Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Оцінювання процесів розробки програмного забезпечення для малих та середніх проєктів з гнучкою організацією

Виконав(ла): студент(ка) 6 курсу, групи СНд-2
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

(підпис) Волович В.Я. (прізвище та ініціали)

Керівник (підпис) Боднарчук І.О. (прізвище та ініціали)

Нормоконтроль (підпис) Мацюк О.В. (прізвище та ініціали)

Завідувач кафедри (підпис) Боднарчук І.О. (прізвище та ініціали)

Рецензент (підпис) Петрик М.Р. (прізвище та ініціали)

Тернопіль
2022

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«21» вересня 2021 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

студенту Волович Володимир Ярославович
(прізвище, ім'я, по батькові)

1. Тема роботи **Оцінювання процесів розробки програмного забезпечення для малих та середніх проєктів з гнучкою організацією**

Керівник роботи **к.т.н., доц. Боднарчук І.О.**
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «*22*» *листопада* 2022 року № *4/7-952*

2. Термін подання студентом завершеної роботи 21 грудня 2022 р.

3. Вихідні дані до роботи Літературні джерела з тематики роботи

4. Зміст роботи (перелік питань, які потрібно розробити)
ВСТУП; 1 ПОСТАНВОВКА ЗАДАЧІ та методика дослідження; 1.1 Оцінка якості літературних джерел для огляду; 1.2 Отримання даних; 1.3 Презентація результатів; 2 AGILE ДЛЯ КОМПАНІЙ МАЛОГО ТА СЕРЕДНЬОГО РОЗМІРУ; 2.1 Моделі зрілості SPI; 2.2 Моделі зрілості SPI для МСП; 2.3 Поєднання гнучких методологій і моделей зрілості SPI; 2.4 Гнучкі методології та моделі зрілості SPI для МСП; 3 КІЛЬКІСНЕ ПРЕДСТАВЛЕННЯ РЕЗУЛЬТАТІВ ОЦІНЮВАННЯ; 3.1 Результати опитування розробників; 3.2 Узагальнення опитування розробників; 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ; 4.1 Синдром професійного вигорання в IT; 4.2 Створення і функціонування системи моніторингу довкілля з метою інтеграції екологічних інформаційних систем, що охоплюють певні території; ВИСНОВКИ; СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ; ДОДАТКИ

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Мацюк О.В., к.т.н., доц.		
Безпека в надзвичайних ситуаціях	Клепчик В.М., ст. викл.		

7. Дата видачі завдання 21 вересня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	22.09.22-27.09.22	*Виконано*
2.	Підбір наукових джерел по темі роботи	28.09.22-04.10.22	*Виконано*
3.	Переклад та опрацювання наукових джерел	05.10.22-11.10.22	*Виконано*
	по темі кваліфікаційної роботи		
4.	Виконання дослідження щодо теми	12.10.22-18.10.22	*Виконано*
	Кваліфікаційної роботи		
5.	Оформлення першого розділу	19.10.22-25.10.22	*Виконано*
6.	Оформлення другого розділу	26.10.22-01.11.22	*Виконано*
7.	Оформлення третього розділу	02.11.22-08.11.22	*Виконано*
8.	Виконання завдання до підрозділу «Охорона праці»	09.11.22-15.11.22	*Виконано*
9.	Виконання завдання до підрозділу «Безпека в	16.11.22-22.11.22	*Виконано*
	надзвичайних ситуаціях»		
10.	Оформлення кваліфікаційної роботи	23.11.22-29.11.22	*Виконано*
11.	Нормоконтроль	30.11.22-09.12.22	*Виконано*
12.	Перевірка на плагіат	10.12.22	*Виконано*
13.	Попередній захист кваліфікаційної роботи	14.12.22	*Виконано*
14.	Захист кваліфікаційної роботи	22.12.2022	

Студент _____ Волович В.Я.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

"Оцінювання процесів розробки програмного забезпечення для малих та середніх проєктів з гнучкою організацією" // Кваліфікаційна робота освітнього рівня «Магістр» // Волович Володимир Ярославович // Тернопільський національний технічний університет ім. І. Пулюя, центр перепідготовки та післядипломної освіти, кафедра комп'ютерних наук, група СНд-2 // Тернопіль, 2022 // с. – 59, рис. – 5, табл. – 1, джерел – 37.

Ключові слова: гнучка розробка Agile; покращення процесів розробки; огляд; систематизація; малі та середні підприємства, МСП

Моделі вдосконалення програмного забезпечення (Software Process Improvement – SPI) були розроблені, щоб допомогти організаціям підвищити якість розробки програмного забезпечення. Гнучкі методології використовуються для забезпечення продуктивності та якості програмного продукту. Серед іншого вони застосовуються на малих і середніх підприємствах (МСП, SMEs – Small and Medium-sized Enterprises). Однак мало відомо про поєднання гнучких методологій і моделей зрілості SPI щодо МСП і результати такого впровадження, оскільки всі поточні моделі SPI стосуються більшим чином великих компаній та організацій. Поєднання цих методологій може призвести до підвищення якості програмних продуктів, покращення методології управління проєктами.

ANNOTATION

“Assessment of Software Development Processes for Small and Medium-Sized Agile-Managed Projects” // Master’s degree qualification paper // Volodymyr Yaroslavovych Volovych // Ternopil Ivan Puluj National Technical University, Center for Retraining and Postgraduate Education, Department of Computer Science, group СНд-61 // Ternopil, 2022 // p. 59, fig. - 9, tables - 1, references. - 37.

Key words: Agile development; Software Process Improvement; review; survey; Small and Medium-sized Enterprises, SMEs

Software Process Improvement (SPI) models were developed to help organizations to improve the quality of software development. Agile methodologies are used to ensure the effectiveness and quality of the software product. Among other things, they are used in small and medium-sized enterprises (SMEs). However, it is the lack of knowledge about the combination of Agile methodologies and SPI maturity models for SMEs and the results of such implementation, as all current SPI models are more relevant to large companies and organizations. The combination of these methodologies can lead to an increase in the quality of software products and an improvement in project management methodology.

ЗМІСТ

ВСТУП

Актуальність задачі.

Малі та середні підприємства (МСП) є найпоширенішим типом підприємств у різних глобальних економіках [1][2] сучасного світу і їх можна назвати основою та «двигуном» промислового зростання [3]. МСП – це підприємства, на яких чисельність працівників менше 50 для малих підприємств і менше 250 для середніх. Роль підприємств малого та середнього розміру та особливості їх управління висвітлені в різних наукових працях, в тому числі в статтях працівників ТНТУ, зокрема професора Ірини Струтинської [1].

Багато з цих МСП зосереджені на розробці програмного забезпечення. Усі ці підприємства більшою чи меншою мірою застосовують методи розробки програмного забезпечення для створення своїх продуктів. Практики програмної інженерії набули великого значення, тому що, якщо вони не застосовуються та не виконуються належним чином, під час розробки програмного забезпечення, можуть виникнути різні проблеми.

Неправильне виконання практик програмної інженерії може, наприклад, призвести до розробки системи, яка містить властивості, які не були затребувані [4]. Основним впливом є переробка, яку необхідно виконати; доведено, що переробка може коштувати до 40% від загальної вартості проекту. Якщо помилки виявляються пізно в процесах розробки програмного забезпечення, вартість може бути в 200 разів більшою, ніж виявлення їх на ранніх стадіях процесу розробки [4].

Крім того, застосування гнучких методологій є кроком вперед у середовищі розробки програмного забезпечення, але все ж існує багато інших аспектів, які не повністю враховуються впровадженням гнучких методологій, таких як забезпечення якості, управління часом тощо.

Мета роботи.

Метою дослідження є визначення основних гнучких методологій і моделей зрілості SPI, які застосовуються в малих і середніх підприємствах, комбінацій

цих методологій і результатів, які можуть виникнути. За допомогою цих комбінацій пропонуються нові рамки розробки програмного забезпечення. Більше того, результати цього дослідження можуть бути використані як керівництво з відповідним поєднанням для кожного SME, як краща методологія управління проектами або як вдосконалення поточної практики розробки програмного забезпечення.

Об'єкт дослідження: Процеси розробки програмного забезпечення зав гнучкими методологіями.

Предмет дослідження: моделі управління проєктами по гнучких методологіях.

Наукова новизна отриманих результатів. Наукова новизна полягає у вирішенні задачі застосування і систематизації гнучких методів управління проєктами для малих та середніх підприємств, а саме:

1. Які гнучкі методології в поєднанні з моделями зрілості вдосконалення програмного забезпечення, створеними для малих і середніх підприємств, існують?

2. У яких ситуаціях і як можна застосувати ці методології?

