

Collaborative Development Method for Improving Uncertainty Management in Agile Software Development

Maha Makkass¹, Youness Laghouaouta², Adil Anwar³

Siweb, EMI, Mohammed V University in Rabat, Rabat, Morocco^{1,3}
National Institute of Posts and Telecommunications (INPT), Rabat, Morocco²

Abstract—Software development projects frequently face uncertainty arising from evolving requirements, communication gaps, and inconsistent coordination among stakeholders. This study presents the Collaborative Development Method (CDM), an agile framework designed to strengthen collaboration, improve requirements engineering practices, and support uncertainty management throughout the software development lifecycle. CDM integrates collaborative activities, requirements traceability, user story validation, and continuous alignment between business objectives and technical implementation within a structured development process. To evaluate its effectiveness, an empirical study was conducted with software developers from Moroccan software companies. The measurement model was first validated using confirmatory factor analysis, and the proposed relationships were then examined through a Robust Least Squares regression model after diagnostic tests indicated violations of ordinary least squares assumptions. The results show that collaboration among project stakeholders, structured communication, requirements traceability, user story validation, and business–technical alignment significantly improve perceived uncertainty management. In contrast, the use of a Domain-Specific Language for structuring user stories and adapting iteration length to functionality size did not demonstrate a statistically significant effect. These findings provide empirical evidence that CDM offers a promising framework for improving coordination, requirements management, and decision-making in agile software development environments.

Keywords—Agile leadership; uncertainty management; agile software development; requirements engineering; Collaborative Development Method (CDM); software process improvement

I. INTRODUCTION

The rapid evolution of digital projects has profoundly transformed the conditions of design, coordination, and management within development teams. In this context, organizations are confronted with environments characterized by the variability of needs, the frequency of functional adjustments, the diversity of the actors involved, and the uncertainty that accompanies the translation of business expectations into technical solutions. Faced with these constraints, Agile Software Development has emerged as a privileged framework for improving team responsiveness and strengthening their adaptive capacity. However, the relevance of this approach does not lie solely in its iterative rhythm or in the application of practices derived from Scrum. It also depends on the way collaboration among stakeholders is

organized, on how requirements are clarified and monitored, and on the ability of agile leadership to support continuous mechanisms of dialogue, validation, and adjustment. From this perspective, uncertainty management constitutes a central issue, since project performance depends on the ability to reduce ambiguities without rigidifying the development process. The question, therefore, is not only whether agility makes it possible to respond more quickly to change, but also under what conditions it helps make such change more understandable, better coordinated, and more fully integrated into the overall course of the project, particularly through the integration of a collaborative agile DSL.

It is within this logic that the present study focuses on the role of the application of CDM in improving uncertainty management in a development environment. The analysis is based on the idea that the control of uncertainty depends on the quality of interactions among actors, on the structuring of requirements, and on the alignment between business objectives and technical choices. It therefore lies at the intersection of Agile Software Development, Requirements Engineering, and Software Process Improvement. Within this framework, the use of formalization mechanisms, particularly through a collaborative agile Domain-Specific Language (DSL), may be regarded as a support intended to make needs more explicit, strengthen the consistency of user stories, and facilitate exchanges among the different project stakeholders. However, the actual scope of these instruments depends on their integration into a collective dynamic sustained by agile leadership capable of articulating coordination, learning, and adaptation. The objective of this research is therefore to examine the extent to which several dimensions related to the application of CDM contribute to improving uncertainty management in software projects. Through this approach, the study seeks to better understand how agile practices can move beyond their procedural dimension to become a concrete lever for the stabilization, intelligibility, and sustainable improvement of the development process.

II. LITERATURE REVIEW

A. Agile Management and Uncertainty Management

Agile management can be presented as a mode of governance particularly well suited to software development when uncertainty affects needs, technical choices, and the conditions under which the project is carried out. Highsmith

[19] recalls that the deliverables of a project cannot be fully specified before its start and must emerge throughout its implementation, which from the outset places software development within a logic of adjustment. From the same perspective, Highsmith and Cockburn [20] show that high iteration frequencies strengthen teams' ability to cope with risk and uncertainty, particularly through the continuous integration of customer feedback. Although agile practices can implicitly contribute to risk reduction through iterative development and continuous feedback, explicit risk management mechanisms remain important for systematically identifying, assessing, and monitoring project risks [36]. Schwaber and Sutherland [40] describe Scrum as a lightweight framework aimed at generating value through adaptive solutions to complex problems, relying on empiricism, Lean Thinking, transparency, inspection, and adaptation. Waterman [47] also explains that agility, risk, and uncertainty are closely linked from the design phase onward. In the same vein, Hazzan and Dubinsky [17] present agility as a philosophy that makes it possible to deal with uncertain situations and processes of change. Agile management thus appears, in software development, as a project management framework that organizes adaptation rather than assuming the initial stability of development conditions.

This logic becomes particularly important when software development takes place in environments marked by technological and organizational turbulence. Ahmed et al. [1] emphasize that volatile, uncertain, complex, ambiguous, radical, and rapid contexts intensify the challenges associated with change management, thereby increasing the relevance of agile management. Akkaya and Ahmed [2] extend this idea by showing that the transition to Industry 5.0 reinforces the need for rapid adaptation through computing power capable of handling complex problems more effectively. Conforto et al. [7] indeed observe that agile project management is attracting increasing interest beyond software development precisely because it responds to the frequent evolution of requirements. Zaslavskaya and Zaslavskiy [49], for their part, consider that the agile project-based approach ensures the flexibility and adaptability of structures in a turbulent environment. Lill et al. [27] likewise argue that agility proposes flexibility as a response to uncertainty, whereas methods based on detailed planning are better suited to relatively stable situations. Finally, Teece et al. [43] relate this orientation to organizational agility, showing that it makes it possible to dynamically adapt the firm's capabilities in the face of risk and uncertainty in the innovation economy.

In the field of software development, agile management is not limited to absorbing external uncertainty; it also transforms the way teams interpret and address it. Fagerholm and Pagels [13] thus show that a register of uncertainty reduction appears in lean and agile values through the intention to base actions on evidence rather than on unjustified orders. Dönmez and Grote [11] indicate that agile software development teams perceive uncertainty both as a threat and as an opportunity, which directly influences their project management practices. Salin and Lundgren [39] add that agile risk management is based on the acceptance of uncertainty as a normal component of the organizational environment, as well as on the creation of a

climate of psychological safety that makes it possible to discuss openly errors, failures, and emerging uncertainties.

Levitt [26], for his part, emphasizes the delegation of authority to competent actors and the existence of shared global awareness, elements that strengthen software teams' ability to react rapidly to the unexpected.

The role of agile management appears particularly clearly in Requirements Engineering, where uncertainty directly concerns the understanding, prioritization, and evolution of needs. Logue and McDaid [29] show that agile approaches can address the uncertainty inherent in software development projects through a requirements prioritization and planning method that incorporates variability management. Their use of statistical simulation is intended to model fluctuations in requirements and estimates so as not to freeze specifications prematurely. Highsmith and Cockburn [20], for their part, emphasize that the continuous integration of customer feedback through closely spaced iterations allows the product to be adjusted more precisely to changing requirements. Ries [38] advocates an approach based on continuous innovation, iterative development, and the regular validation of hypotheses with customers, rather than on a rigid plan. Mendonça de Sá Araújo et al. [33] also show that Design Thinking practices, such as collaborative ideation, empathy, problem framing, divergent thinking, and rapid prototyping, can be structured with agile frameworks to better capture business opportunities in an uncertain environment. In software development, Requirements Engineering thus becomes an evolutionary process of clarifying needs rather than a stage of definitive requirements fixation.

This logic of continuous adjustment extends into Software Process Improvement, where improving the development process depends less on fixed standardization than on a capacity for the progressive revision of practices. Schwaber and Sutherland [40] explain that Scrum makes the relative effectiveness of existing practices visible so as to enable improvements, thereby giving agility an explicit process-improvement function. Tavares et al. [42] show that the analysis of risk management in Scrum projects makes it possible to integrate uncertainty management into the very core of agile practices. Van Oorschot et al. [46] nevertheless point out that iteration length influences the uncertainty of outcomes and that cycles that are too short or too long may produce fluctuations that are unfavorable to performance. Kreye et al. [24] confirm that the detailed planning of project means becomes of limited relevance as long as deliverables remain uncertain, which justifies significant replanning between sprints. Cooper [8] adds that the combination of agile methods and stage-gate can offer flexibility, speed, and improved communication in new product development. From a similar perspective, Lishner and Shtub [28] show that risk and uncertainty must be integrated into the very choice of development methodology, which reinforces the strategic scope of agility in improving the software process.

