	15/18	8/8	5/9	2/8
c_{24}	16/18	7/8	3/9	3/8
c_{25}	15/18	8/8	7/9	4/8
c_{26}	6/18	8/8	5/9	4/8
c_{27}	Medium	High	Medium	High
c_{28}	3/4	4/4	3/4	4/4
c_{29}	Yes	Yes	Yes	Yes
c_{30}	Yes	Yes	Yes	Yes
c_{31}	Yes	Yes	Yes	Yes
c_{32}	No	Yes	No	Yes
c_{33}	No	Yes	No	No
c_{34}	No	Yes	Yes	Yes
c_{35}	No	No	No	No
c_{36}	Yes	Yes	Yes	Yes
c_{37}	Yes	Yes	No	Yes
c_{38}	Yes	No	No	No

Table 6 – The importance ranking of the criteria.

Rank j	1	2	3	4	5	6	7	8	9	10	11	12
Criterion (c_i)	c_{24}	c_4	c_{18}	c_{26}	c_{23} c_{33}	c_7	c_{25}	c_3	c_{28}	c_{21}	c_{27}	c_9
Rank j	13	14	15	16	17	18	19	20	21	22	23	-
Criterion (c_i)	c_{10}	c_{19} c_{20}	c_1	c_{32}	c_8	c_{14}	c_2	c_{12} c_{11}	c_{38}	c_{34}	c_{37}	-

Table 7 – Cycles, inputs and outputs provided by the elicitation by decomposition.

Cycles	**Consequence A** (x_i)	**Consequence B: Best of**	**Decision maker's choice**	**Potentially optimal alternatives**
1	$x_{24} = 0.61$	c_{37}	B	DSDM, SCRUM, XP2
2	$x_{24} = 0.75$	c_4	B	SCRUM, XP2
3	$x_{24} = 0.82$	c_4	B	SCRUM, XP2
4	$x_4 = 0.778$	c_{18}	B	SCRUM, XP2
5	$x_4 = 0.834$	c_{18}	B	SCRUM, XP2
6	$x_{18} = 28.75$	c_{26}	B	SCRUM, XP2
7	$x_{18} = 25.875$	c_{26}	B	SCRUM, XP2
8	$x_{26} = 0.833$	c_{23}	I	SCRUM

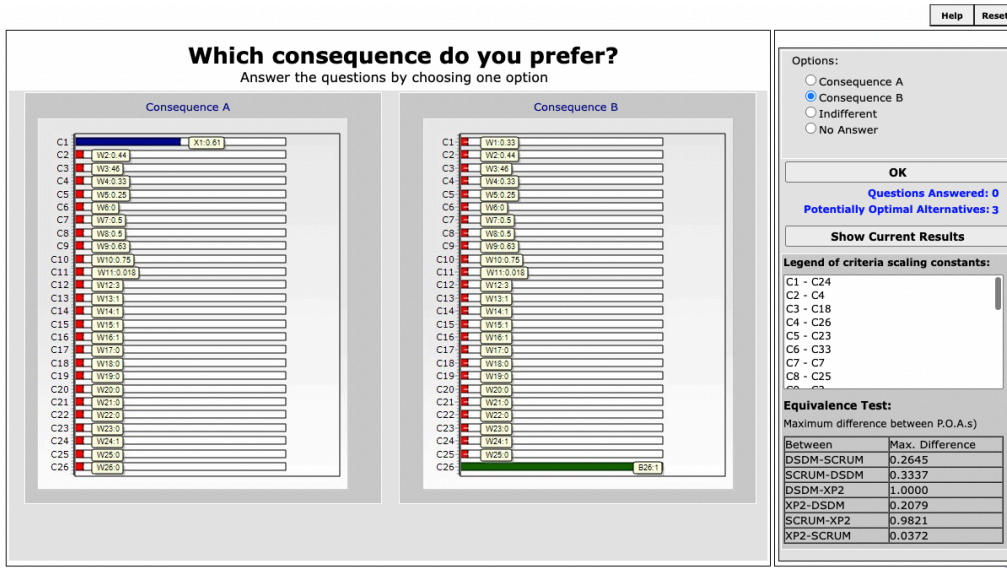

Figure 4 – Evaluating of preferences by decomposition.