- Які результати, переваги та недоліки може забезпечити кожна методологія?
- Коли слід застосовувати кожну методологію?
- Які причини невдачі?

3. Чи справді ці методології застосовуються в МСП?

Практичне значення отриманих результатів. Було визначено та обговорено ряд гнучких методологій і різні моделі зрілості SPI. Крім того, представлено комбінацію різних гнучких методологій і моделей зрілості вдосконалення програмного забезпечення, а також їхні переваги та недоліки, які можуть виникнути в малих і середніх компаніях.

Апробація результатів та особистий внесок здобувача. Основні положення роботи доповідались, розглядались та обговорювались на науковій конференції Тернопільського національного технічного університету імені Івана

Пулюя. Результати кваліфікаційної роботи опубліковані у тезах студентської наукової конференції "Інформаційні моделі системи та технології – 2022", яка проводились у ТНТУ.

1 ПОСТАНВОВКА ЗАДАЧІ ТА МЕТОДИКА ДОСЛІДЖЕННЯ

За останні роки було опубліковано немало робіт, які зосереджуються на сфері практик і методів розробки програмного забезпечення, наприклад. роботи [5], [6]. Завдяки цим дослідженням було доведено, що практика програмної інженерії (SE – Software Engineering) здійснюється на малих і середніх підприємствах інакше, ніж у великих компаніях [7]. Наприклад, у малих і середніх підприємствах (МСП) хоча й існують різні ролі, вони не настільки чіткі, і кожен працівник виконує багато різних видів завдань. Таким чином, робота виконується в досить неформальній обстановці, значною мірою на основі співпрацю між учасниками команди.

Щодо моделей покращення для МСП наукових досліджень стосовно практики процесів програмної інженерії на сьогодні небагато. Це пояснюється тим, що МСП володіють унікальними характеристики, які залежать від самої компанії. Малі та середні підприємства стикаються з особливими проблемами через свій розмір і бюджет, в рамках якого вони працюють. Крім того, вони демонструють низький рівень зрілості та менше ресурсів, щоб запровадити процеси забезпечення якості і покращення процесів розробки. Лише деякі МСП документують свої процеси розробки програмного забезпечення.

МСП також не акцентують увагу на навчанні, оскільки у них є стислі терміни, а час, доступний для вдосконалення, обмежений. Натомість МСП дотримуються спрощеного життєвого циклу продукту, де вони приділяють особливу увагу розробці та тестуванню. Навіть більше, вони демонструють відсутність процедур контролю, управління проектами та навичок управління ризиками [8].

Багато малих і середніх підприємств бажають вдосконалювати свої процеси розробки програмного забезпечення, щоб розробляти успішні програмні системи [9]. Однак їм важко реалізувати ці процеси вдосконалення через високі витрати, які потрібні для їх впровадження. Крім того, обмежені ресурси та суворі

часові обмеження, якими працюють МСП, ускладнюють впровадження цих методів.

Більше того, дослідження програмного процесу мотивується тим фактом, що якість процесу пов’язана з якістю програмного продукту. Метою вдосконалення процесу програмного забезпечення є не лише підвищення якості продукту, але й підвищення надійності, послідовності та передбачуваності та скорочення часу виходу на ринок [10], [11].

Незважаючи на те, що кілька методів вдосконалення процесів програмного забезпечення було введено для застосування в малих і середніх підприємствах, таких як CMMI [12], ISO/IEC 15504 [13], PRINCE2 [14] тощо, проблема полягає в тому, що зазвичай усі ці методи вдосконалення підходять лише для великих підприємств. Необхідно ввести зміни в ці методології, щоб вони могли відповідати потребам МСП. Крім того, оскільки умови, в яких можна застосовувати ці методи, недостатньо чітко визначені, переваги, які може надати кожен метод, недостатньо чітко визначені також.

Таким чином, досвідчені практики запропонували ще один підхід до вдосконалення. Цей підхід був названий гнучкою розробкою програмного забезпечення. Цей метод має великий вплив на розробку програмного продукту в усьому світі. Однак, незважаючи на те, що за останні роки було розроблено багато гнучких методологій, мало відомо про те, як ці методи застосовуються на практиці та який вони мають ефект. Наприклад, методології Agile можуть негативно вплинути на основні характеристики проекту, такі як масштаб, час, вартість і якість. Загалом Agile-методології страждають від відсутності дисциплінованого планування [15].

Оскільки методи вдосконалення процесів програмного забезпечення виявилися непридатними для малих і середніх підприємств, дослідження показують, що малі та середні компанії можуть прийняти методології гнучкої розробки, дотримуючись моделей зрілості SPI [16]. Ці методології використовують переваги гнучкості та адаптивності методологій Agile, а також основну цінність методів SPI: контроль. Поєднання моделей зрілості SPI та

гнучких методів було б вигідним, оскільки перші говорять нам, що робити, а другі говорять нам, як це робити. Основна проблема полягає в тому, що немає досліджень, які б аналізували та порівнювали різні можливі моделі та комбінації. З цієї причини малим і середнім підприємствам непросто вирішити, яку з цих моделей вони можуть прийняти.

На дослідницькі питання, сформульовані у вступі до цієї роботи, а саме: «Які гнучкі методики в поєднанні з моделями зрілості вдосконалення програмного забезпечення, розроблені для малих і середніх підприємств, існують?» та «У яких ситуаціях і як можна застосовувати ці методології?», які переваги та недоліки кожної методології?, як їх можна застосовувати та які причини невдачі?» можна дати відповідь шляхом виконання літературного пошуку.

Процес відбувався наступним чином. На початку були визначені відповідні ключові слова. Ключові слова були синонімами термінів, використаних у темі дипломної роботи та досліджуваних питаннях. Далі рядки пошуку були сформульовані на основі визначених ключових слів. Пізніше було проведено пробний пошук у бібліотеках. Якщо результати були недостатніми, питання дослідження або ключові слова були змінені. Використовувались бази даних ACM Digital Library, IEEE Xplore, Science Direct та Inspec.

Для того, щоб звузити кількість результатів, використовувалися фільтри для пошуку відповідних робіт у галузі програмної інженерії або комп’ютерних наук. Крім того, документи, які були зібрані, належать до 2001 року та пізніше, завдяки Agile Manifesto [17]. На фінальному етапі були видалені дублікати. Критерії відбору літературних джерел наведено нижче.

- Статті, опубліковані з 2001 року та пізніше завдяки Agile Manifesto.
- Фільтри в для обмеження результатів пошуку лише у сфері програмної інженерії та комп’ютерних наук.
- До дослідження були включені лише рецензовані статті.
- Зачитано анотацію та ключові слова у великій кількості робіт.
- Кількість цитувань кожної статті враховувалась як допоміжний метод.

• У разі виникнення розбіжностей проводилося вивчення статей, а потім їх обговорення.

1.1 Оцінка якості літературних джерел для огляду

Після того, як відібрані статті для дослідження були зібрані, необхідно було провести оцінку якості на основі конкретних питань, а саме:

Питання, які використовувалися для оцінки якості вибраних досліджень, представлені нижче.

1. Чи є чітке формулювання цілей використаної методології дослідження:
    1) Реальний приклад?
        i. Пошукове дослідження?
        ii. Дослідження дій?
        iii. Описове дослідження?
    2) Порівняльні дослідження?
    3) Експериментальні дослідження?
    4) Опитування?
    5) Інший тип?
2. Чи є адекватний опис контексту, в якому
    1) Реальний приклад?
        i. Пошукове дослідження?
        ii. Дослідження дій?
        iii. Описове дослідження?
    2) Порівняльні дослідження?
    3) Експериментальні дослідження?
    4) Опитування?
    5) Інший тип?
3. Чи відповідав дизайн дослідження цілям
    1) Реальний приклад?
        i. Пошукове дослідження?

ii. Дослідження дій?

iii. Описове дослідження?

2) Порівняльні дослідження?

3) Експериментальні дослідження?

4) Опитування?

5) Інший тип?

4. Чи була контрольна група?

5. Чи належним чином застосована методологія дослідження з точки зору типу дослідження, аналізу даних тощо?

6. Чи дані були зібрані достатньо точним чином, щоб вирішити проблему дослідження?

7. Чи є чітке формулювання результатів?

8. Чи стосуються висновки мети дослідження?

Крім того, запитання оцінювалися наступним чином: Y (Так), P (Частково) і N (Ні). Процедура підрахунку балів була Y=1, P=0,5 і N=0 або невідома (тобто інформація не вказана). Статті, які отримали оцінку нижче 5, були виключені з огляду літератури. Проводилась процедуру підрахунку балів окремо, а коли виникали розбіжності, вони обговорювали питання, поки не досягли згоди.