Agile management also affects the quality of software development through the forms of coordination and learning that it makes possible among team members. Gren et al. [15] show that teamwork, open communication, and agile planning

are strongly correlated with perceived agility in large companies, which highlights the importance of collaborative mechanisms in software environments. Tolfo et al. [45] explore the relationship between agile practices and the development of entrepreneurial skills in software development, suggesting that these practices can strengthen capabilities that are useful in evolving environments. Hendler [18] shows that, when it comes to articulating physical and digital development, close collaboration, Scrum ceremonies, and increment planning facilitate mutual adjustment and continuous learning, which is decisive in phases of high uncertainty. Azonuche and Enyejo [4], for their part, emphasize that agile risk management is based on the decentralization of responsibility, combined with the use of real-time data and predictive dashboards enabling continuous adaptation. Dobrowolski et al. [10] add that agile methodology implies a constant readiness to initiate change, to learn from each evolution, and to create value for the customer. In software development, this combination of collaboration, learning, and distributed decision-making strengthens teams' ability to maintain project continuity despite uncertainty.

B. CDM as a New Method for Agile Uncertainty Management

The Collaborative Development Method can be justified as a structured extension of the principles of Agile Software Development, but also as an operational framework of Agile Management oriented toward the control of uncertainty [30,31]. Fig. 1 indeed shows that the method does not rely solely on a succession of activities, but on a precise organization of exchanges among the Customer, the Product Owner, and the development team. From operation 1, in which the client explains business needs, the method places the clarification of business needs at the center, which is consistent with Alsaqqa et al. [3], for whom agile development is characterized by its ability to integrate changing requirements throughout the project. This interpretation is reinforced by Rasheed et al. [37], who recall that agility gives priority to interactions, collaboration with the customer, and rapid response to change, as well as by Perkusich et al. [35], who present agility as a change-driven approach based on the empirical learning of customer needs. Through operation 2, in which the Product Owner creates the User Story and updates the priorities, and then operation 3, in which the PO validates the new US and prioritized feature, Fig. 1 highlights a logic of framing, arbitration, and continuous validation that falls directly within Agile Management. This logic corresponds to the view of Gheorghe et al. [14], who define Agile Software Development as a set of values and principles oriented toward adaptive, incremental, and iterative production, carried out by self-organized teams. It also aligns with Kuhrmann et al. [25], who emphasize that agility is as much a mindset and a culture as it is a mere process. Accordingly, CDM can be understood as a form of agile management aimed at better structuring decisions, stabilizing exchanges, and, above all, preventing the emergence of ambiguities that fuel uncertainty in a development context.

The specific value of the method becomes even more apparent when one considers the way Fig. 1 organizes the transformation of need into understandable and shareable requirements. After the first validation, operation 4 indicates that the PO updates his Product Backlog; then operation 5

shows that the PO presents the highest-priority US for the next Sprint, before operation 6, in which the Dev team validates the comprehension. This sequence reflects a form of Agile Management based on progressive clarification, synchronization among actors, and the reduction of interpretive gaps between business need and technical translation. This is precisely what Chen et al. [5] support with ScrumOntoSPL, by proposing a collaborative method combining Scrum, agile engineering, and uniform requirements specification based on an ontology to reduce coordination costs. The presence in Fig. 1 of the User Story respecting DSL shows that this structuring relies on a Domain-Specific Language, which strengthens the stability of formulations, the traceability of requirements, and their shared understanding. CDM thus fits within a Requirements Engineering perspective in which formalization is not opposed to agility, but rather placed at the service of a more controlled management of interactions. This interpretation is consistent with Khan and Keung [21], who show that the improvement of practices depends on an understanding of the enabling factors and barriers that influence process evolution, as well as with Choraś et al. [6], who emphasize the importance of concrete metrics to assess and guide agile practices. Although CDM shares with ScrumOntoSPL the objective of strengthening requirements coordination within an agile setting, their structural emphasis differs. ScrumOntoSPL primarily combines Scrum with ontology-based requirements specification for software product-line reuse, whereas CDM organizes the broader collaboration cycle among the Customer, Product Owner, and development team around need clarification, prioritization, traceability, validation, prototyping, and iterative adjustment. The distinctive contribution of CDM therefore lies less in introducing an isolated requirements representation mechanism than in integrating these mechanisms within a continuous uncertainty-management process.

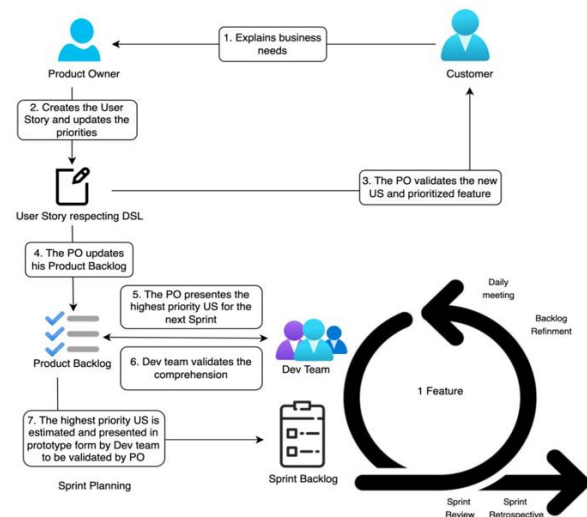


Fig. 1. Application of CDM and agile uncertainty management.

It also aligns with Marinho et al. [32], Klünder et al. [23], and Tell et al. [44], who show that, in complex or distributed environments, effective coordination requires methodological combinations capable of adjusting processes to the real constraints of collective work. Fig. 1 therefore suggests that

CDM is not only intended to manage the flow of user stories more effectively, but also to reduce upstream the sources of interpretive instability that feed project uncertainty.

The link with Uncertainty Management becomes even more explicit in the final part of Fig. 1. Operation 7 indicates that the highest priority US is estimated and presented in prototype form by the Dev team to be validated by the PO before its inclusion in the Sprint Backlog and its processing within a loop comprising Daily meeting, Backlog Refinement, Sprint Review, and Sprint Retrospective, with an iteration centered on 1 Feature. This architecture shows that CDM seeks to control uncertainty through the decomposition of needs, incremental validation, continuous feedback, and a focus on a specific functionality. In other words, Agile Management here takes the form of a sequential governance of uncertainty, in which each operation in Fig. 1 contributes to reducing part of the ambiguities likely to affect understanding, estimation, coordination, or decision-making. This interpretation is directly consistent with Gupta et al. [16], who consider risk and uncertainty management to be a critical factor in software product management. It also aligns with Ezghari and Zahi [12], who recall that estimations are affected by data inaccuracies and the limits of human reasoning; Nahar et al. [34], who emphasize the usefulness of mitigation mechanisms in the face of uncertainty; Dey and Lee [9], who show the insufficiency of requirements analysis and modeling techniques to address it properly; and Kläs and Vollmer [22], who identify specific sources of uncertainty related to scope, data quality, and model adequacy. In the same vein, Włodarski et al. [48] show that the agile approach improves team cohesion and productivity, while Sinha and Das [41] observe that it becomes particularly effective when sprints are devoted to clearly delimited functional improvements, which corresponds to the logic of 1 Feature highlighted in Fig. 1. Thus, CDM can be understood as a mechanism that articulates Collaborative Development Method, Agile Software Development, Agile Management, Scrum, Requirements Engineering, DSL, Software Process Improvement, and Uncertainty Management within the same logic of progressive development stabilization: the more needs are clarified, validated, prioritized, and tested through the operations; the more uncertainty becomes governable.

III. METHODS

A. Research Hypotheses

In light of the preceding theoretical developments, the improvement of uncertainty management in a software development environment can be understood as the result of a set of complementary mechanisms relating simultaneously to Agile Management, Requirements Engineering, and Software Process Improvement. Indeed, the literature reviewed shows that uncertainty does not stem solely from the project's external instability, but also from interpretive gaps among actors, ambiguities surrounding requirements, coordination difficulties, shortcomings in validation, and the potential mismatch between business logic and technical translation. From this perspective, the application of the Collaborative Development Method appears as a structured framework that makes it possible to better organize interactions among the client, the Product Owner, and the development team, to

strengthen the progressive clarification of needs, to improve the formalization and traceability of requirements, to support their continuous validation, and to ensure a more coherent treatment of functionalities throughout the iterations. Thus, CDM cannot be reduced to a simple operational sequence, but constitutes a mechanism of collective regulation through which exchanges, priorities, formulations, and adjustments become more intelligible and more manageable. It is on this conceptual basis that the following hypotheses are formulated:

- H1: Improving collaboration among the client, the Product Owner, and the development team has a positive effect on the perceived improvement of uncertainty management through the application of CDM.
- H2: The formalization of exchanges among stakeholders and the clarification of needs from the definition phase onward have a positive effect on the perceived improvement of uncertainty management through the application of CDM.
- H3: The use of DSL to structure the formulation of user stories has a positive effect on the perceived improvement of uncertainty management through the application of CDM.
- H4: Improving requirements traceability has a positive effect on the perceived improvement of uncertainty management through the application of CDM.
- H5: The validation of user stories and functionalities with the relevant actors has a positive effect on the perceived improvement of uncertainty management through the application of CDM.
- H6: The alignment between business objectives and technical objectives has a positive effect on the perceived improvement of uncertainty management through the application of CDM.
- H7: The adaptation of the iteration to the size of the requested functionality has a positive effect on the perceived improvement of uncertainty management through the application of CDM.

