International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Enhancing Quality and Productivity in Software Engineering: An Ontology-Driven Prescriptive Agile Framework

Deepa T.[1], Dr. Thaddues S.[2], Dr. Dhamodharan G.[3]

Submitted: 03/10/2023 **Revised**: 21/11/2023 **Accepted**: 01/12/2023

Abstract: The prescriptive agile framework is a novel approach that leverages an integrated process ontology, involving agile Scrum and OpenUp methodologies. The objective is to enhance both productivity and quality within software development projects where fresh developers are engaged. Key Process elements of OpenUP and agile Scrum are used to construct the ontology. The ontology is defined and instantiated with software projects having the complete user stories. Subsequently a three-sprint project experiment is conducted. The metrics of Earned Value (EV) and Code Maintainability are used to gauge productivity and quality. This research contributes to improving productivity and code quality in software engineering particularly when there are many amateurs.

Keywords: *Integrated Process Framework, Ontology, Productivity, Quality, Software Team, Code Development, Validation, Earned Value Management, Code Maintainability Index.*

1. Introduction

Agile engineering methods face several challenges that need to be addressed for successful implementation [1]. One of the key challenges in software engineering is the impact on productivity and quality when a development team comprises developers with varying levels of experience, especially in agile models. The self-organization aspect of agile methodologies becomes challenging when inexperienced team members struggle to contribute effectively or make informed decisions [2]. Additionally, the reduced emphasis on detailed requirement documentation in agile can be problematic for inexperienced developers who need more explicit guidance. As the annual "State of Agile" survey [3] emphasizes, it is vital to recognize these challenges and offer suitable support and mentorship to overcome their effects on productivity and quality in diverse teams.

Combining the key process elements of prescriptive and agile scrum models, a better software engineering environment can be provided for agile, heterogeneous teams. Using an integrated process ontology of selected elements of agile and prescribed process models, this research paper presents a framework for an integrated process ontology. The ontology facilitates effective knowledge organization, querying, and reasoning within the domain of software development processes. Additionally, the paper highlights how projects can be instantiated using this framework and validates the effectiveness of the framework.

This research article demonstrates how integrating OpenUP's process elements (Activities, work products such as SRS, Glossary, and Vision documents) into user stories improves software engineering quality and productivity. By providing developers with more specific information for each story, they can create more precise code. This leads to increased productivity and improved code quality. The research showcases how this framework positively impacts software engineering processes, optimizing code generation and enhancing project results.

2. Literature Review

Many researchers have explored the utilization of ontologies for process definition and enactment. Liao et al. establish an ontology called SPO for software processes - the activities and tasks in software product development, encompassing common concepts and relations like process areas, goals, outcomes, and practices. [4]. Oveh, R.O et al. introduce an ontology-based method for conceptually articulating software processes, utilizing ontology to grasp universal and specific aspects and connections within software process subdomains like planning, analysis, design, implementation, testing, deployment, and maintenance. The research assesses the SPO ontology's quality and consistency using the OntoClean method, demonstrating its value in software process modeling and representation. [5, 6] . W. A. Ortega-Ordoñez et al propose an ontology called OntoAgile, which aims to suggest a common and consistent terminology that allows sharing the knowledge generated around the implementation of agile approaches in software processes in a generic and formal way. The paper also

[1]*Research Scholar,* [2]*Associate Professor,* [3]*Assistant Professor*
[123]*PG & Research Department of Computer Science,*
Don Bosco College (Co-Ed), Guezou Nagar, Yelagiri Hills, Tamilnadu-635854.
(Affiliated to Thiruvalluvar University)
[1]*margaret.deepa23@gmail.com,*[2]*thad@boscoits.com,*
[3]*haidhamo@gmal.com*

shows how OntoAgile can facilitate the assessment of the agility of the software processes from the identification of the relationships between the elements of the software processes and the agile principles and values [7].

The concept of the framework in this paper is derived from the ontology-driven model for Knowledge-Based Software Engineering (OSEE) [8]. It is a model that integrates knowledge acquisition activities with software development, so that developers can be both consumers and producers of software engineering knowledge. The model uses ontology to represent explicit knowledge of the application domain and software processes, as well as instances of tacit knowledge from various software projects. The goal is to build a knowledge base that can provide context-sensitive assistance to software developers based on accumulated and structured knowledge. The preliminary results have shown that this model can enhance domain modeling and process improvement in software engineering.

3. Methodology

A quantitative approach is employed to understand the relationship between the process ontology and the outcome variables (productivity and quality). The experiment involves two projects split into three sprints, each containing user stories of varying complexity levels. The Integrated Process Ontology defines the structural framework for organizing and representing knowledge about Agile Scrum and OpenUp methodologies. This process ontology incorporates not just the activities and work products, but also their interrelationships and the roles associated with them.