Більше того, доповіді, які не були оприлюднені на конференціях чи у журналах, частинами книг, були виключені.

1.2 Отримання даних

1.2.1 Ключові слова

Ключові слова, які використовувалися для пошуку, були синонімами теми дослідження та питань дослідження. Зокрема, ключові слова описані нижче:

- гнучкі методології;
- покращення процесу програмного забезпечення;
- МСП (SME);

- моделі зрілості.

1.2.2 Рядки пошуку

Пробні пошуки показали, що перелічені вище ключові слова можуть дати надто широкі результати пошуку. Таким чином, пошукові запити були створені з ідентифікованих ключових слів та їх модифікацій і логічних операторів, таких як «І» та «АБО». Рядки пошуку застосовувалися лише для заголовків і тез.

Спочатку було вирішено пошукати в літературі основні методології Agile та основні моделі зрілості вдосконалення програмного забезпечення. Далі використовувались комбінації з добавленням третього рядка пошуку та ідентифікуючи найпоширеніші моделі, які посилаються на комбінації. Після визначення найпоширеніших методологій виконано перехід до пошуку конкретної літератури, використовуючи останні три рядки пошуку. Зокрема, пошукові запити описані нижче:

- (Метод Agile*) І МСП
- ((Удосконалення процесу SPI АБО програмного забезпечення) І метод*) І МСП
- ((Гнучкий метод*) І ((SPI АБО вдосконалення програмного процесу) І метод*)) І МСП
- (XP АБО Екстремальне програмування АБО Scrum АБО ASD АБО адаптивна розробка програмного забезпечення АБО Lean розробка АБО Методологія розробки динамічних систем АБО Методи Crystal) І МСП
- (CMM АБО CMMI АБО SPICE АБО ISO 15504 АБО P3M3 АБО OPM3 АБО PRINCE2) І МСП
- (XP АБО Екстремальне програмування АБО Scrum АБО ASD АБО адаптивна розробка програмного забезпечення АБО Lean розробка АБО Методологія розробки динамічних систем) І (CMM АБО CMMI АБО SPICE АБО ISO 15504 АБО P3M3 АБО OPM3 АБО PRINCE2) І МСП.

1.2.3 Результати пошуку

Після застосування пошукових запитів кількість знайдених робіт представлена в таблиці 1.1.

Таблиця 1.1 – Кількісний підбір публікацій для дослідження

Джерела	Виявлено	Застосування критеріїв	Відповідні дослідження
IEEE	65	61	36
Science Direct	817	151	42
ACM	271	125	25
Inspec	353	220	60

Через те, що багато відповідних досліджень було знайдено в більш ніж одній базі даних, повторювані роботи були видалені. Загальна кількість досліджень, які були отримані авторами для потреб дисертації, склала 69.

1.3 Презентація результатів

У цьому розділі підсумовуються результати систематичного огляду літератури. Всі публікації були розбиті на наступні три групи: перша – методології Agile, друга – моделі зрілості SPI і третя – комбінація Agile та SPI. Результати дослідження представлені нижче.

Для того, щоб задовольнити потреби нового середовища для промисового створення програмних продуктів, яким характерні зміни вимог, була створена група методологій, які мали б вирішити проблему непередбачуваності, покладаючись на «людей та їхню творчість, а не на процеси» – Agile методології, викладені в маніфесті [17]. Основні ідеї гнучкого підходу наступні:

- «Індивіди та взаємодія щодо процесу та інструментів.
- Робоче програмне забезпечення над повною документацією.
- Співпраця з клієнтом при укладенні контракту.
- Реагування на зміни замість дотримання плану».

Згідно з цими визначеннями ключовими атрибутами Agile організації є: швидкість, гнучкість, навчання та сприйняття змін [3]. Тим не менш, практики погоджуються, що Agile передбачає більше, ніж просто дотримання методології, яка робить проект Agile. Бути гнучким – це більше, ніж набір практик; це спосіб мислення.

За допомогою систематичного огляду літератури було визначено такі основні методології Agile:

- Lean методологія;
- Scrum;
- Екстремальне програмування (XP);
- методологія Crystal;
- модель динамічної розробки системи (DSDM);
- розробка, керована функціями (FDD);
- адаптивна розробка програмного забезпечення (ASD).

У наступному підрозділі аналізуються результати застосування гнучких методологій на МСП.

2 AGILE ДЛЯ КОМПАНІЙ МАЛОГО ТА СЕРЕДНЬОГО РОЗМІРУ

Нове середовище змушує компанії приймати Agile-методології. Це можна чітко побачити в опитуванні 2005 року в США та Європі, яке показало, що 14 відсотків компаній використовували методи Agile, а 49 відсотків компаній, які знають про методи Agile, були зацікавлені в їх застосуванні [18].

Чимало досліджень довели дієвість основних методологій. Коли вони обговорюють Agile загалом, головними зауваженими перевагами є: покращена комунікація і координація команди, більша командна робота та ефективність у швидкому створенні програмного забезпечення, яке відповідає вимогам клієнтів.

Здається, Agile-методології – це підхід до досконалості. Але це не так, деякі практики стверджують, що предметом досліджень в основному є XP і що зростає потреба в більшій різноманітності (Scrum, Lean тощо) та глибших дослідженнях для визначення валідності цих методологій [19].

Зазвичай кажуть, що гнучкі процеси більше підходять для невеликих проектів із низьким ризиком [20], але питання найкращих практик у малих і середніх підприємствах завжди було проблемою. Таким чином, багато практиків стверджують, що впровадження гнучких методологій пов’язане з високою вартістю впровадження нових ідей і все ще багато інструментів і методів застосовуються неправильно [21].

2.1 Моделі зрілості SPI

Покращення процесів розробки програмного забезпечення (SPI – Software Process Improvement) – це «системна процедура для покращення продуктивності існуючої системи шляхом зміни або оновлення існуючих процесів.

На жаль, немає конкретної моделі SPI для малих та середніх компаній, оскільки всі поточні моделі SPI були розроблені для великих фірм, і ці моделі вдосконалення є складними та в багатьох випадках непридатними для використання малими фірмами з розробки програмного забезпечення через те,

що вони є надто складними та дорогими для реалізації. Проте деякі дослідники вказали, що SPI можна використовувати як конкурентну стратегію просування як для малих, так і для великих організацій [4].

Після виконаного аналізу літературних джерел робимо висновок, що найпоширенішими моделями зрілості SPI є наступні:

- CMM;
- CMMI;
- ISO / IEC 15504 (ISO / IEC 33001);
- PRINCE2;
- OPM3;
- P3M3.

У наступному підрозділі обговорюються результати застосування моделей зрілості вдосконалення процесу програмного забезпечення на МСП.

2.2 Моделі зрілості SPI для МСП

Метою кількох моделей зрілості для вдосконалення процесів програмного забезпечення, таких як CMM, CMMI, SPICE тощо, є надання шаблонів якості та інфраструктури управління, які підприємство може запровадити для покращення процесу розробки програмного забезпечення [4].

На жаль, було помічено, що успішне впровадження таких моделей, як правило, неможливе в контексті малих і середніх організацій з розробки програмного забезпечення, оскільки вони не здатні нести витрати на впровадження цих програм удосконалення процесу програмного забезпечення, як було сказано вище. Невеликі компанії зазвичай потребують зовнішньої допомоги в плануванні та впровадженні вдосконалення процесів, щоб бути в курсі найсучасніших досліджень і практики програмної інженерії.

Багато малих і середніх підприємств визнали, що потреба у вдосконаленні та оцінці свого програмного продукту здається недостатньою, оскільки відомо, що якість продукту значною мірою залежить від процесу, який використовується

для його створення. Багато дослідників стверджують, що МСП характеризуються браком ресурсів, відсутністю середовища розвитку та підтримки, відсутністю бюджету та залежністю від великих організацій.

Сьогодні галузь програмного забезпечення є одним із секторів, що найбільш швидко розвиваються, і ця ситуація особливо стимулює постійне створення малих компаній, які відіграють важливу роль в економіці кожної країни.

2.3 Поєднання гнучких методологій і моделей зрілості SPI

Питання впровадження процесів забезпечення якості в продуктах розробки програмного забезпечення є одним з найважливіших заходів. У цьому контексті процес вимірювання та аналізу в моделях і стандартах якості, таких як ISO/IEC 15504 [22] та ISO/IEC 33001 [23], широко прийнятий індустрією програмного забезпечення для надання своїх програмних продуктів і послуг.

Однак для малих та середніх підприємствах труднощі в їх прийнятті цих стандартів зростають через широкий спектр цих моделей. Як наслідок, МСП мають більший інтерес у прийнятті гнучких методологій, які гарантують їм доставку програмного забезпечення відповідно до можливостей.