B. The Model

In order to examine empirically the factors associated with the improvement of uncertainty management in a development environment, this study adopts a statistical model linking the perception of this improvement to several dimensions derived from the application of CDM. The model specified below makes it possible to assess the role of collaboration simultaneously, the formalization of exchanges, the structuring of user stories, traceability, validation, business–technical alignment, and iterative adjustment in explaining the phenomenon under study.

$$\begin{aligned} UNCDM = & \beta_0 + \beta_1.COLLB + \beta_2.REQFM \\ & + \beta_3.DSLST + \beta_4.REQTR \\ & + \beta_5.VALUS + \beta_6.GOAAL \\ & + \beta_7.ITERF + \varepsilon \end{aligned}$$

The dependent variable UNCDM refers to the perceived improvement of uncertainty management through the application of CDM. It is measured based on six items: UNCDM1: the application of CDM helps reduce ambiguities in the definition of needs; UNCDM2: the application of CDM improves the understanding of client expectations; UNCDM3: the application of CDM facilitates the anticipation of changes related to requirements; UNCDM4: the application of CDM contributes to reducing uncertainties during development; UNCDM5: the application of CDM improves the control of requirements throughout the project; UNCDM6: the application of CDM makes the development process more predictable in the face of change. This variable thus makes it possible to assess the overall contribution of CDM to a better control of uncertainty in development. The variable COLLB measures the improvement of collaboration among the client, the Product Owner, and the development team. It is based on six items: COLLB1: CDM improves communication among the client, the Product Owner, and the development team; COLLB2: CDM fosters better information sharing among stakeholders; COLLB3: CDM facilitates coordination among the actors involved in the project; COLLB4: CDM reduces misunderstandings among stakeholders; COLLB5: CDM strengthens the involvement of actors in exchanges related to requirements; COLLB6: CDM improves the quality of interactions among project members. This variable aims to capture the extent to which the application of CDM strengthens the collective dynamic surrounding requirements management. The variable REQFM corresponds to the formalization of exchanges among stakeholders and the clarification of needs from the definition phase onward. It is measured through six items: REQFM1: CDM makes it possible to formalize exchanges among stakeholders from the beginning of the project; REQFM2: CDM improves precision in the expression of needs; REQFM3: CDM helps clarify the client's functional expectations; REQFM4: CDM reduces ambiguities in the initial formulation of requirements; REQFM5: CDM facilitates the establishment of a shared understanding of needs; REQFM6: CDM improves the structuring of discussions around requirements. This variable therefore makes it possible to assess the capacity of CDM to make the initial process of needs definition more explicit and more widely shared.

The variable DSLST refers to the use of DSL to structure the formulation of user stories. It is approached through six items: DSLST1: the use of DSL makes the formulation of user stories clearer; DSLST2: the use of DSL improves consistency in the writing of user stories; DSLST3: the use of DSL reduces errors in the interpretation of user stories; DSLST4: the use of DSL facilitates the development team's understanding of user stories; DSLST5: the use of DSL makes it possible to standardize the writing of user stories; DSLST6: the use of DSL improves the overall quality of the formulation of user stories. This variable thus measures the contribution of DSL to a better structuring of the language used in the formalization of requirements. The variable REQTR refers to the improvement of requirements traceability. It is based on six items: REQTR1: CDM facilitates the monitoring of requirements from their definition to their implementation; REQTR2: CDM makes it possible to better link requirements to the functionalities developed; REQTR3: CDM improves the identification of the

origin of expressed needs; REQTR4: CDM facilitates the tracking of changes made to requirements; REQTR5: CDM improves visibility regarding the evolution of requirements throughout the project; REQTR6: CDM strengthens continuity between initial needs and final deliverables. This variable makes it possible to assess the extent to which CDM strengthens informational continuity and the monitoring of needs over time. The variable VALUS measures the validation of user stories and functionalities with the relevant actors. It is assessed through six items: VALUS1: CDM facilitates the validation of user stories with the relevant stakeholders; VALUS2: CDM improves the verification of the conformity of functionalities with expressed needs; VALUS3: CDM makes it possible to obtain faster feedback on the proposed user stories; VALUS4: CDM promotes better acceptance of functionalities by the relevant actors; VALUS5: CDM reduces the risks of discrepancy between expectations and the functionalities developed; VALUS6: CDM improves the process of validating needs before final development. This variable therefore aims to capture the role of CDM in securing the process of validating requirements and functionalities.

The variable GOAAL corresponds to the alignment between business objectives and technical objectives. It is measured by six items: GOAAL1: CDM improves consistency between business objectives and technical choices; GOAAL2: CDM facilitates the translation of business needs into technical requirements; GOAAL3: CDM strengthens the alignment between client priorities and the work of the technical team; GOAAL4: CDM reduces the gaps between functional expectations and the proposed technical solutions; GOAAL5: CDM enables better articulation between business vision and technical development; GOAAL6: CDM improves the consideration of business objectives in technical decisions. This variable therefore makes it possible to assess the capacity of CDM to bring the functional and technical dimensions of the project closer together. The variable ITERF refers to the adaptation of the iteration to the size of the requested functionality. It may be measured based on six items: ITERF1: CDM makes it possible to adjust the duration of the iteration to the complexity of the functionality; ITERF2: CDM facilitates the breakdown of functionalities into suitable iterations; ITERF3: CDM improves the fit between workload and the content of the iteration; ITERF4: CDM reduces the gaps between the size of the functionality and iteration planning; ITERF5: CDM promotes better priority management in the iterative breakdown; ITERF6: CDM improves consistency between iteration planning and functional needs. This variable makes it possible to apprehend the extent to which the application of CDM makes the iterative pace better adjusted to the concrete characteristics of the functionalities to be developed. The items associated with all the variables in the model were measured using a five-point Likert scale, ranging from 1 "strongly disagree" to 5 "strongly agree."

C. Construction of Variables

After the initial construction of the variables, a confirmatory factor analysis was conducted in order to verify the convergent validity of the items and to refine the constructs. Only the items whose factor loadings were greater than 0.5 were retained, while the others were removed.

Convergent validity and construct reliability were assessed using standardized factor loadings, average variance extracted (AVE), and composite reliability (CR). AVE was calculated as $AVE = \frac{\sum_{i=1}^n \lambda_i^2}{n}$, where λ_i denotes the standardized factor loading of item i . Composite reliability was calculated as $CR = \frac{\sum_{i=1}^n \lambda_i^2}{\sum_{i=1}^n \lambda_i^2 + \sum_{i=1}^n (1 - \lambda_i^2)}$. Items with standardized loadings below 0.50 were excluded before the final construct scores were computed. The recalculated composite reliability coefficients range from 0.787 for DSLST to 0.913 for ITERF, indicating satisfactory construct reliability. AVE values range from 0.556 for DSLST to 0.727 for ITERF and therefore exceed the conventional 0.50 threshold for all constructs. These results support convergent validity for the retained measurement model (Table I).

TABLE I. RESULTS OF THE CONFIRMATORY FACTOR ANALYSIS FOR CONVERGENT VALIDITY AND CONSTRUCT RELIABILITY

Items	Standardized Factor Loading	Composite Reliability (CR)	AVE
UNCDM 2	0.753617	0.804706	0.583555
UNCDM 3	0.889353		
UNCDM 5	0.625922		
COLLB 1	0.620869	0.858672	0.607233
COLLB 3	0.883678		
COLLB 4	0.735819		
COLLB 6	0.849197		
REQFM 2	0.960432	0.790954	0.568807
REQFM 3	0.612614		
REQFM 5	0.639292		
DSLST 1	0.879949	0.786674	0.556081
DSLST 4	0.693511		
DSLST 5	0.642631		
REQTR 2	0.736849	0.836306	0.630673
REQTR 3	0.835277		
REQTR 5	0.807084		
VALUS 2	0.720613	0.827529	0.616331
VALUS 4	0.781365		
VALUS 6	0.848044		
GOAAL 1	0.624159	0.811529	0.595338
GOAAL 3	0.744317		
GOAAL 5	0.917840		
ITERF 1	0.924147	0.912991	0.727204
ITERF 4	0.660281		
ITERF 5	0.898398		
ITERF 6	0.900931		

The examination of composite reliability and AVE further supports the psychometric quality of the retained constructs. Composite reliability values range from 0.786674 to 0.912991, exceeding the commonly accepted threshold of 0.70. AVE values range from 0.556081 to 0.727204 and remain above the 0.50 threshold for all constructs, supporting convergent

validity. These results indicate that the retained items provide satisfactory reliability and explain an adequate proportion of variance in their respective constructs.