Having defined the ontology, it is instantiated with user stories of the target-projects. This involves mapping project components, such as user stories, sprints, and work products, to their corresponding ontology entities. This serves to make the ontology actionable and testable in a real-world environment.

The core of the research methodology is a three-sprint project experiment. The requirement given to each sprint are at different range

- For the Sprint 1: Only the Card and Confirmation of user stories are given
- For the Sprint 2: The Conversation of user stories, which details the interaction between the product owner and development team regarding the user story requirements are included.
- For the Sprint 3: 3C user stories are mapped to related work products based on the prescriptive process model. This links the user stories to tangible outputs in the development process.

The effectiveness of the instantiated ontology and the overall agile framework is evaluated using the following metrics: Productivity is assessed using Earned Value (EV), a widely recognized project management tool that measures project performance and progress in a single integrated system. Quality is evaluated through Code Maintainability. It measures how easily software can be understood, repaired, or enhanced.

Prescriptive Agile Process Framework

Previous research proposed and validated an ontology-driven software engineering environment framework [8]. The process ontology is redefined by merging agile scrum and prescriptive process models. By enriching agile scrum user stories with associated work products from the prescriptive model, we enhance detail for better code generation. This approach improves developer productivity and code quality, regardless of experience level.

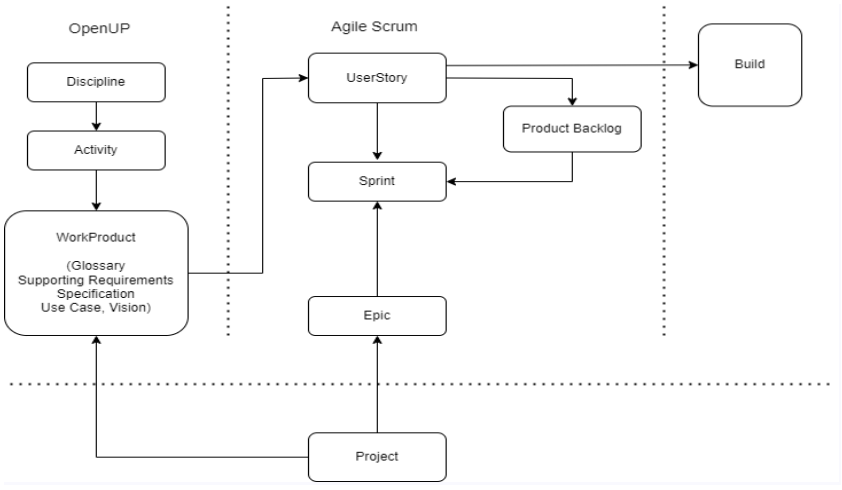

Fig. 1. Prescriptive Agile Framework

A software process based on an ontology has been developed that integrates the Open Unified Process (OpenUP) and Scrum methodologies. Key components of OpenUP, such as disciplines, activities, and guidelines, along with corresponding work products like the Architecture Notebook, Software Requirement Specifications, Glossary, Vision, and Use Cases, are mapped to the user stories in the experiment. These work products are created following OpenUP's prescribed activities and guidelines for a project.

In contrast, Scrum operates within fixed-length iterations known as sprints, typically lasting 2 or 4 weeks. With Scrum, the project's features are examined, and the project itself is divided into large units of work called epics. These epics are further subdivided into user stories. All user stories from the project are gathered into a product backlog. A subset of this backlog is selected for execution within a sprint, known as the Sprint Backlog.

The work products, derived from following the OpenUP guidelines, are supplied as auxiliary information for the development of the user stories. This integrated approach ensures the production of high-quality software builds.