Деякі можливі переваги поєднання методологій Agile та моделей зрілості SPI, які можуть виникнути для МСП, полягають у покращенні якості програмного продукту, більш ефективних методах управління проектами, чіткому процесі розробки програмного забезпечення та скороченні вартості розробки вартість [24].

Хоча були проведені дослідження щодо сумісного застосування гнучких методологій і моделей зрілості SPI, але такі питання потребують все ще подальших досліджень.

Таким чином, постійне вдосконалення процесів організації розробки програмного забезпечення є важливим для підвищення можливостей організації. Традиційні підходи до організаційних SPI, однак, необхідно

змінити, щоб уможливити співіснування гнучких проектів та організаційних заходів SPI. Шляхом аналізу літературних джерел в роботі наведено такі найпоширеніші комбінації моделей зрілості SPI та гнучких методологій:

- CMMI і XP;
- CMM і XP;
- PRINCE2 і XP;
- CMMI та Scrum;
- CMM і Scrum;
- CMMI та Lean;
- CMMI і шість сигм;
- PRINCE2 і DSDM.

У наступному підрозділі аналізуються наведені вище комбінації, а також обговорюється їх застосування на МСП. Зокрема, виконано аналіз можливості застосування цих моделей для МСП з обговоренням їхніх переваг і недоліків.

2.3.1 CMMI і XP

Головною причиною об’єднання цих двох методологій є можливості, які надає XP для вивчення, застосування та адаптації. Крім того, XP вже відповідає рівню 2 CMMI [4]. Для об’єднання методологій було використано методологію поєднання ключових процесів (Key Process Activities – KPA) CMMI у поєднанні з кожним принципом XP, щоб побачити, чи можна їх комбінувати. Нижче наведено кроки, які необхідно виконати, щоб застосувати цю комбінацію.

- Визначити цільовий рівень CMMI – якого рівня хоче досягнути підприємство.
- Проаналізувати різні проблеми при поєднанні методологій.
- Перевірити, який фактичний стан компанії, як для CMMI, так і для XP.
- Нарешті, застосуйте комбінацію, внісши необхідні зміни та розширення методології XP, щоб повністю досягти бажаного рівня CMMI в цілому.

Поєднання двох моделей надає розробникам широкий спектр інструментів і опцій [13], крім того, було показано, що семінари та спостереження є продуктивними способами збору даних оцінки з Agile-проєктів [25]. Нарешті, завдяки додаванню CMMI до XP було значно зменшено ризики невдачі проєктів.

2.3.2 CMM і XP

Дослідження показали, що CMM може бути застосований у малих і середніх підприємствах і знижує вартість, час розробки, якість і задоволеність клієнтів [26]. XP є клієнтоцентричним, а, отже, надається перевага командній роботі та децентралізованому підходу, де всі зацікавлені сторони мають рівний ранг. Команда зосереджується на проблемах і рухається до вирішення із взаєморозумінням та ефективним способом.

Можна легко помітити, що функції Agile вбудовані в CMM. Дослідження показали, що поєднання цих двох методологій може зменшити витрати на навчання та документацію, які потрібні компаніям. Таким чином МСП могли б зберегти цінні ресурси.

У XP основними етапами є планування, управління, проектування, кодування та тестування. CMM складається з п’яти різних рівнів – початкового, повторюваного, визначального, керованого та оптимізуючого. Однак деякі ключові дії процесу є спільними – функції Agile вбудовані в CMM – для цих двох моделей, а саме запобігання дефектам, орієнтація на організаційний процес, розробка програмного продукту, міжгрупова координація, планування програмного проекту, забезпечення якості програмного забезпечення та керування конфігурацією програмного забезпечення.

Зокрема, компанія чи навіть команда, щоб застосувати цю комбінацію, має застосувати функції XP на різних рівнях CMMI. На рівні 5 CMMI – оптимізація підприємство має застосувати постійну інтеграцію разом із запобіганням дефектам. На рівні 3 CMMI – функції XP, які мають бути застосовані, – це зосередженість на команді разом із зосередженістю на організаційному процесі, простий дизайн, стандарт кодування та відпочинок,

які мають застосовуватися разом із ключовим процесом розробки програмного продукту.

Крім того, парне програмування має застосовуватися разом із міжгруповою координацією CMMI. Нарешті, на рівні 2 CMMI – Repeatable, властивості XP зменшена версія та парне програмування варто застосовувати разом із KPA CMMI, а саме плануванням проєкту та забезпеченням якості програмного продуту відповідно.

Більше того, хороші сторони цих двох методологій, які можуть використовуватися МСП, полягають у наступному:

• Зменшення вартості розробки та часу розробки, задоволеність клієнтів, підвищення якості програмного продукту та зменшення бюрократії.

• Деякі інші переваги, а саме створення інноваційних та високопродуктивних продуктів.

• Краща взаємодія під час командної роботи та програмування в парах. Крім того, CMM стверджує, що є гнучкою моделлю, яку можна адаптувати до багатьох моделей життєвих циклів [27].

З іншого боку, у цій комбінації також були помічені недоліки.

• CMM здається дорогим, оскільки вимагає навчання персоналу.

• Це збільшує обсяг документацію, що суперечить принципам Agile.

• Крім того, XP не підходить для великих проєктів і більше орієнтований на код, а не на проєкт.

Незважаючи на недоліки та інші можливі проблеми, адаптація цих практик зменшить вартість навчання, а на ранніх стадіях розробки програмного забезпечення вимоги до документації мінімальні. Таким чином малі та середні підприємства заощадять більше, ніж витратять.

2.3.3 PRINCE2 і XP

PRINCE2 – це дуже популярний метод управління, відомий своєю жорсткістю, суворістю та орієнтованістю на документи. PRINCE2 був дуже корисним методом планування та відстеження програмного процесу. З іншого

боку, XP як метод Agile страждає від відсутності строгого планування. Основна мета цієї комбінації – отримати найкраще з XP і PRINCE2. Це призведе до нового методу, який використовуватиме основні принципи XP – гнучкість та адаптивність, а також основну цінність PRINCE2 – контроль.

Кроки, які має виконати підприємство, щоб застосувати цю комбінацію, описані нижче. Ця комбінація робить наголос на простому дизайні, оскільки це полегшує планування часу, бюджету і сприяє спілкуванню між командою розробників і клієнтами. Крім того, компанія також має застосувати фазу аналізу, оскільки вона отримає вигоду від створення більш реалістичних і точних проєктів. Більше того, підприємство має застосовувати функції XP, такі як постійна інтеграція та тестування, щоб зменшити ймовірність ризику недотримання нефункціональних вимог.

Також PRINCE2 вимагає строгих формальних процедур контролю для тестування та інтеграції. Таким чином, необхідно, щоб усі процедури були належним чином задокументовані. Підприємствам слід звернути особливу увагу на забезпечення та управління проєктом. Ці дві функції мають бути спрямовані на планування якості на початку проєкту, а потім на верифікацію створеної частини продукту.

Нарешті, ще одне важливе питання – стандартизація. Керівники проекту повинні зосередитися на стандартній структурі кодування та правилах іменування ідентифікаторів, оскільки в протилежному випадку дуже важко контролювати якість.

Переваги, які можуть виникнути в результаті цієї комбінації:

- PRINCE2 значною мірою залежить від паперової роботи – майже кожен крок PRINCE2 документується та реєструється. Навпаки, методології Agile і особливо XP покладаються на усні повідомлення, а не на документи.
- Більше того, в XP немає ієрархії, тоді як PRINCE2 залежить від 4-рівневої ієрархічної структури в команді.
- Вважається, що PRINCE2 є менш гнучким за будь-який гнучкий метод розробки через те, що він вимагає контролю для кожного кроку життєвого циклу

проекту з розробки програмного забезпечення. Таким чином, XP вимагає максимально можливої гнучкості в прийнятті змін вимог. Тим не менш, ця комбінація має деякі недоліки.

- PRINCE2 вимагає важкої бюрократії.
- PRINCE2 є методом управління проектами та підтримує якість продукту; з іншого боку, XP не вимірює та не планує аспект якості розробки програмного забезпечення. Щоб подолати це, потрібне навчання команд розробників.

Незважаючи на недоліки, гнучкість і дисциплінованість не виключають одне одного. Швидше, вони можуть доповнювати один одного. Гнучкість може сприяти розвитку креативності та покращити відносини з клієнтами, а дисциплінованість дозволить утримувати проект на правильному шляху та в рамках обмежень бюджету, часу та якості. Таким чином, немає неправильної відповіді щодо того, який метод керування використовувати в XP.