The retained constructs therefore demonstrate satisfactory convergent validity and composite reliability. In addition, the very low VIF values reported in the subsequent regression diagnostics indicate limited linear redundancy among the explanatory variables. Dedicated discriminant-validity criteria such as the Fornell–Larcker criterion or HTMT were not available from the retained analytical outputs and should therefore be examined in future replications to further consolidate the measurement-model assessment.

D. Data

Participants were software developers working in an organizational context in Morocco in which agile development practices were used. The survey specifically targeted developers involved in software development activities and exposed to collaborative requirements-management practices or to the application of CDM. Eligibility was therefore aligned with the purpose of the study, which required respondents to have sufficient practical exposure to agile development and collaborative requirements activities to evaluate the proposed mechanisms. A criterion-based purposive sampling approach was used to target software developers with relevant exposure to agile development and collaborative requirements-management practices. This sampling strategy was consistent with the study objective, as evaluation of the CDM-related dimensions required respondents with practical knowledge of the development context under investigation. The questionnaire also collected general participant characteristics, including age group, gender, software development experience, team size, use of agile methods, and exposure to CDM. Before completing the CDM-related items, respondents were provided with a presentation of the study, the conceptual role of CDM, and the main dimensions examined in the questionnaire. Participation was voluntary, anonymous, and confidential, and all constructs were assessed using five-point Likert-scale items ranging from strongly disagree to strongly agree.

Table II summarizes questionnaire completion and validity. A total of 250 questionnaires were completed by developers working in software development companies in Morocco. After data-quality screening, 154 questionnaires were retained for analysis, while 96 were excluded because they did not meet the criteria for statistical usability. Valid questionnaires therefore represented 61.6% of the completed questionnaires. Because the total number of individuals initially invited to participate was not available in the retained study records, an overall survey response rate could not be calculated.

TABLE II. QUESTIONNAIRE COMPLETION AND VALIDITY

Completed	Valid	Invalid	Valid questionnaires / Completed questionnaires
250	154	96	61.6 %

Table III, presenting the descriptive statistics, first shows a strong proximity between the means and medians for all variables, which indicates a distribution that is globally centered around the intermediate level of the scale. The means

range from 2.943182 for COLLB to 3.099567 for REQTR, while all medians are equal to 3.000000. This configuration suggests that the developers surveyed in software development companies in Morocco express moderate to slightly favorable perceptions regarding the contribution of CDM. The standard

deviations, ranging from 0.723932 to 0.870004, reflect a contained dispersion, indicating some variability but without excessive heterogeneity among the responses observed in the sample.

TABLE III. DESCRIPTIVE STATISTICS

	Mean	Median	Max	Min	Std. Dev.	Skewness	Kurtosis
UNCDM	3.021645	3.000000	5.000000	1.000000	0.806361	-0.223477	2.665365
COLLB	2.943182	3.000000	5.000000	1.000000	0.772549	0.096656	2.595066
REQFM	3.010823	3.000000	5.000000	1.000000	0.846556	-0.077309	2.286297
DSLST	2.978355	3.000000	5.000000	1.000000	0.789984	0.009580	2.580690
REQTR	3.099567	3.000000	5.000000	1.000000	0.870004	0.011773	2.405846
VALUS	3.012987	3.000000	5.000000	1.000000	0.772515	-0.142265	2.598047
GOAAL	2.997835	3.000000	5.000000	1.000000	0.803493	-0.034496	2.953082
ITERF	3.040584	3.000000	5.000000	1.000000	0.723932	0.113296	2.613382

The examination of the shape indicators confirms this relative stability of the distributions. The skewness coefficients remain low, ranging from -0.223477 for UNCDM to 0.113296 for ITERF, which indicates distributions close to symmetry, with slight inclinations depending on the variable. Likewise, the kurtosis coefficients vary from 2.286297 to 2.953082, values close to the reference threshold, which suggests the absence of extreme concentrations or abnormally long tails. The minimum and maximum values cover almost the entire response scale, from 1 to 5, showing that all response categories were used by the respondents, without excessive polarization in the judgments expressed.

E. Choice of Methodology

The examination of the Ramsey RESET test (Table IV), applied to the model estimated by ordinary least squares, reveals a deficiency in statistical specification. This test aims to assess the validity of the retained functional form by detecting possible omissions of relevant variables, nonlinearity effects, or, more generally, an inadequacy between the theoretical structure of the model and the relationship actually observed in the data. In this case, the results obtained show a t-statistic of 3.009505 associated with a probability of 0.0031, an F-statistic of 9.057119 also significant at the 1% level, as well as a likelihood ratio of 9.330813 with a probability of 0.0023. The convergence of these three indicators toward the same conclusion leads to the rejection of the null hypothesis of correct specification.

The examination of the Jarque-Bera normality test applied to the residuals of the model estimated by ordinary least squares reveals a significant deviation from the normality assumption. Indeed, in Fig. 2, the Jarque-Bera statistic reaches 25.54575, with a probability of 0.000003, a value well below the conventional 5% threshold. This result therefore leads to the rejection of the null hypothesis according to which the residuals follow a normal distribution. This conclusion is consistent with the descriptive characteristics of the residual series, marked by positive skewness (skewness = 0.854400) and a kurtosis coefficient slightly above the normal reference value of 3 (kurtosis = 4.030127), reflecting a distribution that is

both asymmetric and more concentrated around the mean, with relatively heavy tails.

TABLE IV. RAMSEY RESET TEST FOR MODEL SPECIFICATION (OLS)

Ramsey RESET Test Equation: UNTITLED Specification: UNCDM C COLLB REQFM DSLST REQTR VALUS GOAAL ITERF			
Omitted Variables: Squares of fitted values			
	Value	Df	Probability
t-statistic	3.009505	145	0.0031
F-statistic	9.057119	(1, 145)	0.0031
Likelihood ratio	9.330813	1	0.0023
F-test summary:			
	Sum of Sq.	Df	Mean Squares
Test SSR	0.537285	1	0.537285
Restricted SSR	9.138960	146	0.062596
Unrestricted SSR	8.601675	145	0.059322

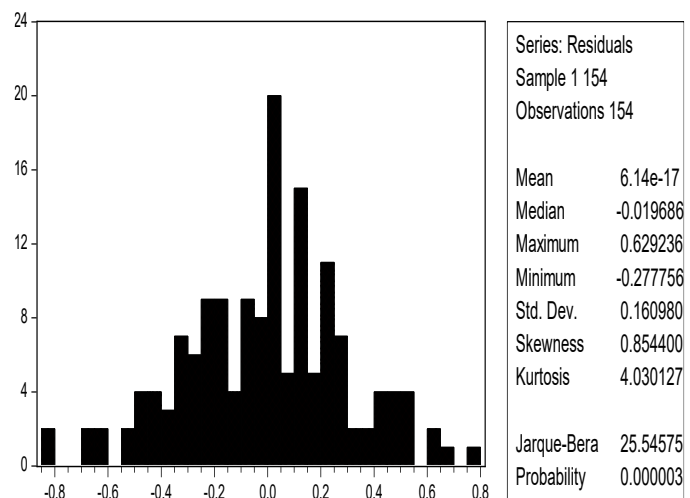


Fig. 2. Jarque-Bera normality test (OLS).

The examination of the Breusch-Pagan-Godfrey heteroskedasticity test applied to the model estimated by ordinary least squares (Table V) reveals the presence of non-constant residual variance. Indeed, the three reported statistics converge toward the same conclusion. The F-statistic stands at 2.592992 with a probability of 0.0150, the Obs*R-squared statistic reaches 17.02851 with a probability of 0.0172, while the Scaled explained SS statistic rises to 23.18845 with a probability of 0.0016. Since all these probabilities are below the conventional 5% threshold, the null hypothesis of homoskedasticity is rejected. This result means that the dispersion of the errors is not stable from one observation to another, reflecting heterogeneity in the variance of the disturbances.