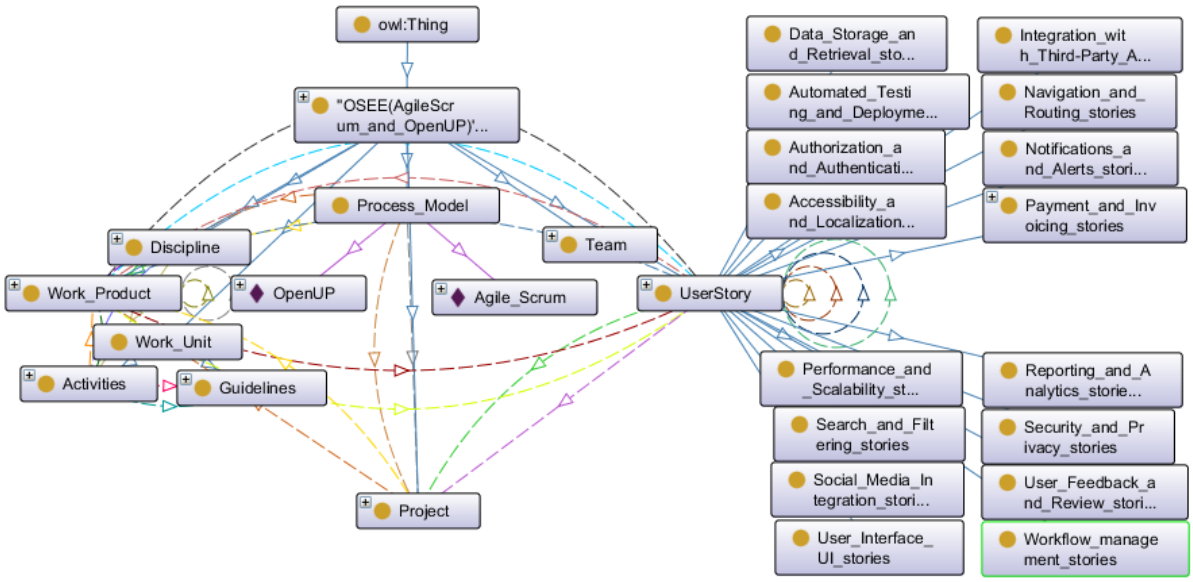

Fig. 2. Ontology based - PrescriptiveAgile Framework

The ontology framework in figure 2 integrates the software development process, combining the methodologies of Scrum and OpenUP. The concepts and principles derived from both methodologies are represented as classes and interconnected through relationships within the ontology.

At the core of the framework is the root class "OSEE" which serves as the overarching entity encapsulating the entire ontology of both agile and prescriptive models. It acts as a foundation for organizing and structuring the software development process. User stories are categorized into ten groups and presented as a subclass of user stories. The Classes, SubClasses and Individuals encompassed in OSEE are given in Table 1.

Table 1*:* Classification of Entities within OSEE Ontology for Agile Scrum and OpenUP

Classes	Subclasses	Individuals
Thing	OSEE(AgileScrum_and_OpenUP)	
	Discipline	Architecture, Requirements
	Process_Model	Agile_Scrum, OpenUP

	Project	Higrade, Edustar
	Team	Developers, ProductOwner, ScrumMaster
	UserStory	User_Interface_(UI)_stories, Accessibility_and_Localization_stories, Authorization_and_Authentication_stories, Automated_Testing_and_Deployment_stories, Data_Storage_and_Retrieval_stories, Integration_with_Third-Party_APIs_stories, Navigation_and_Routing_stories, Notifications_and_Alerts_stories, Payment_and_Invoicing_stories, Performance_and_Scalability_stories, Reporting_and_Analytics_stories, Search_and_Filtering_stories, Security_and_Privacy_stories, Social_Media_Integration_stories, User_Feedback_and_Review_stories, Workflow_management_stories
	Work_Product	Architecture_Notebook, Build, Design, Developer_Test, Glossary, Implementation, Iteration_Plan, Project_Plan, Risk_List, Supporting_Requirements_Specification, Test_Case, Test_Log, Test_Script, Use-Case_Model, Use_Case, Vision, Work_Items_List
	Work_Unit	Activities, Guidelines
	Activities	ArchitectureActivities, Configuration_and_change_managementActivities, DevelopmentActivities, Project_ManagementActivities, RequirementActivities, TestActivities
	Guidelines	ArchitectureGuidelines, Configuration_and_change_managementGuidelines, DevelopmentGuidelines, Project_ManagementGuidelines, RequirementsGuidelines, TestGuidelines

By structuring the ontology with these classes, subclasses, Object Property and Data Property, the framework effectively captures the integration of Scrum and OpenUP methodologies, providing a comprehensive representation of the software development process and its associated elements.

4.1 Mapping User Stories and Work Products

Enriching user stories with associated work products is of paramount importance as it allows for the provision of detailed information about the implied software. By incorporating relevant work products such as Software Requirements Specification (SRS), Glossary, and Vision documents, the user stories gain context and depth. This enrichment aids in providing a comprehensive understanding of the requirements, functionality, and expectations associated with the user stories. It helps to reduce ambiguity, clarify the desired outcomes, and mitigate potential misunderstandings during the development process. Additionally, detailed work products contribute to improved estimation accuracy, task allocation, and overall productivity of the development team. They serve as valuable references for ensuring that the implemented software aligns with the specified standards and requirements, thereby enhancing the quality of the final product.