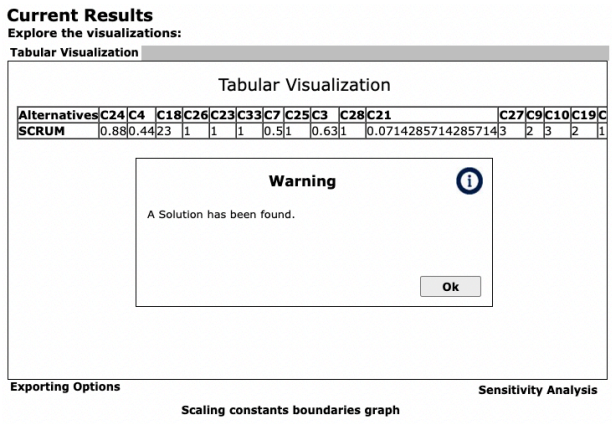

Figure 5 – Current result with the optimal alternative.

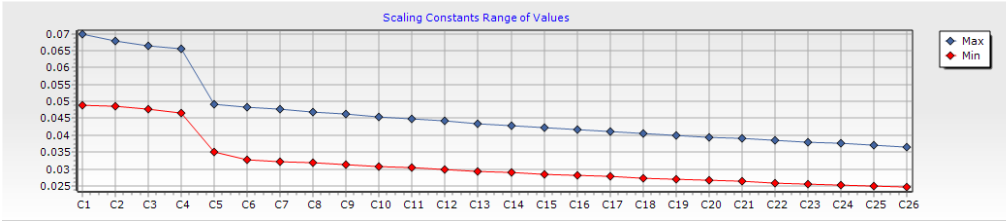

Figure 6 – Scaling constants boundaries graph for the optimal alternative.

Optimal Alternatives during the sensitive analysis interactions. Therefore, the decision maker became more confident to choose the SCRUM model.

4.2 Holistic Evaluation

In this section, the holistic evaluation available in the FITradeoff DSS was performed with the same decision maker. Moreover, it was based on the same ranking of criteria used before (Table 6).

The main purpose of this step is to explore different ways to choose the best-fit agile model, maybe reducing the number of total questions answered by the decision maker or at least evaluating if he/she is confident with the optimal solution recommended through the elicitation by decomposition process performed before.

The first result available in the holistic evaluation process is presented in Figure 9. The analyst showed to decision maker four ways that he could explore the results. Thus, he preferred the visualization by radar graph. According to him, he has chosen this way because it is better to explore the results of alternatives in each criterion, and compare this with the consequences presented in the elicitation by decomposition process. In the first result it is possible to see that the SCRUM agile model has the best performance in seven of the first ten more important criteria. Based on this insight, the analyst asked if the decision maker is confident to perform a holistic

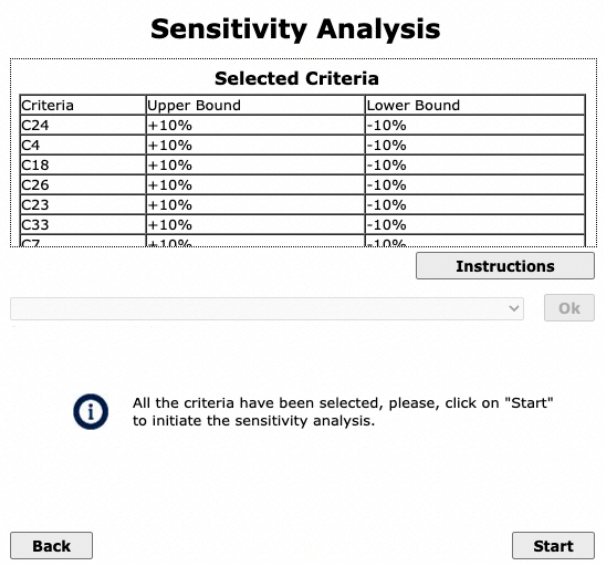

Figure 7 – Sensitivity analysis parameters.