TABLE V. BREUSCH-PAGAN-GODFREY HETEROSKEDASTICITY TEST (OLS)

Test statistic	Value	Degrees of freedom	Probability
F-statistic	2.592992	F(7, 146)	0.0150
Obs*R-squared	17.02851	Chi-square(7)	0.0172
Scaled explained SS	23.18845	Chi-square(7)	0.0016

In light of the diagnostics carried out on the model estimated by ordinary least squares, the use of robust regression through RLS appears methodologically justified. Indeed, the Ramsey RESET test revealed a specification deficiency, suggesting that the relationship between the variables is not fully captured by the retained linear functional form. In addition, the Jarque-Bera test led to the rejection of the normality assumption for the residuals, revealing an asymmetric and leptokurtic distribution of the errors, while the Breusch-Pagan-Godfrey test confirmed the presence of heteroskedasticity, that is, a non-constant variance of the disturbances. The combination of these three results indicates that the classical conditions ensuring the efficiency and inferential validity of OLS estimators are not fully satisfied. In this context, estimation by RLS constitutes an appropriate alternative, insofar as it makes it possible to reduce the influence of atypical observations, improve the robustness of the estimated coefficients, and produce more reliable inference when the standard assumptions of OLS are weakened.

IV. RESULTS

A. Robustness Analysis

The application of the Ramsey RESET test to the model estimated by the Robust Least Squares method indicates, this time, an improvement in statistical specification. Indeed, this test makes it possible to verify whether the retained functional form remains adequate after correcting the shortcomings identified under OLS estimation. The results in Table VI show a t-statistic of 1.149077 associated with a probability of 0.2524, an F-statistic of 1.320377 with a probability likewise equal to 0.2524, as well as a likelihood ratio of 1.395985 corresponding to a probability of 0.2374. Since all these probabilities are well above the conventional 5% threshold, the null hypothesis of correct model specification cannot be rejected. This result suggests that, under robust estimation, the explanatory model reflects the relationship under study more satisfactorily. Under robust estimation, the corresponding RESET diagnostic no

longer indicated evidence against the retained functional specification.

TABLE VI. RAMSEY RESET TEST FOR MODEL SPECIFICATION (RLS)

Ramsey RESET Test Equation: UNTITLED Specification: UNCDM C COLLB REQFM DSLST REQTR VALUS GOAAL ITERF			
Omitted Variables: Squares of fitted values			
Test	Value	Df	Probability
t-statistic	1.149077	145	0.2524
F-statistic	1.320377	(1, 145)	0.2524
Likelihood ratio	1.395985	1	0.2374
F-test summary			
Test	Sum of Sq.	Df	Mean Squares
Test SSR	2.983366	1	2.983366
Restricted SSR	330.6080	146	2.264438
Unrestricted SSR	327.6246	145	2.259480

The analysis of the variance inflation factors applied to the model estimated by the Robust Least Squares method, presented in Table VII, shows the absence of a multicollinearity problem among the retained explanatory variables. This diagnostic is based primarily on the Centered VIF values, which make it possible to assess the intensity of the linear correlations between each regressor and all the other variables in the model. In this case, all the observed values are very low and remain well below the critical thresholds commonly accepted in the statistical literature. They range from 1.015819 for COLLB to 1.053895 for VALUS, while the other variables, namely REQFM, DSLST, REQTR, GOAAL, and ITERF, also display levels very close to unity. Such a configuration indicates that each explanatory variable provides distinct information, without significant linear redundancy with the others.

TABLE VII. VARIANCE INFLATION FACTOR ANALYSIS (RLS)

Variance Inflation Factors Sample: 1 154 Included observations: 154			
Variable	Coefficient Variance	Uncentered VIF	Centered VIF
C	0.313689	21.33338	NA
COLLB	0.176150	3.562348	1.015819
REQFM	0.194895	4.433069	1.049236
DSLST	0.201126	4.507918	1.045842
REQTR	0.179376	4.424994	1.024247
VALUS	0.175523	3.927746	1.053895
GOAAL	0.178327	3.655873	1.021817
ITERF	0.191091	4.874647	1.044690

The examination of the Jarque-Bera normality test applied to the residuals of the model estimated by the Robust Least Squares method in Fig. 3 indicates that the normality hypothesis cannot be rejected. Indeed, the Jarque-Bera statistic stands at 1.540661, associated with a probability of 0.462860, a

value well above the conventional threshold of 5%. This result therefore leads to the conclusion that the distribution of the residuals is compatible with a normal distribution. This conclusion is consistent with the observed descriptive characteristics, notably a very moderate skewness (skewness = 0.149395) and a kurtosis coefficient of 2.611634, which is close to the reference value associated with normality. The residual histogram also confirms a relatively balanced distribution around the mean, without marked distortion or excessive concentration in the tails of the distribution. Under these conditions, robust estimation appears to have substantially improved the distributional behavior of the errors.

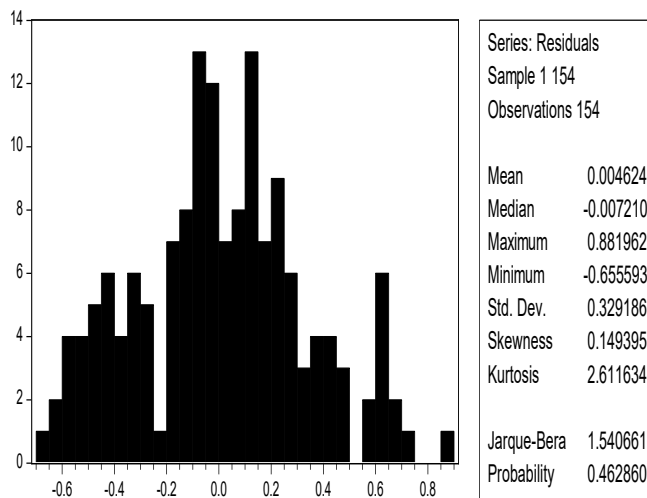


Fig. 3. Jarque-Bera normality test (RLS).

The examination of the Breusch-Pagan-Godfrey heteroskedasticity test applied to the model estimated by the Robust Least Squares method, presented in Table VIII, indicates the absence of a significant problem of residual heteroskedasticity. Indeed, the three test statistics converge toward the same conclusion. The F-statistic is equal to 0.926060 with a probability of 0.4883, the Obs*R-squared statistic stands at 6.546938 with a probability of 0.4775, while the Scaled explained SS statistic reaches 2.096089 with a probability of 0.9543. Since all these probabilities are well above the conventional 5% threshold, the null hypothesis of homoskedasticity cannot be rejected. This result means that the variance of the residuals now appears stable from one observation to another, which suggests a substantial improvement in the statistical quality of the model after the use of robust estimation. The corresponding residual diagnostics under RLS no longer indicated the irregularities observed in the initial OLS specification.

TABLE VIII. BREUSCH-PAGAN-GODFREY HETEROSKEDASTICITY TEST (RLS)

Test statistic	Value	Degrees of freedom	Probability
F-statistic	0.926060	F(7, 146)	0.4883
Obs*R-squared	6.546938	Chi-square(7)	0.4775
Scaled explained SS	2.096089	Chi-square(7)	0.9543

The examination of the influence statistics based on Covratio highlights an overall satisfactory stability of the

estimation. Fig. 4 indeed shows that the Covratio values remain concentrated within a relatively narrow interval around unity, without abrupt breaks or excessive deviations likely to indicate the existence of observations exerting a disproportionate influence on the variance-covariance matrix of the coefficients. The observed fluctuations remain moderate and follow a fairly regular overall pattern, which suggests that the hypothetical removal of one observation would not substantially modify the overall precision of the estimators. In other words, the structure of the model does not appear to depend on a few atypical points capable of significantly altering the stability of the results.

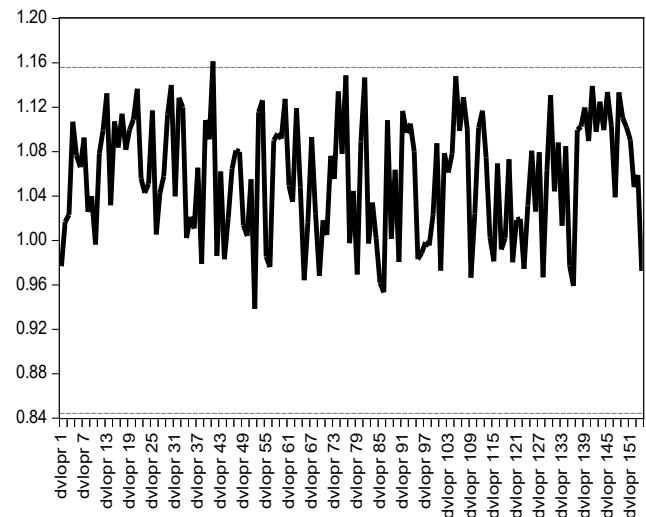


Fig. 4. Covratio influence statistics (RLS).