2.3.4 CMMI та Scrum

За допомогою огляду літератури було виявлено, що модель зрілості CMMI та гнучка методологія Scrum можуть бути сумісні. CMMI зосереджується на високому рівні абстракції, наприклад, на тому, що роблять проекти (функціонал), а не на тому, яка методологія розробки була використана. Натомість Scrum зосереджується на тому, як проекти власне реалізують розробку програмних продуктів.

Тому Scrum і CMMI можуть співіснувати і разом забезпечують більш потужну комбінацію адаптивності та передбачуваності, ніж кожен окремо. Крім того, Scrum надає розробку програмного забезпечення методологію «як робити», – саме те, чого немає в CMMI. CMMI надає методи системної інженерії, які допомагають Scrum у великих проектах. Крім того, CMMI забезпечує також управління процесами та підтримує практики, які можуть допомогти розгортати, підтримувати та постійно вдосконалювати розгортання Scrum у малих і середніх підприємствах [28].

Відображення практик Scrum у областях процесу CMMI було основним способом поєднання цих двох методологій. Щоб зробити це відображення ефективним, було зроблено кілька адаптацій до практик Scrum. Ці зміни в основному стосуються гнучкого управління ризиками, управління проблемами та методами оцінки [28]. Основні ідеї щодо застосування цієї комбінації описані нижче.

• CMMI зосереджується на компанії та її роботі. Це більш вигідно, коли потрібно, щоб усі процеси розвитку були спрямовані на вдосконалення. З іншого боку, Scrum не належить до рівня організації.

• Scrum не може охопити всі сфери процесу управління проектами. Однак його можна адаптувати для відповідності вимогам CMMI. Крім того, керовані планом процеси CMMI можна було б покращити за допомогою методів Scrum.

• Що стосується управління ризиками, Scrum може ідентифікувати потенційні ризики, але він не має жодної практики для визначення джерела, параметрів, а також для аналізу та контролю ресурсів для управління ризиками. Таким чином, це не передбачає жодних стратегій для зменшення ризиків. З іншого боку, CMMI пропонує різні стратегії для протистояння потенційним ризикам. Тому особливу увагу слід приділити CMMI для управління ризиками.

• Нарешті, Scrum не має жодних спеціальних практик для вирішення питань управління конфігурацією та забезпечення якості. Однак CMMI надає механізми для їх підтримки. Зокрема, він встановлює та підтримує цілісність роботи, використовуючи контроль версій, ідентифікацію та статус. Крім того, щоб забезпечити якість продукту, CMMI надає спеціальні практики для оцінки розробки програмного забезпечення, продуктів і послуг.

Переваги цієї комбінації:

• Основна перевага полягає в тому, що найкращі речі двох методологій адаптовані до одне до одного з врахуванням таких вдосконалень, як краща оцінка ризиків, проблем і оцінки завдяки принципам CMMI [28].

• Приймаючи CMMI, комбінація також отримує краще задокументовані вимоги, що призводить до покращення якості продукту.

• З іншого боку, застосовуючи Scrum, комунікація всередині команди та з клієнтом покращується, що відображається на підвищенні продуктивності команди.

Позитивним результатом є також те, що ця методика не має особливих недоліків. Як це трапляється в усіх комбінаціях, основними недоліками є час, який потрібно інвестувати, щоб мати можливість отримати найкращі речі від методології.

Підсумовуючи, проекти, які поєднують Agile-методології з CMMI, є більш успішними у створенні якіснішого програмного забезпечення, яке задовольняє потреби клієнтів швидше.

2.3.5 CMM і Scrum

Метою CMM є забезпечення якості проекту програмного забезпечення. Крім того, це мінімізує проєктні ризики, оскільки відповідає за відстеження процесу розробки проекту, а розробка повинна дотримуватися планів. Натомість Scrum несе відповідальність за взяття сміливих зобов’язань і надання пріоритету хорошій комунікації в компанії і з клієнтами.

Найважливішим критерієм вибору для цієї комбінації є готовність консультантів інтерпретувати вимоги CMM з точки зору Agile. Свобода в цьому тлумаченні існує, оскільки ця комбінація стосується того, що мають робити підприємства, а не того, як вони мають це робити. Завдяки такому поєднанню забезпечується якість проекту, переваги комунікації та підтримка сталого темпу розвитку, який можуть забезпечити методи Agile.

Більше того, для підприємства доцільно прийняти цю комбінацію, спочатку застосувати методологію Scrum для швидкого вирішення різних проблем. Потім вони могли застосувати CMM для порівняння та вимірювання процесу розробки.

Переваги, які можуть виникнути в результаті поєднання CMM і Scrum, полягають у наступному [29]:

• Спеціалісти з оцінювання можуть визначити можливі сильні та слабкі сторони організації.

• Групи оцінювачів можуть визначити ризики, такі як терміни, якість продукту, контракти тощо, і запропонувати рішення для їх мінімізації.

• Менеджери та персонал можуть розуміти необхідні дії для планування та впровадження вдосконалення процесу програмного забезпечення для своєї організації.

• Зниження витрат і точність планування ресурсів у процесі розробки програмного забезпечення.

• Підвищення якості продукції та продуктивності.

Хоча переваги цієї комбінації були проаналізовані раніше, однак є деякі недоліки, які слід взяти до уваги:

• CMM не завжди вказує конкретний спосіб досягнення цілей покращення лише тому, що якщо одна організація дотримується правил, встановлених CMM, це не гарантує, що вона буде успішною, оскільки існують інші чинники.

• CMM говорить, що вам потрібно, а не як це зробити. Крім того, CMM звертається до процесів, а не до людей.

• За такої комбінації керівнику проекту важко структурувати, організувати та спланувати проект, якому бракує чіткого визначення.

• Часті зміни, часте постачання продукту та невизначеність щодо точного характеру готового продукту призводять до досить інтенсивного життєвого циклу проекту для всіх учасників.

• Потрібне навчання, щоб члени підприємства могли навчитися використовувати CMM і Scrum. У протилежному випадку, якщо учасники не будуть добре розуміти особливості конкретного проєкту, він може навіть провалитися.

Незважаючи на те, що недоліків у такої комбінації достатньо, переваг, які можуть виникнути для компанії, більше. Перш за все, цю комбінацію можна використовувати в організаціях малого та середнього розміру з дуже невеликими коригуваннями, такими як створення менших артефактів і спрощення процесів, які вимагає CMM. Крім того, помічено покращення якості програмного продукту та зниження вартості розробки. Нарешті, було виконано належне використання часового ресурсу.

2.3.6 CMMI та Lean

Обидва підходи – CMMI та Lean – мотивують мислення в «досконалих ощадливих процесах» і дозволяють використовувати спільну термінологію. Завдяки поєднанню цих двох можна отримати такі результати [30]:

- Lean дозволяє побачити «відходи», тобто «зайві» витрати.
- CMMI має вбудовані механізми, щоб уникнути «відходів».
- CMMI формує розширений набір інструментів для впровадження ощадливого мислення в середовищах розробки / обслуговування / придбання.
- CMMI надає чітку дорожню карту для орієнтації процесу як на рівні проекту, так і на рівні організації.
- Lean підтримується CMMI через концепцію інституціоналізації та організаційного навчання.
- Обидві методології вимагають відданості та проактивного лідерства на всіх управлінських ієрархіях.

Інформації про таку комбінацію методологій є небагато, а у відкритих джерелах немає взагалі

2.3.7 CMMI і шість сигм (Six Sigma)

CMMI використовується для створення інфраструктури організаційних процесів шляхом звернення до певних доменів, таких як програмне забезпечення та системна інженерія [31]. Six Sigma – це ініціатива «зверху вниз», яка охоплює все підприємство, включаючи такі сфери, як інженерія, продажі, маркетинг і

дослідження. Шість сигм призначено для реалізації з акцентом на проблемах і можливостях, часто з вузькими рамками, які дадуть значні переваги для бізнесу.

Він зосереджується на продуктивності процесів і практик, як вони реалізовані, а не на перевірці на відповідність визначенню чи моделі. Хоча ці дві ініціативи вдосконалення відрізняються за дизайном, вони взаємозалежні у своєму використанні. Наприклад, Six Sigma можна використовувати для виявлення потреб процесів у більшій повторюваності, CMMI можна використовувати для запровадження процесів на основі передового досвіду спільноти, а потім Six Sigma можна використовувати для оптимізації цих процесів.

CMMI пропонує функції інституціоналізації, яких не вистачає в Six Sigma. Six Sigma посилює фокус на місії, а її стратегія розгортання на підприємстві сприяє зміні культури, яка підтримує впровадження CMMI. Існує чотири різні стратегії, які можна використовувати для поєднання CMMI і Six Sigma. Ці стратегії описано нижче:

- Варто впроваджувати процеси CMMI, як процеси Six Sigma.