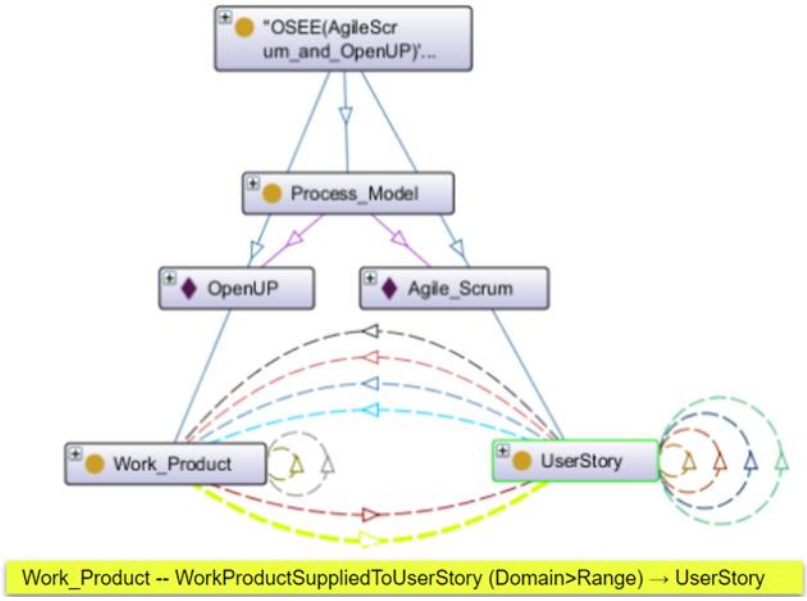

Fig.3. Mapping UserStory and WorkProducts

Fig 3 shows that the Work_Product and UserStory classes are mapped through the object property "WorkProductSuppliedToUserStory" in the integrated framework ontology. This object property establishes a relationship between instances of the Work_Product class and instances of the UserStory class.

By utilizing these object properties, the ontology establishes relationships between the entities, enabling a representation of how the "ProductBackLog," "Sprint," and "UserStory" are interconnected in the software development process.

4. Experiment and Results

Two software projects, each developed using distinct technologies, were subjected to an in-depth observation and analysis in this study. The projects, labeled Project A and Project B, were selected with three unique sprints each.

Project A, built with Python in the Odoo Framework, was handled by a team of six developers, four of whom had less than two years of experience. The three sprints for this project were closely matched in terms of Story Points (SP), a measure of a User Story's complexity. Specifically, the first, second, and third sprints scored 109 SP, 107 SP, and 110 SP, respectively.

On the other hand, Project B was constructed with AngularJS by a smaller team of three developers, two of whom were relatively new with less than two years of experience. This project's sprints were weighted differently at 58, 61, and 63 SP respectively.

The execution of the sprints for both projects proceeded through three stages. In the initial stage, sprints were based solely on supplied documents and information, referred to as the Card and Confirmation. During the second stage, additional information was introduced in the form of Card Conversation and Confirmation, aligning with the Agile process's 3C format. Finally, the third stage involved the use of OpenUP work products, which offer detailed guidance for the execution of each User Story. The sample Sprint Backlog with three Userstory is provided in Table 2.

Table 2: Exemplary Selection from Sprint Backlog with User Story Descriptions and Story Points

US-ID	USERSTORY	SP
FMS01	As an Accountant / Office admin I must be able to Configure Bank Accounts detail for Students so that Bank Account can be selected by the student while paying fee .	5
FMS02	As an Office Admin / Fee Admin / System admin, I must be able to Configure the Fee Categories For the Students , So that While selecting Fee Account the Fee Categories can be selected for paying the fees.	15
FMS03	As an Office Admin / Fee Admin , I must be able to Configure the Different Fee Heads for Students , So that Payment can be Enabled and refund can be made for students .	5

The varying levels of requirement detail provided for the three Sprints are outlined below.

Sprint1- Card and Confirmation

CardID - FMS01

Card: As an Accountant Office admin I must be able to Configure **Bank Accounts** for Students so that Bank Account can be selected by the student while paying fee .

Confirmation:

As an Accountant Office admin: Given When I select Bank Account sub menu from Configuration main menu Then I must be able to Configure the Bank Account details from the following fields,

Account Name, Description, Account No, IFSC Code, Bank Name, Branch - Text Box

Sprint2- 3C (Card, Conversation and Confirmation)

CardID - FMS12

Card: As an Office Admin Fee Admin Accountant Parent I must be able to Transfer Fee Amount of students So that Fee Transaction details Students details can be tracked and maintained.