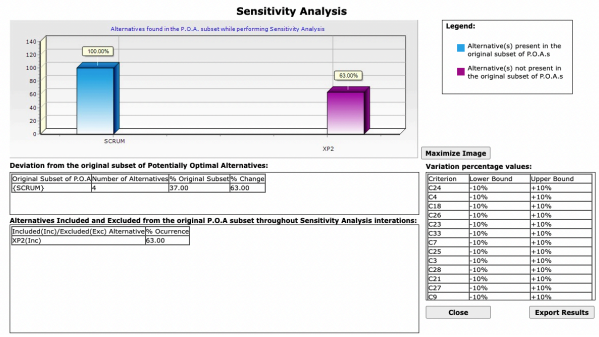

Figure 8 – Sensitivity analysis result.

evaluation, aiming to choose the best alternative or to eliminate the worst ones. He decided to came back to the elicitation by decomposition process, arguing that he would like to use the radar graph to support him in this process.

Using the radar graph in the elicitation by decomposition process, decision maker verified that he could choose consequence A in the first question because the SCRUM agile model is the best one in both criteria, c_{24} and c_{37}, and it is the single alternative with a available value in c_{37}. In the second question, he choose the consequence A because the difference of performance among alternatives in c_{24} is greater than in c_4. In the third question, the subset of potential optimal alternatives decreased to only two agile models (DSDM and SCRUM), thus, he saw the current results in the holistic evaluation and choose the consequence B because the difference of performance among alternatives in c_{18} is smaller than in c_4. In the fourth and fifth questions,

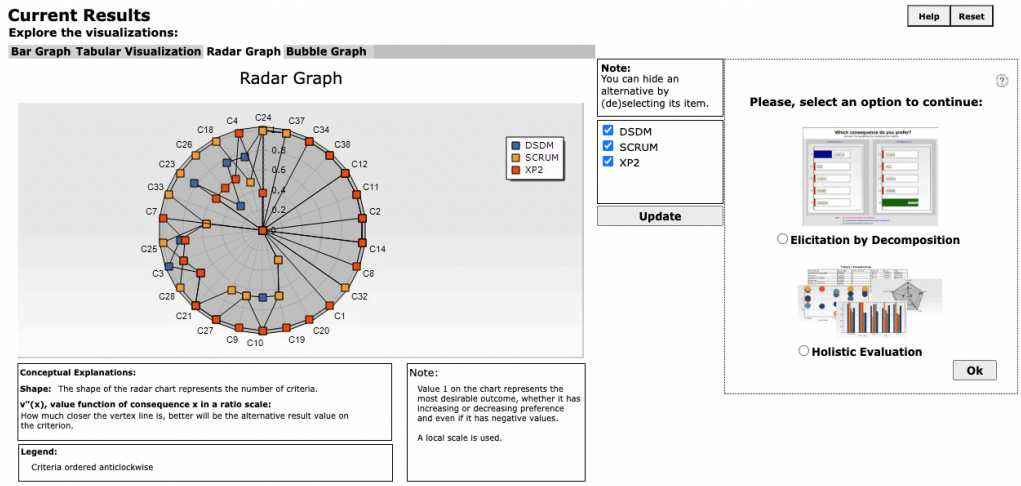

Figure 9 – Current results in holistic evaluation.

he choose the consequence B because SCRUM is the best alternative in c_{18}, c_{26}, and c_{23}. In the sixth and seventh questions, he was indifferent because he saw the current results in the holistic evaluation and both alternatives will give him the same benefits. In the eighth question, he choose consequence B because SCRUM present the maximum difference in relation to DSDM and it has the best performance in c_{28}. In the ninth and tenth questions, he choose consequence A and B, respectively, because SCRUM was the single alternative with evaluation in c_{21}. In the eleventh and twelfth questions, he saw that the SCRUM model still has a maximum difference in relation to DSDM ($DSDM - SCRUM = 0.0179$ and $SCRUM - DSDM = 0.2835$), thus, he chooses consequence B and A, respectively.