The initial OLS estimation revealed several statistical weaknesses through the Ramsey RESET test, the Jarque-Bera normality test, and the Breusch-Pagan-Godfrey heteroskedasticity test, indicating respectively a specification deficiency, non-normality of the residuals, and a non-constant error variance. The use of RLS estimation made it possible to substantially improve the statistical quality of the model. The results obtained after re-estimation show, through the same tests, a specification that has become acceptable, a residual distribution compatible with normality, and a more stable error structure. Finally, the examination of the Covratio influence statistics confirms the overall stability of the model.

B. Regression Results

Table IX presents the results of the estimation using the Robust Least Squares method. The estimation is based on M-estimation with Bisquare weighting, a scale founded on median-centered MAD, and Huber-type robust standard errors, which strengthens the model's resistance to disturbances likely to affect the precision of the results.

Standardized coefficients were also computed to facilitate comparison of the relative magnitude of the predictors. Among the statistically significant predictors, collaboration shows the largest standardized association with perceived uncertainty management ($\beta = 0.278$), followed by requirements formalization ($\beta = 0.238$), user-story validation ($\beta = 0.205$), requirements traceability ($\beta = 0.192$), and business-technical alignment ($\beta = 0.186$).

TABLE IX. ROBUST LEAST SQUARES ESTIMATION RESULTS

Dependent Variable: UNCDM Method: Robust Least Squares Sample: 1 154 Included observations: 154 Method: M-estimation M settings: weight=Bisquare, tuning=4.685, scale=MAD (median centered) Huber Type I Standard Errors & Covariance					
Variable	Coefficient	Std. Error	z-Statistic	Prob.	Standardized β
C	2.531772	0.684337	3.699599	***0.0003	
COLLB	0.289812	0.088981	3.257014	***0.0014	0.278
REQFM	0.226525	0.080578	2.811250	***0.0056	0.238
DSLST	0.001834	0.089058	0.020589	0.9836	0.002
REQTR	0.177941	0.080146	2.220210	**0.0279	0.192
VALUS	0.213618	0.090198	2.368323	**0.0191	0.205
GOAAL	0.186551	0.085948	2.170514	**0.0315	0.186
ITERF	0.122503	0.094912	1.290706	0.1987	0.110

*** Significant at 1%; ** Significant at 5%; * Significant at 10%.

COLLB is associated with a positive coefficient that is significant at the 1% level (p-value = 0.0014). This result shows that better collaboration among the client, the Product Owner, and the development team is associated with a perceived improvement in uncertainty management within the CDM framework. Hypothesis H1 is therefore accepted. This finding highlights that the quality of exchanges among project actors constitutes an important support for limiting misunderstandings and better framing adjustments throughout the development process. REQFM also displays a positive sign and significance at the 1% level (p-value = 0.0056). This result confirms that the formalization of exchanges among stakeholders and the clarification of needs from the outset contribute to a better perceived management of uncertainty. Hypothesis H2 is therefore accepted. This result suggests that a more explicit initial framing of expectations reduces margins of ambiguity and promotes more controlled project management. DSLST shows a positive coefficient, but it is not significant, even at the 10% level (p-value = 0.9836). Under these conditions, no statistically established effect can be retained between the use of DSL to structure the formulation of user stories and the perceived improvement of uncertainty management. Hypothesis H3 is therefore rejected. This suggests that the structuring of the language of user stories, by itself, is not sufficient to produce a perceptible change in the control of uncertainty.

REQTR emerges with a positive coefficient that is significant at the 5% level (p-value = 0.0279). This result indicates that better requirements traceability is associated with a perceived improvement in uncertainty management through the application of CDM. Hypothesis H4 is therefore accepted. It can be inferred that more rigorous monitoring of requirements, from their initial expression to their evolution, helps preserve project coherence and reduce zones of uncertainty. VALUS presents a positive coefficient that is

significant at the 5% level (p-value = 0.0191). This result shows that the validation of user stories and functionalities with the relevant actors contributes positively to the perceived improvement of uncertainty management. Hypothesis H5 is thus accepted. This result underlines the value of shared validation in bringing developed functionalities closer to the expectations actually expressed by stakeholders.

GOAAL also displays a positive coefficient that is significant at the 5% level (p-value = 0.0315). This result confirms that alignment between business objectives and technical objectives promotes a perceived improvement in uncertainty management. Hypothesis H6 is therefore accepted. This shows that better consistency between the project's functional logic and its technical translation contributes to making development more intelligible and more stable. ITERF presents a positive coefficient, but it is not significant (p-value = 0.1987). This result therefore does not allow validation of the existence of a statistically significant effect of adapting the iteration to the size of the requested functionality on the perceived improvement of uncertainty management. Hypothesis H7 is rejected. This indicates that the adjustment of the iterative pace, considered in isolation, does not appear sufficient to strengthen uncertainty control in a clearly observable manner.

V. DISCUSSION

The results first show that the perceived improvement of uncertainty management relies to a large extent on the quality of collaboration among the client, the Product Owner, and the development team. This result confirms the idea that agility does not produce its effects solely through iteration, but also through the concrete organization of interactions among stakeholders. It is therefore consistent with Highsmith and Cockburn [20], who associate adaptive capacity with the continuous integration of feedback, as well as with Schwaber and Sutherland [40], for whom Scrum is based on transparency, inspection, and adaptation in dealing with complex problems. It also aligns with Gren et al. [15], who show the importance of teamwork, open communication, and agile planning in perceived agility. In the same logic, the positive effect of the formalization of exchanges and the initial clarification of needs confirms that the reduction of uncertainty depends on the early explication of expectations rather than on the late regulation of discrepancies. This finding extends the work of Logue and McDaid [29], who emphasize that uncertainty related to requirements can be better contained when prioritization and planning avoid freezing specifications prematurely, as well as that of Ries [38], for whom the regular validation of hypotheses makes it possible to adjust the product more precisely. The results therefore indicate that, within the CDM framework, uncertainty control depends first on the ability of the actors to construct a language of exchange that is sufficiently clear to reduce ambiguities during development.

The results then show that uncertainty management is strengthened when requirements remain visible, monitored, and validated throughout the project. The positive effect of traceability confirms that uncertainty does not arise solely from the variability of demands, but also from possible breaks between the expressed need, its intermediate reformulation,

and the actual development. This result is consistent with the logic of Software Process Improvement highlighted by Schwaber and Sutherland [40], according to which practices must be made visible in order to be improved, as well as with Tavares et al. [42], who show that risk management can be integrated into Scrum practices themselves. Likewise, the positive effect of validating user stories and functionalities is consistent with Gupta et al. [16], who consider risk and uncertainty management to be a critical factor in software projects, as well as with Dey and Lee [9], who emphasize the limitations of approaches that are insufficiently equipped to deal properly with uncertain requirements. The positive effect of alignment between business objectives and technical objectives also extends the logic described in Fig. 1, where clarification, prioritization, estimation, prototyping, and validation follow one another in order to make uncertainty progressively governable. From this perspective, the results confirm that the main contribution of CDM lies in its ability to maintain continuous coherence between functional logic and technical translation, rather than in execution speed alone.

By contrast, the results qualify the scope of certain instruments when they are considered in isolation. The absence of a statistically established effect of DSL suggests that the formal structuring of user stories is not, by itself, sufficient to produce a perceptible improvement in uncertainty management. This result does not necessarily contradict the expected contributions of a structured language, but it shows that its effectiveness remains conditional on the way it is embedded in collective coordination practices. From this perspective, the results invite a more cautious reading of the promises associated with formalization tools and are consistent with the idea defended by Kuhrmann et al. [25], according to which agility is as much a matter of mindset and culture as of process. Similarly, the absence of a significant effect of adapting the iteration to the size of the functionality shows that the iterative pace, taken in isolation, does not constitute a sufficient lever for uncertainty control. This observation is consistent with Van Oorschot et al. [46], who show that iteration length influences outcomes without automatically guaranteeing greater stability, as well as with Lishner and Shtub [28], who recall that risk and uncertainty must be integrated into the broader choice of methodology. Overall, the results therefore confirm that uncertainty management depends less on a particular technical device than on a coherent combination of collaboration, clarification, traceability, validation, and alignment, supported by agile leadership capable of articulating these dimensions within the CDM framework.