Conversation

1. Implement Bank Reconciliation submenu
   - Create a Bank Reconciliation submenu under the Activity main menu.
   - Rename the menu to Collect Fee DDCheque.
   - Implement the form to select the required fields for fee transfer.
2. Implement Fee Schedule Tab
   - Create a Fee Schedule tab under Bank Reconciliation.
   - Display the necessary fields such as academic year fee category installment fee schedule nameroll number actual concession paid due and to pay.
3. Implement Remittance Tab
   - Create a Remittance tab under Bank Reconciliation.
   - Display the necessary fields such as student amount transaction date time payment mode paid by and status.
4. Implement Deposit Functionality
   - Add a Deposit button in the Bank Reconciliation form.
   - Display a confirmation popup message.
   - Implement the logic to handle the DDCheque realization when the user confirms.
5. Implement Bounced Functionality
   - Add a Bounced button in the Bank Reconciliation form.
   - Display a confirmation popup message.
   - Implement the logic to handle the DDCheque bounce when the user confirms.
6. Implement Realized Functionality
   - Add a Realized button in the Bank Reconciliation form.
   - Display a confirmation popup message.
   - Implement the logic to handle the DDCheque realization when the user confirms.

Confirmation

Given When The user selects '' Bank Reconciliation sub menu under Activity main menu of Fee management module, Then The user must be able to Select the following fields

- Instrument type - Radio button box
- DD Cheque, DD Cheque number, Confirm DD Cheque number, DD Cheque amount, Bank name, Branch name, - Text box

- DD Cheque date, Received Date,Deposited date, Realized bounced date - Calendar dialog box
- Payee - list box
- Settlement Date - Date Field

As a Office admin Fee admin, Given When user clicks '' Fee Schedule '' Tab of Bank reconciliation

Then The following fields need to be Displayed under Fee Schedule tab

- Academic year, Fee category, Installment, Fee schedule, Name Roll number, Actual, Concession, Paid , Due , To pay

As a Office admin Fee admin, Given When user clicks '' Remittance '' tab

Then The following fields need to be Displayed under Remittance tab

- Student, Amount , Transaction date Time , Payment mode , Paid by , Status

As a Office admin Fee admin, Given When user clicks '' Deposit '' button box a pop up message appears for confirmation., Then The user must be able to Click '' Ok ''button and the DD Cheque is Realized

As a Office admin Fee admin, Given When user clicks '' Bounced '' button box a pop up message appears for confirmation, Then The user must be able to Click '' Ok ''button and the DD Cheque is Bounced

As a Office admin Fee admin, Given When a user clicks '' Realized '' button box a pop up message appears for confirmation, Then The user must be able to Click '' Ok ''button and the DD Cheque is Realized.

Sprint3 - 3C and WorkProducts related to the target-projects namely Vision Document, SRS and Glossary.

5.1 Productivity Analysis

Earned Value Analysis(EVA) is used to analyze and monitor the productivity of each sprint. EVA is a systematic approach to tracking project performance by comparing planned scope, schedule, and cost to what is actually being delivered. It involves three key components: planned Value (PV), Actual Cost (AC), and Earned Value (EV). These values help calculate key metrics like Cost Variance, Schedule Variance, Cost Performance Index, and Schedule Performance Index, providing insights on whether a project is on track or not. EVM is valuable for early problem detection, allowing project managers to make necessary adjustments and forecasts.

Earned Value:Sprint1

Table 3: Sprint-1: Earned Value Report

	FMS - 01	**02**	**03**	**04**	**05**	**06**	**07**	**08**	**09**	**10**	**11**	**12**	**Total**
SP	5	15	5	5	10	12	15	10	13	8	12	14	124
Hr / SP	6.2	0.9	1.0	1.2	1.3	1.2	1.5	3.0	1.2	5.0	4.2	5.7	32
PV	31	14	5	6	13	14	22	30	16	40	50	80	321
CSP	2	3	3	3	5	6	10	5	5	4	8	7	61
AV	40	20	10	12	20	23	37	45	26	50	75	90	448
EV	12.4	2.8	3	3.6	6.5	7	14.7	15	6.2	20	33	40	164

SP - Story point of the Userstory

Hr / SP - The number of hours required to complete a Story Point can vary according to each Story Point's weightage. This fluctuation is due to the fact that certain user stories might demand less development time but more testing time, or vice versa.

*PV - Planned Value in Hr (SP * Hr / SP)*

CSP - Completed Story Point

AV - Actual Value in Hr

EV - Earned Value in Hr

Earned Value Analysis is a method used to measure the performance and progress of projects. Key metrics include Cost Variance (CV), Schedule Variance (SV), Cost Performance Index (CPI), and Schedule Performance Index (SPI).

Given the values provided, EVM metrics are calculated as given below::

1. CV = EV-AV= 164 - 448 = -284 A negative CV suggests the project is over budget.
2. SV = EV-PV= 164 - 321 = -157 A negative SV implies the project is behind schedule.
3. CPI = EV/AV = 164 / 448 = 0.366 A CPI of less than 1 indicates that the project's costs are higher than anticipated.
4. SPI = EV/PV = 164 / 321 = 0.51 An SPI of less than 1 suggests the project is progressing slower than planned.