За допомогою цієї стратегії мета команди розробників Six Sigma полягає в тому, щоб реалізувати процес або групу процесів. Їх завдання полягає в тому, щоб визначити проблему або можливість і використовувати наявні дані для інформування про вдосконалення або проектування процесів, які будуть служити місії компанії та відповідати вимогам моделі.

- Використання Six Sigma в якості рушія для високої продуктивноситі команди та високого рівня зрілості компанії.

Тактика Six Sigma може бути використана для безпосереднього збагачення визначених процесів, які варто покращити з точки зору CMMI.

- Варто застосовувати Six Sigma, щоб покращити або оптимізувати стратегію вдосконалення та процеси організації.

Шість сигм можна використовувати для прийняття рішень щодо впровадження ініціатив щодо покращення, а також для управління накладними витратами, пов’язаними із впровадженням. Крім того, використовуючи CMMI

для керівництва та, можливо, як управління для конкретних удосконалень, організація могла б застосувати Six Sigma для кожного удосконалення та просуватися до «контролю» та «оптимізації» одного проекту за раз.

Це підхід до встановлення стратегії організації. Він більш довгостроковий і перспективний та підтримує ідею, що організація повинна взяти під контроль свою долю та керувати своїми ініціативами, а не бути під їх управлінням. Зокрема, незалежно від ярлика, ідея залишається незмінною: організація встановлює набір стандартних процесів, який включає всі особливості вибраних ініціатив. Ця ідея передбачає, що свідомі рішення повинні бути прийняті на організаційному рівні для прийняття цих ініціатив. Також передбачається, що процеси адаптуються з часом і стійкі до реалій організації.

Що стосується переваг, то вони описані в наступних пунктах.

• CMMI та Six Sigma разом створюють міцну основу для покращення продуктивності.

• Зосередженість Six Sigma може допомогти зменшити ризики, пов’язані з прагненням до вдосконалень. Крім того, він також надає основи вдосконалення та аналітичні методи, які дозволяють досягти цілей CMMI.

• Six Sigma надає організації миттєвий знімок поточної ефективності підприємства, який можна використовувати як дорожню карту для передбачення майбутньої продуктивності та вдосконалення.

• З іншого боку, інфраструктура процесів CMMI є основою для процесів Six Sigma. Крім того, це допомагає інженерним процесам підприємства пов’язувати його з бізнес-процесами.

Тим не менш, є деякі недоліки, які слід враховувати.

• Обидві ці моделі демонструють тенденцію до мінімуму – уникайте важких речей.

• Вони розроблені для більших організацій і мають бути внесені модифікації, щоб МСП могли їх використовувати.

Визначення того, що є відповідним, вимагає розуміння вибраних ініціатив та їхніх відмінностей, синергії та зв'язків. CMMI і Six Sigma не можуть поєднувати одна одну, оскільки це різні типи моделей. Їх спільне розгортання є синергетичним. Потенційна цінність, яка додається, полягає в прискореному досягненні цілей ефективності, прискореному досягненні впровадження CMMI, сильніших базових навичках моніторингу та аналізу для забезпечення кращої кількісної оцінки результатів, а також усі відповідні зміни культури, які супроводжуються цими вдосконаленнями.

2.3.8 PRINCE2 і DSDM

Поєднання DSDM з PRINCE2 виглядає так, що ці дві методології доповнюють один одного. PRINCE2 може забезпечити контроль, а DSDM – гнучкість. Цю комбінацію можна знайти в таких концепціях, як планування на основі продукту, залучене партнерство користувачів і розробників і сильний акцент на базовому бізнес-обґрунтуванні [32]. Поєднання PRINCE2 і DSDM здається безпечним підходом, оскільки ці дві моделі мають багато спільного.

DSDM із робочою групою PRINCE2 розглядають дві методології як взаємодоповнюючі. PRINCE2 може забезпечити контроль, а DSDM – гнучкість. Крім того, у статті [33] стверджується, що розробники DSDM мали на увазі PRINCE2. Це можна знайти в таких концепціях, як планування на основі продукту, залучене партнерство користувачів і розробників і сильний акцент на базовому бізнес-обґрунтуванні.

PRINCE2 – це метод управління проектами, призначений для всіх типів проектів, тоді як DSDM – метод швидкої розробки додатків. Для організації, яка використовує PRINCE2 для IT, забезпечує спільність між DSDM та іншими типами проектів. Підприємство, щоб застосувати цю комбінацію, має виконати ряд кроків:

• На початковому етапі розробки проекту ранні етапи PRINCE2 перекривають етапи DSDM. Крім того, обидві методології мають головну

контрольну точку в кінці ініціації. У цьому пункті необхідно підтвердити рішення про продовження та розглянути можливість відмови від проекту.

• На поточній фазі розробки PRINCE2 не потребує жодних етапів управління, щоб відповідати технічним вимрнгам. Крім того, етап управління може складатися з кількох часових блоків DSDM. Кількість необхідних етапів має визначатися співвідношенням необхідного управлінського контролю до потенційних накладних витрат.

• Наприкінці проекту закриття PRINCE2 збігається з фазою впровадження DSDM. Крім того, перевірка проекту виконується на кожному кроці DSDM і пов'язана з оцінкою кінцевої стадії PRINCE2, яка є останньою процедурою в проекті.

Завдяки поєднанню PRINCE2 і DSDM можна отримати деякі з наступних переваг:

• Хороша комунікація між командою проекту та іншими зацікавленими сторонами.

• Механізми роботи з відхиленнями від плану проекту.

• Гнучкі точки прийняття рішень.

• Пріоритизація чітко визначена та виконується на початку проекту.

• Швидка видимість процесу розробки та вирішення потенційних проблем.

• Користування обома моделями – PRINCE2 і DSDM безкоштовне.

• Постачання проектів програмного забезпечення за вартістю та вчасно.

• Доставка бізнес-продуктів протягом проекту, а не тільки в кінці.

• Вітає зміну вимог, навіть на пізній стадії проекту, використовуючи пріоритети та розподіл часу, щоб контролювати це в межах часу та бюджету, щоб використовувати зміни для конкурентної переваги клієнта.

• Невеликі команди з уповноваженими представниками користувачів як повноцінними та постійними членами команди.

• Спрощені семінари та особисте спілкування, мінімізація документації, де це можливо.

Хоча переваги, які можуть виникнути завдяки такій комбінації, є достатніми, проте існують і недоліки:

- Мало інформації про цю комбінацію.
- Потрібна бюрократія, хоча DSDM як гнучка техніка використовує семінари та спілкування віч-на-віч.

У світі, де швидкість доставки часто важливіша, ніж наявність 100% функціональності, і де проекти повинні виконуватися в межах часу та бюджетних обмежень, щоб скористатися ринковими можливостями або відповідати складним вимогам, DSDM працює найкраще.

Крім того, у середовищі, де багато організацій змушені демонструвати, що вони ефективно контролюють свої проекти та дають найкраще співвідношення ціни та якості, працює PRINCE2.

Оскільки обидві ці потреби часто виконуються одночасно, використання PRINCE2 для його керування та DSDM для його гнучкості є потужною комбінацією. Більш того, їх поєднання дає результат, де ціле більше, ніж сума складових частин.

2.4 Гнучкі методології та моделі зрілості SPI для МСП

У попередніх розділах були проаналізовані різні способи поєднання гнучких методологій і моделей зрілості SPI. Постійне покращення процесів програмного забезпечення є важливим для підвищення можливостей організації. Умову зміни можна належним чином описати наступним рівнянням:

$$N * F * S > C \qquad (2.1)$$

де N – незадоволення поточним станом речей;

F – бачення майбутнього стану;

S – здійснення перших кроків для досягнення бачення;

C – можливість змін в компанії.

Якщо в компанії виконані умови з цього рівняння, то вони можуть покращити свою практику розробки програмного забезпечення. Однак традиційні підходи до SPI необхідно змінити, щоб можна було забезпечити співіснування методологій Agile та SPI.

Більше того, МСП страждають від відсутності досліджень для вирішення проблеми вдосконалення процесів розробки програмного забезпечення. Таким чином виявляється, що наразі організації, особливо МСП, все частіше використовують Agile-методології – найпоширенішими методологіями є XP, Scrum і DSDM – і моделі зрілості SPI – найпоширеніші моделі зрілості. є CMM, CMMI та PRINCE2 – у своїх програмних проектах. Однак широкомасштабне та систематичне впровадження методологій Agile та моделей зрілості SPI досить складне, оскільки вони в основному зосереджені на діяльності на рівні проекту.