In the thirteenth, fourteenth and fifteenth questions, he was indifferent. In the last question, he choose the consequence B because he could receive a better benefit in c_{18}. The results are summarized in Table 8.

5 DISCUSSION

The first phase of a MCDM/A study is the problem structuring that includes: to identify interested parts whose preferences will be modeled, that is the decision maker(s); to determine the set of alternatives; and to define the objectives, which should be converted into measurable variables (criteria). Regarding the alternatives, we have considered the most used agile process models by small and medium-scale enterprises, but the model allows to include other options. The set of criteria was based on a study performed in the specialized literature, resulting in a overarching set of 38 measurable criteria with their respective evaluation scale. The set of criteria was not validated with potential users of the model, which can be a weakness of the model to be tackled in future works.

Table 8 – Cycles, inputs and outputs provided using both elicitation by decomposition and holistic evaluation.

Cycles	Consequence A (x_i)	Consequence B: Best of	Decision maker's choice	Potentially optimal alternatives
1	$x_{24} = 0.61$	$c_{37} = 1$	A	DSDM, SCRUM, XP2
2	$x_{24} = 0.61$	$c_4 = 0.89$	A	DSDM, SCRUM, XP2
3	$x_4 = 0.665$	$c_{18} = 23$	B	DSDM, SCRUM
4	$x_{18} = 34.5$	$c_{26} = 1$	B	DSDM, SCRUM
5	$x_{26} = 0.665$	$c_{23} = 1$	B	DSDM, SCRUM
6	$x_7 = 0.75$	$c_{25} = 1$	I	DSDM, SCRUM
7	$x_{25} = 0.75$	$c_3 = 0.75$	I	DSDM, SCRUM
8	$x_3 = 0.69$	$c_{28} = 1$	B	DSDM, SCRUM
9	$x_{28} = 0875$	$c_{21} = 0.714$	A	DSDM, SCRUM
10	$x_{21} = 0.045$	$c_{27} = 1$	B	DSDM, SCRUM
11	$x_{27} = 2$	$c_9 = 3$	B	DSDM, SCRUM
12	$x_9 = 2$	$c_{10} = 3$	A	DSDM, SCRUM
13	$x_{10} = 2$	$c_{19} = 3$	I	DSDM, SCRUM
14	$x_{19} = 2$	$c_1 = 1$	I	DSDM, SCRUM
15	$x_{24} = 0.47$	$c_4 = 0.89$	I	DSDM, SCRUM
16	$x_4 = 0.778$	$c_{18} = 23$	B	SCRUM

We provided an example of application of the model, however, we considered a real-life decision maker, who was an experienced project manager of a medium-scale enterprise located in the Northeastern of Brazil.

As for the preference modeling, the decision maker was able to perform various pairwise comparisons, which correspond to the performance of the alternatives in the set of criteria considered. In this sense, firstly, we applied the elicitation by decomposition approach and after eight cycles, the FITradeoff method managed to recommend an optimal solution. A sensitivity analysis was performed to confirm the robustness of the result.

Moreover, in order to test the new feature of FITradeoff DSS, we also performed the modeling of preferences by combining the elicitation by decomposition and the holistic evaluation that was useful to give more insights to the decision maker, who opted to not finish the process until the procedure recommends the optimal alternative. After sixteen cycles, the FITradeoff method managed to recommend a solution that was considered satisfactory by the decision maker because it was the same obtained when only the elicitation by composition was performed. Although this analysis had made the process longer, it was very important to increase the confidence of the decision maker, making him more comfortable to decide.

Regarding of the usability of FITradeoff DSS, one negative aspect that was observed is that if a mistake is made during the elicitation process, it is necessary restart it from the beginning, losing all information that had been already provided by the decision maker; only the matrix of consequences inputs are stored.