From these calculations, it appears that the project is both over budget and behind schedule. The Cost and Schedule Performance Indices are both less than 1, which suggests the need for closer scrutiny of project management and potential corrective actions.

In the same way, the Earned value of the three Sprints of Project A is calculated and presented in Table 4.

Table 4: Comparative Earned Value Metrics Across Three Sprints of Project A

Prj A EV Analysis	PV	AV	EV	CV	SV	CPI	SPI
Sprint1	321	448	164	-284	-157	0.36	0.51
Sprint2	377	427	229.7	-197.3	-147.3	0.54	0.61
Sprint3	1145	1222	1121.63	-100.37	-23.37	0.92	0.98

Likewise, an analysis of the Earned Value has been conducted for Project B, and the assessment report is presented below in Table 5.

Table 5: Comparative Earned Value Metrics Across Three Sprints executed for Project B

Prj B EV Analysis	PV	AV	EV	CV	SV	CPI	SPI
Sprint1	193	269	98	-171	-95	0.36	0.51
Sprint2	226	256	138	-118	-88	0.54	0.61
Sprint3	687	733	673	-60	-14	0.92	0.98

5.2 Quality Analysis

Indeed, the assessment of software development productivity goes beyond just cost and time factors measured in the Earned Value Analysis (EVA). Quality is a crucial factor that must be considered for a more holistic view of productivity. Therefore, incorporating measures such as the Code Maintainability Index (CMI) could provide an in-depth evaluation of code quality within each Sprint. This index is particularly advantageous as it considers both the Halstead metric and Cyclomatic Complexity, which together provide a more comprehensive view of code quality and maintainability.

The Halstead metric, a component of the Code Maintainability Index (CMI), offers an assessment of a program's size and complexity by evaluating its operators and operands. In contrast, Cyclomatic Complexity measures the number of linearly independent paths through a program's source code, providing an estimation of the program's structural complexity. Together, these metrics provide a robust evaluation of code quality and maintainability.

Formula: CMI = (171 - 5.2 * math.log(V) - 0.23 * CV - 16.2 * math.log(L)) * 100 / 171

The table below displays the Code Maintainability Index (CMI) of Project A - Sprint1

Table 6: Code Maintainability Index (CMI) for Sprint1 of Project-A

Sprint - Quality Analysis	FMS01	02	03	04	05	06	07	08	09	10	11	12
V	48	575	118	29	233	45	49	53	34	33	290	53
CV	14	17	8	2	18	3	3	4	3	3	20	4
L	32	41	44	40	39	34	39	42	27	16	64	42
CMI	54	43	49	55	74	55	48	47	53	58	67	46

1. *Compute the natural logarithms: ln(48) ≈ 3.871, ln(32) ≈ 3.465, then substitute these values into the formula: CMI = (171 - 5.2 * 3.871 - 0.23 * 14 - 16.2 * 3.465) * 100 / 171.*
2. *Simplify the formula by performing the multiplications and subtractions: CMI = (171 – 20.11 - 3.22 - 56.12) * 100 / 171 = 91.55 * 100 / 171.*
3. *Finally, divide 91.55 by 171 to get the final CMI value: CMI ≈ 53.53.*

In the same way, the three Sprints of Project A along with their Code Maintainability Index value is measured and presented in Table 7.

Table 7: Code Maintainability Index (CMI) for Three Sprints of Project-A

Sprint1	**FMS01**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	11	12
	54	43	49	55	74	55	48	47	53	58	67	46
Sprint2	**FMS13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**21**	**22**	23	24
	65.01	54.5	64.5	60.15	71.54	54.68	51.9	73.3	64.15	61.8	72.62	57
Sprint3	**FMS25**	**26**	**27**	**28**	**29**	**30**	**31**	**32**	**33**	**34**	**35**	**36**
	82.86	65.35	70	65.33	81	82.32	71.34	82.02	71	82.02	87.43	85

The following graph illustrates the CMI values for each sprint of Project A,

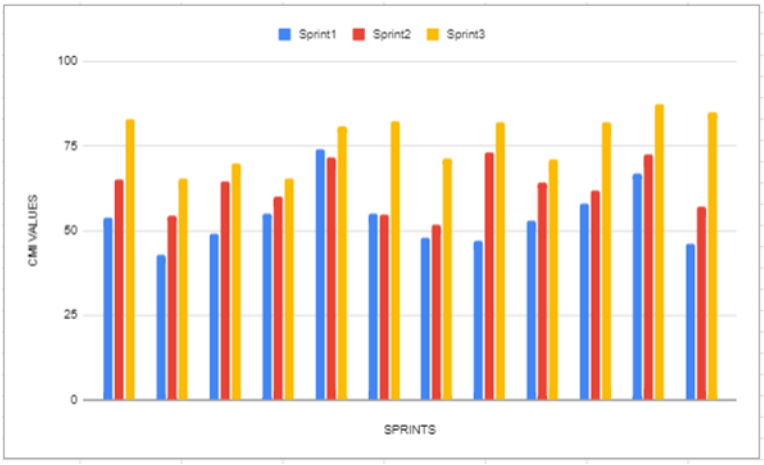

Fig.3. Project A : Sprint vs. CMI

Based on the information provided, it is evident from Figure 4 that Sprint 3 consistently exhibits higher values compared to the other two sprints. This trend indicates that Sprint 3 shows a higher level of performance or achievement in comparison to Sprint 1 and Sprint 2.