VI. CONCLUSIONS AND FUTURE WORK

This study shows that improving uncertainty management in software projects depends less on the declared adoption of an agile approach than on the way its principles are effectively translated into the organization of work. The results highlight that the reduction of ambiguities becomes more tangible when actors build a shared understanding of needs, sustain continuous exchanges, and maintain close coordination throughout development. From this perspective, Agile Software Development cannot be reduced to a succession of iterations or to a simple sequence of ceremonies inspired by

Scrum. Its effectiveness rests primarily on the quality of collaboration, on the circulation of information, and on the capacity of agile leadership to create a framework for dialogue, adjustment, and collective regulation. Uncertainty management thus appears as a socially constructed process, in which uncertainties are contained not through rigid control, but through frequent interactions, shared validation, and a better articulation between client expectations and development choices. Within this interpretation, CDM may be understood not merely as a procedural tool, but as a new form of agile management, insofar as it structures the relationships among the Customer, the Product Owner, and the development team around progressive clarification, prioritization, validation, and iterative adjustment. In that sense, CDM extends agile management by providing a more explicit operational architecture through which coordination becomes more readable and uncertainty more governable. The findings therefore suggest that agility produces its strongest effects when it reinforces collective coherence around requirements, so that changes become more intelligible, more openly discussed, and more easily integrated into the trajectory of the project.

The analysis also shows that the most robust contribution to project stabilization comes from practices that strengthen Requirements Engineering within an agile framework. When needs are clarified, monitored, and validated continuously, uncertainty control becomes more credible and the project gains in coherence between business logic and technical translation. In a Scrum-inspired environment, the value of the process lies less in the speed of iteration itself than in the quality of the collective trade-offs it makes possible. In this respect, the use of DSL, particularly in the form of a collaborative agile DSL, does not appear sufficient on its own to significantly transform uncertainty management, which recalls that formalization tools replace neither dialogue nor shared interpretation. The essential contribution of the study therefore concerns Software Process Improvement, understood as the joint strengthening of communication, validation, alignment, and collective learning. The general conclusion is that agile leadership may provide an enabling context for making agility a durable lever for the control of uncertainty, while CDM can be regarded as a structured and emerging form of agile management specifically oriented toward the progressive stabilization of software development under uncertainty. This study nevertheless presents several limitations. First, it is based on cross-sectional perceptual data collected from developers working in software development companies in Morocco, which limits both causal interpretation and the generalizability of the findings to other institutional or organizational contexts. Second, the analysis relies on self-reported Likert-scale measures and on a perceived improvement in uncertainty management rather than on objective project performance indicators such as delivery delays, defect rates, rework, or requirements volatility. Because predictor and outcome measures were collected from the same respondents at a single point in time, common method variance cannot be completely excluded. Future studies should combine multiple data sources, temporal separation, or objective project indicators to assess the robustness of the observed relationships. Third, the sample retained for the empirical

analysis remains limited, and the study does not compare CDM with other agile configurations or hybrid project management approaches. Finally, the empirical design does not capture the longitudinal dynamics through which uncertainty evolves across sprints and project phases. Future research could therefore extend the analysis through comparative, longitudinal, and multi-actor designs, while also examining more directly whether CDM, as a new form of agile management, improves observable project outcomes in different software development settings.

ACKNOWLEDGMENT

The authors would like to thank all software developers and industry professionals who voluntarily participated in the survey and contributed valuable insights to this study. Their participation made the empirical evaluation of the proposed Collaborative Development Method (CDM) possible.

REFERENCES

- [1] Ahmed, J., Mrugalska, B., & Akkaya, B. (2022). Agile management and VUCA 2.0 (VUCA-RR) during Industry 4.0. In B. Akkaya, M. W. Guah, K. Jermisittiparsert, H. Bulińska-Stangrecka, & Y. Kaya (Eds.), *Agile management and VUCA-RR: Opportunities and threats in Industry 4.0 towards Society 5.0* (pp. 13–26). Emerald Publishing Limited. <https://doi.org/10.1108/978-1-80262-325-320220002>
- [2] Akkaya, B., & Ahmed, J. (2022). VUCA-RR toward Industry 5.0. In B. Akkaya, M. W. Guah, K. Jermisittiparsert, H. Bulińska-Stangrecka, & Y. K. Koçyiğit (Eds.), *Agile management and VUCA-RR: Opportunities and threats in Industry 4.0 towards Society 5.0* (pp. 1–11). Emerald Publishing Limited. <https://doi.org/10.1108/978-1-80262-325-320220001>
- [3] Alsaqqa, S., Sawalha, S., & Abdel-Nabi, H. (2020). Agile Software Development: Methodologies and Trends. *International Journal of Interactive Mobile Technologies (ijim)*, 14(11), pp. 246–270. <https://doi.org/10.3991/ijim.v14i11.13269>
- [4] Azonuche, T. I., & Enyejo, J. O. (2025). Adaptive risk management in agile projects using predictive analytics and real-time velocity data visualization dashboard. *International Journal of Innovative Science and Research Technology*, 10(4), 2032–2047. <https://doi.org/10.38124/ijisrt/25apr2002>
- [5] Chen, J., Huang, R., Jiang, Y., Pu, C., Huang, X., Sun, X., & Liu, Y. (2023). ScrumOntoSPL: Collaborative method of agile product line engineering for software resource reuse. *Electronics*, 12(11), 2421. <https://doi.org/10.3390/electronics12112421>
- [6] Choraś, M., Springer, T., Kozik, R., López, L., Martínez-Fernández, S., Ram, P., Rodríguez, P., & Franch, X. (2020). Measuring and improving agile processes in a small-size software development company. *IEEE Access*, 8, 78452–78466. <https://doi.org/10.1109/ACCESS.2020.2990117>
- [7] Conforto, E. C., & Amaral, D. C. (2016). Agile project management and stage-gate model—A hybrid framework for technology-based companies. *Journal of Engineering and Technology Management*, 40, 1–14. <https://doi.org/10.1016/j.jengtecman.2016.02.003>
- [8] Cooper, R. G. (2016). Agile–Stage-Gate hybrids. *Research-Technology Management*, 59(1), 21–29. <https://doi.org/10.1080/08956308.2016.1117317>
- [9] Dey, S., & Lee, S.-W. (2021). Multilayered review of safety approaches for machine learning-based systems in the days of AI. *Journal of Systems and Software*, 176, Article 110941. <https://doi.org/10.1016/j.jss.2021.110941>
- [10] Dobrowolski, Z., Ledzianowski, J., & Dobrowolska, M. (2021). Towards to agile management control systems at the university: Preliminary research. *European Research Studies Journal*, 24(1), 1220–1229. <https://www.um.edu.mt/library/oar/handle/123456789/81395>
- [11] Dönmez, D., & Grote, G. (2018). Two sides of the same coin – how agile software development teams approach uncertainty as threats and opportunities. *Information and Software Technology*, 93, 94–111. <https://doi.org/10.1016/j.infsof.2017.08.015>
- [12] Ezghari, S., & Zahi, A. (2018). Uncertainty management in software effort estimation using a consistent fuzzy analogy-based method. *Applied Soft Computing*, 67, 540–557. <https://doi.org/10.1016/j.asoc.2018.03.022>
- [13] Fagerholm, F., & Pagels, M. (2014). Examining the structure of lean and agile values among software developers. In *Agile Processes in Software Engineering and Extreme Programming (XP 2014)* (pp. 218–233). Springer. https://doi.org/10.1007/978-3-319-06862-6_15
- [14] Gheorghe, A.-M., Gheorghe, I. D., & Iatan, I. L. (2020). Agile software development. *Informatica Economică*, 24(2), 90–100. <https://doi.org/10.24818/issn14531305/24.2.2020.08>
- [15] Gren, L., Torkar, R., & Feldt, R. (2017). Group development and group maturity when building agile teams: A qualitative and quantitative investigation at eight large companies. *Journal of Systems and Software*, 124, 104–119. <https://doi.org/10.1016/j.jss.2016.11.024>
- [16] Gupta, R. K., Shukla, S., Rajan, A. T., & Aravind, S. (2022). Strategies for effective product roadmap development and execution in data analytics platforms. *International Journal for Research Publication and Seminar*, 13(1). <https://jrps.shodhsagar.com/index.php/j/article/view/1515>
- [17] Hazzan, O., & Dubinsky, Y. (2014). *Agile anywhere: Essays on agile projects and beyond*. Springer Cham. <https://doi.org/10.1007/978-3-319-10157-6>
- [18] Hendler, S. (2019). Digital-physical product development: A qualitative analysis. *European Journal of Innovation Management*, 22(2), 315–334. <https://doi.org/10.1108/EJIM-01-2018-0026>
- [19] Highsmith, J. (2009). *Agile project management: Creating innovative products* (2nd ed.). Addison-Wesley. <https://dl.acm.org/doi/10.5555/1594896>
- [20] Highsmith, J., & Cockburn, A. (2001). Agile software development: The business of innovation. *Computer*, 34(9), 120–127. <https://doi.org/10.1109/2.947100>
- [21] Khan, A. A., & Keung, J. (2016). Systematic review of success factors and barriers for software process improvement in global software development. *IET Software*, 10(5), 125–135. <https://doi.org/10.1049/iet-sen.2015.0038>
- [22] Kläs, M., & Vollmer, A. M. (2018). Uncertainty in machine learning applications: A practice-driven classification of uncertainty. In B. Gallina, A. Skavhaug, E. Schoitsch, & F. Bitsch (Eds.), *Computer safety, reliability, and security: SAFECOMP 2018* (pp. 431–438). Springer. https://doi.org/10.1007/978-3-319-99229-7_36
- [23] Klünder, J., Karajic, D., Tell, P., Karras, O., Munkel, C., Münch, J., MacDonell, S. G., Hebig, R., & Kuhrmann, M. (2020). Determining context factors for hybrid development methods with trained models. In *Proceedings of the International Conference on Software and System Processes* (pp. 61–70). Association for Computing Machinery. <https://doi.org/10.1145/3379177.3388898>
- [24] Kreye, M. E., Ramírez Hernández, T., & Eppinger, S. (2025). Translating agile management practices into a traditional industry context. *Journal of Product Innovation Management*, 42(2), 444–467. <https://doi.org/10.1111/jpim.12760>
- [25] Kuhrmann, M., Tell, P., Hebig, R., Klünder, J., Münch, J., Linssen, O., Pfahl, D., et al. (2022). What makes agile software development agile? *IEEE Transactions on Software Engineering*, 48(9), 3523–3537. <https://doi.org/10.1109/TSE.2021.3099532>
- [26] Levitt, R. E. (2011). Towards project management 2.0. *Engineering Project Organization Journal*, 1(3), 197–210. <https://doi.org/10.1080/21573727.2011.609558>
- [27] Lill, P. A., Wald, A., & Gleich, R. (2020). Agility and the role of project: Internal control systems for innovation project performance. *International Journal of Innovation Management*, 24(7), Article 2050064. <https://doi.org/10.1142/S1363919620500644>
- [28] Lishner, I., & Shtub, A. (2023). Enhancing strategic planning of projects: Selecting the right product development methodology. *Information*, 14(12), 632. <https://doi.org/10.3390/info14120632>
- [29] Logue, K., & McDaid, K. (2008). Handling uncertainty in agile requirement prioritization and scheduling using statistical simulation. In