6 CONCLUSIONS

In this paper, the FITradeoff method was applied to support researchers and practitioners to better select the most appropriate agile model that matches the needs of specific software development projects focusing on small and medium-scale enterprises.

The high number of evaluation criteria indicates the complex nature of this type of decision whose consequences may have great impact on project success and customer satisfaction. The approach aids project managers to choose the software development model in a systematic manner, considering particular aspects of the software development environment and observing the business ecosystem of the project.

For this study, the set of available agile models was reduced to include alternatives that are commonly used by small and medium-scale enterprises. Even so, the approach can be used to other types of environments, including distributed software development and large enterprises. As future work, we intend to increase the evaluation matrix to include other agile models, to validate the set of criteria, and to perform other applications in order to ensure the applicability of the model to support real-life situations.

As for the new version of FITradeoff, we observed that the possibility of combining two different ways for eliciting the preferences of decision maker contributes to increase the confidence of the decision maker in how the process is conducted and in the recommendation provided by the model; from the point of view of the analyst, the confidence in the result come from the strong mathematical foundation that the method has, however, it is important to emphasize that some effort should be applied to improve the usability of the DSS.

Acknowledgment

This work was supported by the Brazilian Research Council – CNPq.

References

AL-ZEWAIRI M, BILTAWI M, ETAIWI W & SHAOUT A. 2017. Agile Software Development Methodologies: Survey of Surveys. *J. Comput. Commun.*, **05**(05): 74–97.

ALAIDAROS H, OMAR M & ROMLI R. 2019. The key factors of evaluating agile approaches: A systematic literature review. *Int. J. Supply Chain Manag.*, **8**(2): 954–964.

BARRON FH & BARRETT BE. 1996a. Decision quality using ranked attribute weights. *Manage. Sci.*, **42**(11): 1515–1523.

BARRON FH & BARRETT BE. 1996b. The efficacy of SMARTER - Simple Multi-Attribute Rating Technique Extended to Ranking. *Acta Psychol. (Amst).*, **93**(1-3): 23–36.

BECK K. 2000. *Extreme Programming Explained: Embrace Change*. Boston: Addison-Wesley.

BECK K & ANDRES C. 2005. *Extreme programming explained : embrace change*. 2 ed.. London: Addison-Wesley. 224 pp.

BECK K, BEEDLE M, BENNEKUM AV, COCKBURN A, CUNNINGHAM W, FOWLER M, GRENNING J, HIGHSMITH J, HUNT A, JEFFRIES R, KERN J, MARICK B, MARTIN RC, MELLOR S, SCHWABER K, SUTHERLAND J & THOMAS D. 2001. Manifesto for Agile Software Development. Available at: https://agilemanifesto.org/.

COCKBURN A. 2004. *Crystal Clear: A Human-Powered Methodology for Small Teams*. Boston: Addison-Wesley. 336 pp.

DE ALMEIDA AT, DE ALMEIDA JA, COSTA APCS & DE ALMEIDA-FILHO AT. 2016. A new method for elicitation of criteria weights in additive models: Flexible and interactive tradeoff. *Eur. J. Oper. Res.*, **250**(1): 179–191.

DE ALMEIDA AT, FREJ EA & ROSELLI LRP. 2021. Combining holistic and decomposition paradigms in preference modeling with the flexibility of FITradeoff. *Central European Journal of Operations Research*, **29**(1): 7–47. Available at: https://doi.org/10.1007/s10100-020-00728-z.

DEMIRTAS N, TUZKAYA UR & SEKER S. 2014. Project Management Methodology Selection Using SWOT-Fuzzy AHP. In: AO S, GELMAN L, HUKINS DWL, HUNTER A & KORSUNSKY A (Eds.), *Proc. World Congr. Eng. 2014*. pp. 1121–1126. London, U.K.: Newswood Limited. Available at: http://www.iaeng.org/publication/WCE2014/.

DIEBOLD P, THEOBALD S, WAHL J & RAUSCH Y. 2019. Stepwise transition to agile: From three agile practices to Kanban adaptation. *J. Softw. Evol. Process*, **31**(5): e2167. Available at: https://onlinelibrary.wiley.com/doi/abs/10.1002/smr.2167.