У результаті проведеного дослідження виявилось, що найпоширенішими методологіями Agile є Scrum, XP і DSDM. Крім того, найпоширенішими моделями зрілості SPI є CMM, CMMI, ISO/IEC 15504 і PRINCE2. Нарешті, найпоширенішими комбінаціями методологій Agile та моделей зрілості SPI є CMMI та XP, CMMI та Six Sigma, CMMI та Scrum, PRINCE2 та DSDM.

Більше того, ці комбінації можуть отримати різні переваги. Усі вони передбачають підвищення якості, мінімізацію ризиків, розробку за вартістю та часом, кращу комунікацію між менеджерами, командою розробників та клієнтами, а також чітке панування у розробці програмного забезпечення. Крім того, ці моделі є фреймворками, і вони містять інструкції для управління та розвитку компанії. Крім того, вони забезпечують механізми, такі як невеликі артефакти, документація, процеси контролю та інші, щоб уникнути можливих відхилень від планів проекту.

Підводячи підсумок, результати показують, що стандарти SPI можуть підходити до методів Agile. Краще буде, якщо компанії зможуть прийняти обидва фреймворки, оскільки вони приносять користь один одному (наприклад, шляхом встановлення кращої комунікації в компанії, зменшення бюрократії

шляхом уникнення великих етапів проектів і створення менших за розміром артефактів. Крім того, хоча великі організації та малі та середні компанії значно відрізняються, проте комбінації з невеликими коригуваннями, такими як зменшення документації та створення менших артефактів, можуть використовуватися в МСП.

3 КІЛЬКІСНЕ ПРЕДСТАВЛЕННЯ РЕЗУЛЬТАТІВ ОЦІНЮВАННЯ

Цей розділ побудований на основі даних, наведених у роботі [34] та має на меті відповісти на сформульовані у вступі задачі дослідження: «У яких ситуаціях і як можна застосувати ці методології?», «Які переваги та недоліки може надати кожна методологія?» та «Чи справді ці методології застосовуються в малих та середніх підприємствах?».

3.1 Результати опитування розробників

Цільовою аудиторією опитування були практики з розробки програмного забезпечення, які працювали із методологіями Agile та моделями зрілості вдосконалення програмного забезпечення. Конкретні особи для опитування відбирались на основі особистих контактів та через Інтернет (наприклад, LinkedIn), щоб знайти відповідний зразок, включаючи досвідчених практиків у галузі програмної інженерії.

Більшість учасників опитування працювали на малих і середніх підприємствах, і більше половини з них працюють на підприємствах, які існують більше 10 років.

Далі наведемо поставлені запитання та відповіді на них цільової авдиторії.

Перше опитування запитання стосується найпоширеніших методологій Agile, використовуваних в командах малих та середніх розмірів.

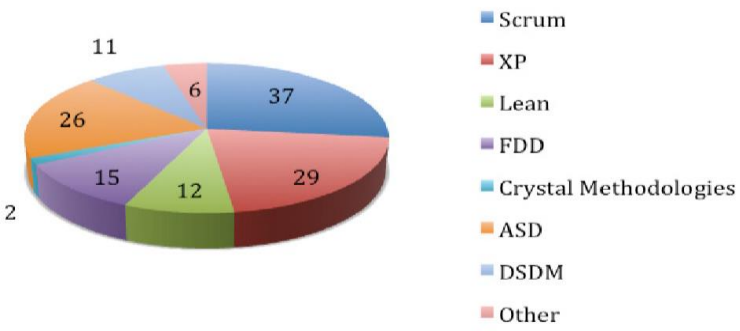

Рисунок 3.1 – Кількісне представлення гнучких методологій розробки, з котрими практично знайомі самі розробники

Як показано на рисунку 3.1, найпоширенішими гнучкими методологіями є Scrum, XP і адаптивна розробка програмного забезпечення.

Наступне запитання стосувалося тривалості періоду, протягом якого учасники використовували методології Agile.

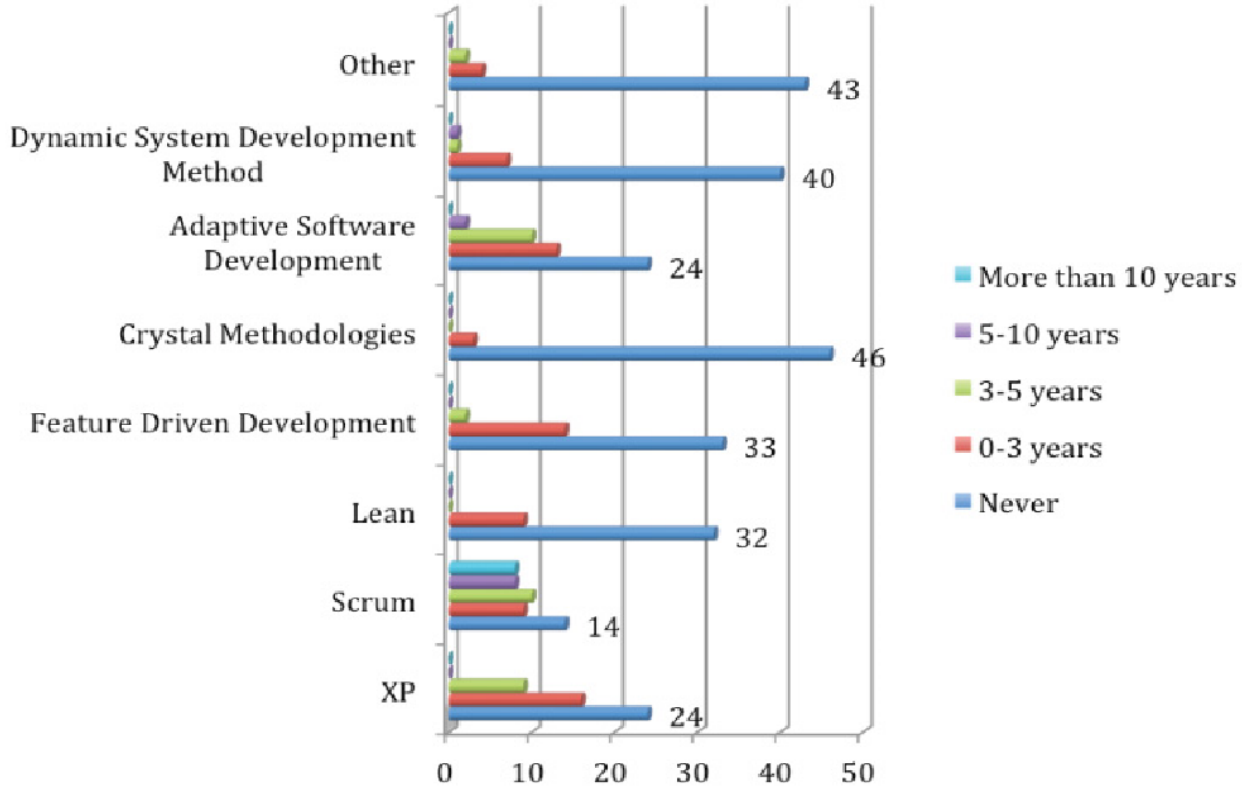

Рисунок 3.2 – Час застосування найпоширеніших методологій Agile

Як видно на рисунку вище, більшість учасників ніколи не використовували методології Crystal, метод розробки динамічних систем і розробку, керовану функціями. Натомість вони використовували XP більше 3 років, Scrum більше 5 років і Adaptive Software Development (ADS) також більше 5 років. Можна легко помітити, що найпоширенішою методологією Agile для учасників є модель Scrum.

Наступне запитання стосується результатів застосування методології Agile.

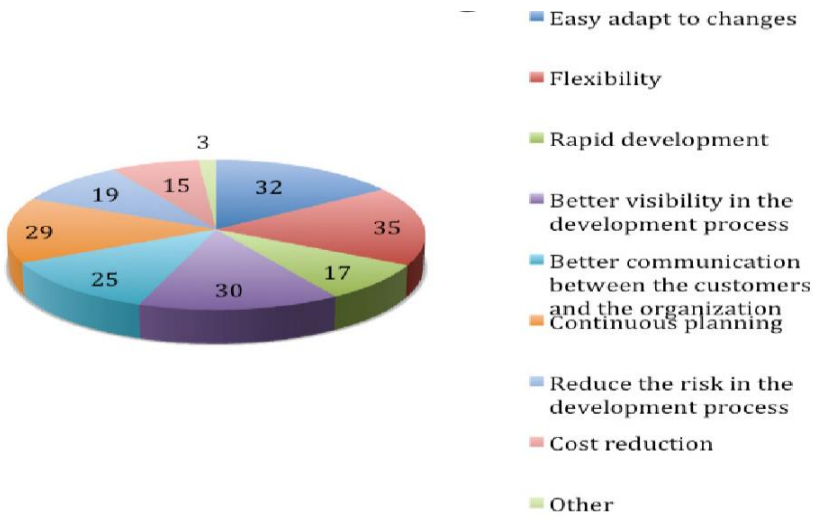

Рисунок 3.3 – Результати застосування методології Agile

На рисунку 3.3 проаналізовано результати застосування найбільш гнучких методологій. Вони забезпечують гнучкість і кращу видимість під час процесу розробки програмного забезпечення, вони легко вносять зміни, і, нарешті, вони покращують комунікацію між клієнтами або іншими зацікавленими сторонами та підприємством.