Likewise, the measurement of CMI value has been measured for Project B and presented below in Table 8.

Table 8: Code Maintainability Index (CMI) for Three Sprints of Project-B

Sprint1	**REG01**	**2**	**3**	**4**	**5**	**6**	**7**					
	54	43	49	55	74	55	48					
Sprint2	**PRE01**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**		
	65.01	54.5	64.5	60.15	71.54	54.68	51.9	73.3	64.15	61.8		
Sprint3	**PRE10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**SUB01**	**2**	**3**	**4**
	82.86	65.35	70	65.33	81	82.32	71.34	82.02	71	82.02	87.43	85

Since the number of Userstory differs for each sprint the Average values of each sprints are taken and represented in the graphical representation in Fig 4.The Average CMI Value of Sprints are: Sprint 1: 54.33 Sprint 2: 61.47 Sprint 3: 77.36

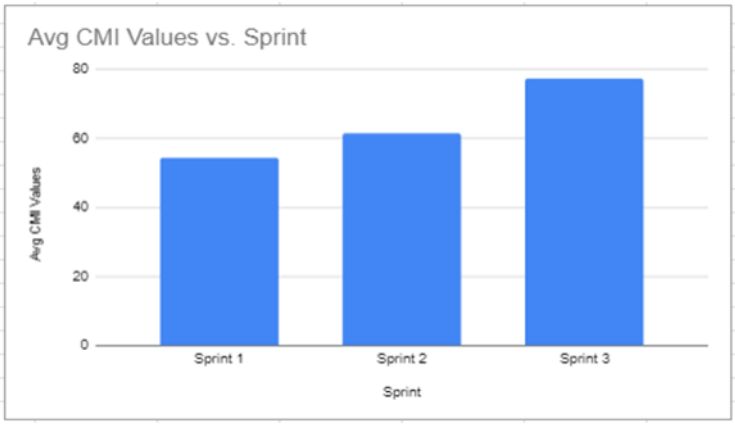

Fig.3. Project B : Sprint vs. Avg. CMI Values

Based on the data, it is evident that the average value of Sprint 3 is consistently higher compared to the other two sprints. This conclusion is supported by Figure 5, which clearly demonstrates the upward trend of the average values for Sprint 3 in comparison to Sprint 1 and Sprint 2.

5. Discussion and Implication

6.1 Productivity

The tables above present comparative Earned Value metrics across three different sprints for Project A and Project B. In both projects, there is a noticeable increase in productivity in Sprint 3, as demonstrated by the highest Cost Performance Index (CPI) and Schedule Performance Index (SPI) in comparison to the earlier sprints. The results, as highlighted in Tables 4 and 5, displayed a notable increase in productivity during the third sprint for both projects, coinciding with the provision of more detailed project information to the developers. This trend underscored a significant correlation between the level of project detail and the productivity output of the development teams, particularly in teams composed of less experienced developers.

6.2 Quality

From the data above, it can be observed that the CMI values for both Project A and Project B increase from Sprint 1 to Sprint 3. which are represented as different levels of information.

In the first sprint of both projects, the CMI values are relatively low. As we move to the second sprint, an increase in the CMI values is noticeable, with the third sprint having the highest CMI values in both projects. The improved CMI values in the third sprints for both projects indicate that as the depth of detail and clarity in user stories provided to the developers increases, the maintainability and thus the quality of the code they produce also increases. Detailed and clear user stories help developers understand the requirements better and thus, they can write more maintainable, efficient, and less complex code, leading to a higher CMI.

Therefore, one could conclude that the depth and clarity of user stories have a direct impact on the code quality. This underlines the importance of clear and detailed user stories for software development projects, as they can significantly improve the maintainability of the code, reduce the chance of misunderstandings, and ultimately lead to higher quality software.