DIKERT K, PAASIVAARA M & LASSENIUS C. 2016. Challenges and success factors for large-scale agile transformations: A systematic literature review. *J. Syst. Softw.*, **119**: 87–108.

DINGSØYR T, NERUR S, BALIJEPALLY V & MOE NB. 2012. A decade of agile methodologies: Towards explaining agile software development. *J. Syst. Softw.*, **85**(6): 1213–1221. Available at: https://linkinghub.elsevier.com/retrieve/pii/S0164121212000532.

DRURY-GROGAN ML, CONBOY K & ACTON T. 2017. Examining decision characteristics and challenges for agile software development. *J. Syst. Softw.*, **131**: 248–265. Available at: https://linkinghub.elsevier.com/retrieve/pii/S0164121217301103.

EDWARDS W & BARRON FH. 1994. Smarts and smarter: Improved simple methods for multiattribute utility measurement. *Organ. Behav. Hum. Decis. Process.*, **60**(3): 306–325.

EL BEGGAR O. 2018. Multicriteria decision aid for agile methods evaluation using fuzzy PROMETHEE. *J. Softw. Evol. Process*, **30**(12).

GENTILE A. 2018. Brazilian software market scenario and trends. Available at: http://www.abessoftware.com.br/dados-do-setor/estudo-2019--dados-2018.

HAMED AMM & ABUSHAMA H. 2013. Popular agile approaches in software development: Review and analysis. In: *2013 Int. Conf. Comput. Electr. Electron. Eng.*. pp. 160–166. 2013 International Conference on Computing, Electrical and Electronic Engineering (ICCEEE). IEEE. Available at: http://ieeexplore.ieee.org/document/6633925/.

HARB Y, NOTEBOOM C & SARNIKAR S. 2015. Evaluating Project Characteristics for Selecting the Best-fit Agile Software Development Methodology: A Teaching Case. *J. Midwest Assoc. Inf. Syst.*, **1**(1). Available at: https://aisel.aisnet.org/jmwais/vol1/iss1/4.

HAZZAN O & DUBINSKY Y. 2005. Social Perspective of Software Development Methods: The Case of the Prisoner Dilemma and Extreme Programming. In: *Lect. Notes Comput. Sci.*, vol. 3556 of *Lecture Notes in Computer Science*. pp. 74–81. Springer Verlag. Available at: https://link.springer.com/chapter/10.1007/11499053_9.

HESARI S, MASHAYEKHI H & RAMSIN R. 2010. Towards a General Framework for Evaluating Software Development Methodologies. In: *2010 IEEE 34th Annu. Comput. Softw. Appl. Conf.*. pp. 208–217. 2010 IEEE 34th Annual Computer Software and Applications Conference. IEEE. Available at: http://ieeexplore.ieee.org/document/5676261/.

HICDURMAZ M. 2012. A Fuzzy Multi Criteria Decision Making Approach to Software Life Cycle Model Selection. In: *2012 38th Euromicro Conf. Softw. Eng. Adv. Appl.*. pp. 384–391. 2012 38th Euromicro Conference on Software Engineering and Advanced Applications. IEEE. Available at: http://ieeexplore.ieee.org/document/6328179/.

HODA R, SALLEH N, GRUNDY J & TEE HM. 2017. Systematic literature reviews in agile software development: A tertiary study. *Inf. Softw. Technol.*, **85**: 60–70.

HURTADO ALEGRÍA JA, BASTARRICA MC & BERGEL A. 2014. Avispa: a tool for analyzing software process models. *J. Softw. Evol. Process*, **26**(4): 434–450.

KALENDA M, HYNA P & ROSSI B. 2018. Scaling agile in large organizations: Practices, challenges, and success factors. *J. Softw. Evol. Process*, **30**(10): e1954. Available at: http://doi.wiley.com/10.1002/smr.1954.

KETTUNEN P & LAANTI M. 2005. How to steer an embedded software project: Tactics for selecting the software process model. *Inf. Softw. Technol.*, **47**(9): 587–608.