- Agile 2008 Conference (pp. 73–82). IEEE. <https://doi.org/10.1109/Agile.2008.79>
- [30] Makkass, M., Laghouaouta, Y., & Anwar, A. (2025). Agile method for collaborative development process by exploring use limits of Scrum. In M. Ben Ahmed, B. A. Abdelhakim, I. R. Karaş, & K. Ben Ahmed (Eds.), *Innovations in Smart Cities Applications Volume 8* (Lecture Notes in Networks and Systems, Vol. 1310, pp. 475–489). Springer. https://doi.org/10.1007/978-3-031-88653-9_47
- [31] Makkass, M., Laghouaouta, Y., & Anwar, A. (2025). The role of agile leadership in team motivation within software development projects: A study case of multinational companies in Morocco. *Journal of Computer Science*, 21(11), 2735–2744. <https://doi.org/10.3844/jcssp.2025.2735.2744>
- [32] Marinho, M., Noll, J., Richardson, I., & Beecham, S. (2019). Plan-driven approaches are alive and kicking in agile Global Software Development [Preprint]. arXiv. <https://doi.org/10.48550/arXiv.1906.08895>
- [33] Mendonça de Sá Araújo, C. M., Miranda Santos, I., Dias Canedo, E., & Favacho de Araújo, A. P. (2019). Design thinking versus design sprint: A comparative study. In A. Marcus & W. Wang (Eds.), *Design, user experience, and usability. Design philosophy and theory* (pp. 291–306). Springer. https://doi.org/10.1007/978-3-030-23570-3_22
- [34] Nahar, N., Zhou, S., Lewis, G., & Kästner, C. (2022). Collaboration challenges in building ML-enabled systems: Communication, documentation, engineering, and process. In *Proceedings of the 44th International Conference on Software Engineering (ICSE '22)* (pp. 413–425). Association for Computing Machinery. <https://doi.org/10.1145/3510003.3510209>
- [35] Perkusich, M., Chaves e Silva, L., Costa, A., Ramos, F., Saraiva, R., Freire, A., Dilozenzo, E., Dantas, E., Santos, D., Gorgônio, K., Almeida, H., & Perkusich, A. (2020). Intelligent software engineering in the context of agile software development: A systematic literature review. *Information and Software Technology*, 119, Article 106241. <https://doi.org/10.1016/j.infsof.2019.106241>
- [36] Odzaly, E. E., Greer, D., & Stewart, D. (2018). Agile risk management using software agents. *Journal of Ambient Intelligence and Humanized Computing*, 9, 823–841. <https://doi.org/10.1007/s12652-017-0488-2>
- [37] Rasheed, A., Zafar, B., Shehryar, T., Aslam, N. A., Sajid, M., Ali, N., Dar, S. H., & Khalid, S. (2021). Requirement engineering challenges in agile software development. *Mathematical Problems in Engineering*, 2021, Article 6696695, 1–18. <https://doi.org/10.1155/2021/6696695>
- [38] Ries, E. (2011). How today's entrepreneurs use continuous innovation to create radically successful businesses. *The lean startup*. <https://ohkliberec.cz/wp-content/uploads/2013/10/START-UP.pdf>
- [39] Salin, H., & Lundgren, M. (2022). Towards agile cybersecurity risk management for autonomous software engineering teams. *Journal of Cybersecurity and Privacy*, 2(2), 276–291. <https://doi.org/10.3390/jcp2020015>
- [40] Schwaber, K., & Sutherland, J. (2020). *Der visuelle Scrum Guide* (November 2020). <https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-German.pdf>
- [41] Sinha, A., & Das, P. (2021). Agile methodology vs. traditional waterfall SDLC: A case study on quality assurance process in software industry. In *2021 5th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)* (pp. 1–4). IEEE. <https://doi.org/10.1109/IEMENTech53263.2021.9614779>
- [42] Tavares, B., da Silva, C. E. S., & de Souza, A. D. (2019). Risk management analysis in Scrum software projects. *International Transactions in Operational Research*, 26(5), 1884–1905. <https://doi.org/10.1111/itor.12401>
- [43] Teece, D. J., Peteraf, M., & Leih, S. (2016). Dynamic capabilities and organizational agility: Risk, uncertainty, and strategy in the innovation economy. *California Management Review*, 58(4), 13–35. <https://doi.org/10.1525/cmr.2016.58.4.13>
- [44] Tell, P., Klünder, J., Küpper, S., Raffo, D., MacDonell, S. G., Münch, J., Pfahl, D., Linssen, O., & Kuhrmann, M. (2019). What are hybrid development methods made of? An evidence-based characterization. In *Proceedings of the International Conference on Software and Systems Process (ICSSP 2019)* (pp. 105–114). IEEE. <https://doi.org/10.1109/ICSSP.2019.00022>
- [45] Tolfo, C., Wazlawick, R. S., Ferreira, M. G. G., & Forcellini, F. A. (2018). Agile practices and the promotion of entrepreneurial skills in software development. *Journal of Software: Evolution and Process*, 30(6), e1945. <https://doi.org/10.1002/smr.1945>
- [46] van Oorschot, K. E., Sengupta, K., & Van Wassenhove, L. N. (2018). Under pressure: The effects of iteration lengths on agile software development performance. *Project Management Journal*, 49(6). <https://doi.org/10.1177/8756972818802714>
- [47] Waterman, M. (2018). Agility, risk, and uncertainty, Part 1: Designing an agile architecture. *IEEE Software*, 35(2), 99–101. <https://doi.org/10.1109/MS.2018.1661335>
- [48] Włodarski, R., Poniszewska-Marańda, A., & Falleri, J. (2020). Comparative case study of plan-driven and agile approaches in student computing projects. In *2020 28th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE. <https://doi.org/10.23919/SoftCOM50211.2020.9238196>
- [49] Zaslavskaya, K., & Zaslavskiy, V. (2024). Implementation of the project approach as a tool for ensuring flexibility and adaptability of business structures in a turbulent environment. *Ukrainian Journal of Applied Economics and Technology*, 2024(3), 181–185. <https://doi.org/10.36887/2415-8453-2024-3-31>