6. Conclusion and Future Work

By mapping the processes of Agile Scrum and OpenUp in the form of an ontology and instantiating them in real-world projects, this research aims to provide a more structured approach to agile development. It not only allows for better productivity and quality measures but also provides a way to manage and improve agile practices through an integrated process ontology. This paves the way for more refined, efficient, and productive software development practices in the future.Through this framework, developers can enhance project management, streamline their workflow, and optimize resource distribution. These insights provide an avenue for software teams to refine their processes, improve project outcomes, and increase customer satisfaction. To facilitate the framework's adoption, teams should be educated about it, and a culture of continuous improvement should be cultivated. Future research directions include conducting longitudinal and comparative studies, leveraging AI and machine learning for the framework's effectiveness, and addressing the challenges of industry-wide adoption. By implementing these recommendations, we can expect enhanced productivity and success in software engineering projects.

References

[1] Dikert, K., Paasivaara, M., & Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. Journal of Systems and Software, 119, 87-108.

[2] Kasauli, R., Knauss, E., Horkoff, J., Liebel, G., & Oliveira Neto, F. G. (2020). Requirements engineering challenges and practices in large-scale agile system development. Journal of Systems and Software, 172, 110851.

[3] VersionOne Inc. (2020). 14th Annual State of Agile Report. Retrieved from https://stateofagile.com/#ufh-i-521251909-14th-annual-state-of-agile-report/473508

[4] Liao, Li, and Hareton KN Leung. "A Software Process Ontology and Its Application." (2014): 207-217.

[5] Oveh, R. O., O. Efevberha-Ogodo, and F. A. Egbokhare. "Software Process Ontology: A case Study of software organizations software process sub domains." *Journal of the Nigerian Society of Physical Sciences* (2019): 122-130.

[6] R. O. Oveh, O. Efevberha-Ogodo and F. A. Egbokhare, "Software Process Ontology: A case study of software organizations software process sub domains," Journal of the Nigerian Society of Physical Sciences, vol. 1, no. 4, pp. 122-130, Nov. 2019, doi: 10.46481/jsps.2019.28.

[7] W. A. Ortega-Ordoñez, C. J. Pardo-Calvache and F. J. Pino-Correa, "OntoAgile: an ontology for agile software development processes," DYNA, vol. 86, no. 209, pp. 79-90, 2019, doi: 10.15446/dyna.v86n209.76670.

[8] T. Singarayan, "Ontology-driven Model for Knowledge-Based Software Engineering," in Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Mysore, India, 2013, pp. 1750-1755, doi: 10.1109/ICACCI.2013.6637460.

[9] G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf and S. Brinkkemper, "The Use and Effectiveness of User Stories in Practice," 2017 IEEE 25th International Requirements Engineering Conference (RE), Lisbon, Portugal, 2017, pp. 174-183, doi: 10.1109/RE.2017.27.

[10] J. Schiel, "The Anatomy of a User Story," Scrum Alliance Resource Library, 2018. [Online]. Available: https://resources.scrumalliance.org/Article/anatomy-user-story.

[11] International Scrum Institute, "Scrum User Stories," 2021. [Online]. Available: https://www.scrum-institute.org/Scrum_User_Stories.php.

[12] Scrum Alliance, "The Anatomy of a User Story," 2018. [Online]. Available: https://resources.scrumalliance.org/Article/anatomy-user-story.

[13] V. Dantas, "Refine User Stories and Acceptance Criteria with Agile," Pluralsight, 2020. [Online].Available: https://www.pluralsight.com/guides/refine-user-stories-and-acceptance-criteria-with-agile.

[14] Agile for Growth, "7 Tips for Writing Acceptance Criteria with Examples," 2017. [Online]. Available: https://agileforgrowth.com/blog/acceptance-criteria-checklist/.

[15] M. Kajko-Mattsson et al., "OpenUP: Basic Concepts and Principles," 2007. [Online]. Available: https://www.eclipse.org/downloads/download.php?file=/technology/epf/OpenUP/published/openup_basic/openup_basic.pdf.

[16] S. Ghani et al., "Integrating OpenUP with Scrum for Agile Software Development," 2018 IEEE International Conference on Innovative Research and Development (ICIRD), Bangkok, Thailand, 2018, pp. 1-6, doi: 10.1109/ICIRD.2018.8376334.

[17] K. Schwaber and J. Sutherland, "The Scrum Guide," 2020. [Online]. Available: https://scrumguides.org/scrum-guide.html.

[18] S. Ghani et al., "Integrating OpenUP with Scrum for Agile Software Development," 2018 IEEE

International Conference on Innovative Research and Development (ICIRD), Bangkok, Thailand, 2018, pp. 1-6, doi: 10.1109/ICIRD.2018.8376334.

[19] Reference.com, “What Is the Formula to Calculate Case-Mix Index?” [Online]. Available: https://www.reference.com/world-view/formula-calculate-case-mix-index-f266b81ea5869e8.

[20] Verifysoft, “Maintainability Index,” [Online]. Available: https://www.verifysoft.com/en_maintainability.html.