LARMAN C & BASILI VR. 2003. Iterative and incremental development: A brief history. *Computer (Long. Beach. Calif).*, **36**(6): 47–56.

MORAN A. 2015. *Managing agile: Strategy, implementation, organisation and people*. Switzerland: Springer International Publishing. 1–266 pp.

PICOT A, HESS T, HÖRNDLEIN C, KALTENECKER N, JABLONKA C, SCHREINER M, WERBIK A, BENLIAN A, NEUBURGER R & GOLD B. 2015. *The Internationalization of German Software-based Companies*. Cham: Springer International Publishing. 82 pp.. Available at: http://link.springer.com/10.1007/978-3-319-13548-9.

PRESSMAN RS & MAXIM BR. 2019. *Software Engineering: A Practitioners Approach*. McGraw-Hill Education. Available at: https://books.google.com.br/books?id=taIKxAEACAAJ.

SCHWABER K. 2004. *Agile Project Management with Scrum — Microsoft*. March. Redmond, WA: Microsoft Press. 163 pp.. Available at: https://www.microsoft.com/learning/en-us/book.aspx?id=6916.

SCHWABER K & BEEDLE M. 2001. *Agile Software Development with Scrum*. Upper Saddle River, NJ: Prentice Hall. 158 pp.. Available at: http://sutlib2.sut.ac.th/sut_contents/H129174.pdf.

SILVA VBS, SCHRAMM F & DAMASCENO AC. 2016. A multicriteria approach for selection of agile methodologies in software development projects. In: *2016 IEEE Int. Conf. Syst. Man, Cybern.*. pp. 002056–002060. 2016 IEEE International Conference on Systems, Man, and Cybernetics (SMC). IEEE. Available at: http://ieeexplore.ieee.org/document/7844542/.

STAPLETON J. 2003. *DSDM: Business focused development*. 2 ed.. London: Pearson Education.

STAVRU S. 2014. A critical examination of recent industrial surveys on agile method usage. *J. Syst. Softw.*, **94**: 87–97.

TAROMIRAD M & RAMSIN R. 2008a. An Appraisal of Existing Evaluation Frameworks for Agile Methodologies. In: *15th Annu. IEEE Int. Conf. Work. Eng. Comput. Based Syst. (ecbs 2008)*. pp. 418–427. 5th Annual IEEE International Conference and Workshop on the Engineering of Computer Based Systems (ecbs 2008). IEEE. Available at: http://ieeexplore.ieee.org/document/4492423/.

TAROMIRAD M & RAMSIN R. 2008b. CEFAM: Comprehensive evaluation framework for agile methodologies. In: *32nd Annu. IEEE Softw. Eng. Work. SEW-32 2008*. pp. 195–204. 32nd Annual IEEE Software Engineering Workshop, SEW-32. IEEE Computer Society.

VALLON R, DA SILVA ESTÁCIO BJ, PRIKLADNICKI R & GRECHENIG T. 2018. Systematic literature review on agile practices in global software development. *Inf. Softw. Technol.*, **96**: 161–180.

VAVPOTIC D & VASILECAS O. 2011. An approach for assessment of software development methodologies suitability. *Elektron. ir Elektrotechnika*, **114**(8): 107–110.

WINTERS T, MANSHRECK T & WRIGHT H. 2020. *Software Engineering at Google: Lessons Learned from Programming Over Time*. O'Reilly Media, Inc.. Available at: https://www.oreilly.com/library/view/software-engineering-at/9781492082781/.

YANG C, LIANG P & AVGERIOU P. 2016. A systematic mapping study on the combination of software architecture and agile development. *J. Syst. Softw.*, **111**: 157–184.

How to cite

SCHRAMM VB, DAMASCENO AC & SCHRAMM F. 2023. Supporting the Choice of the Best-Fit Agile Model Using FITradeoff. *Pesquisa Operacional*, **43** (spe1): e264750. doi: 10.1590/0101-7438.2023.043spe1.00264750.