У наступному запитанні до фахівців-практиків запитували про те, наскільки вони знайомі з найпоширенішими моделями зрілості SPI. На рисунку 3.4 можна помітити, що для учасників найпоширенішими моделями зрілості SPI є CMMI, CMM та ISO/IEC 15504 / ISO 33001.

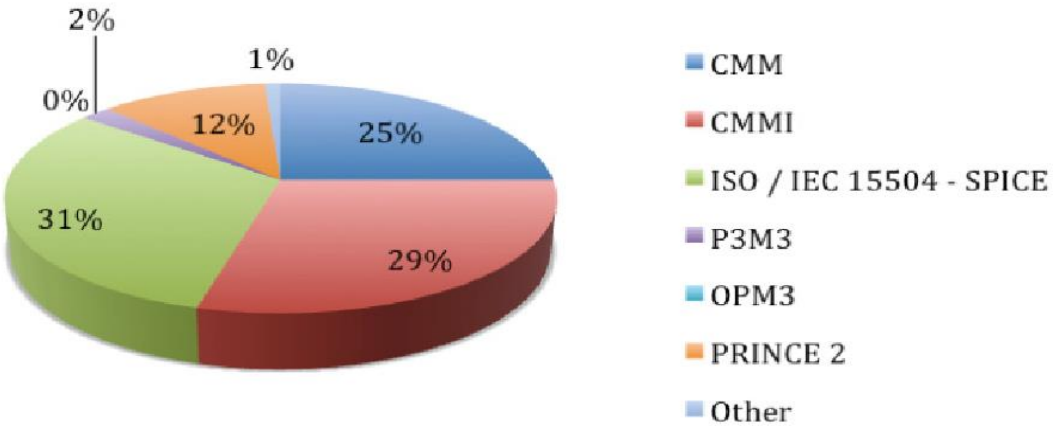

Рисунок 1.4 – Найпоширеніші моделі зрілості SPI

Наступне запитання стосується часу, протягом якого учасники використовували найпоширеніші моделі зрілості SPI.

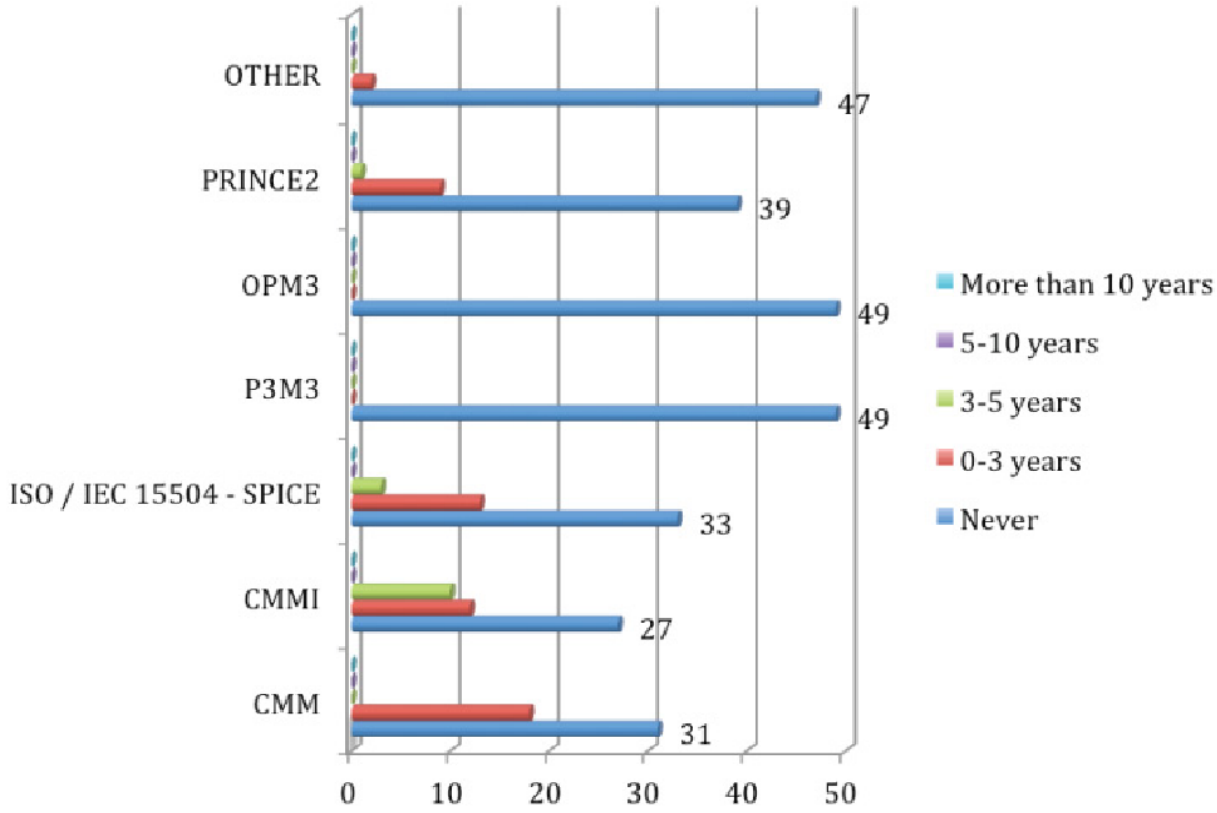

Рисунок 3.5 – Час застосування найпоширеніших моделей зрілості SPI

Як видно з рис. 3.5 майже половина учасників ніколи не використовували жодну з моделей зрілості SPI. Однак близько 10 учасників використовували ISO / IEC 15504, PRINCE2 і CMMI від декількох місяців до 3 років, тоді як біля десяти з них використовували CMMI більше 3 років. Що стосується CMM, то 18 учасників використовували його протягом періоду, меншого 3 років. Особливу увагу слід звернути на те, що ніхто з учасників ніколи не користувався OPM3 і P3M3.

Відповіді на питання, котре стосується можливих результатів, які можуть представити моделі зрілості SPI, представлені на наступному рисунку (див. рис. 3.6). З нього можна відмітити, що учасники стверджували, що моделі

зрілості SPI забезпечують кращий контроль і видимість під час процесу розробки.

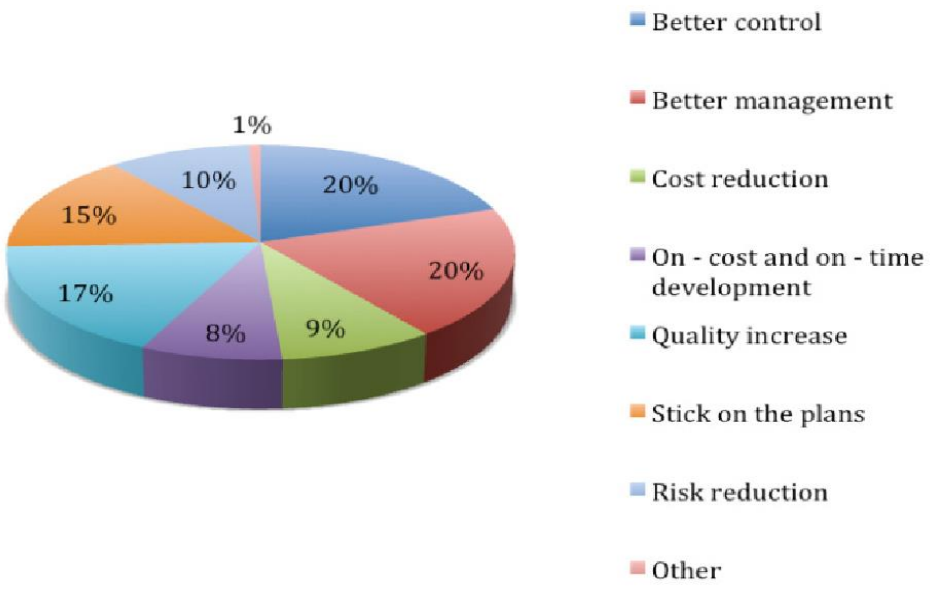

Рисунок 3.6 – Результати впровадження моделей зрілості процесів розробки

Крім того, вони сприяють підвищенню якості програмного продукту, а розробка виконується в рамках часових обмежень.

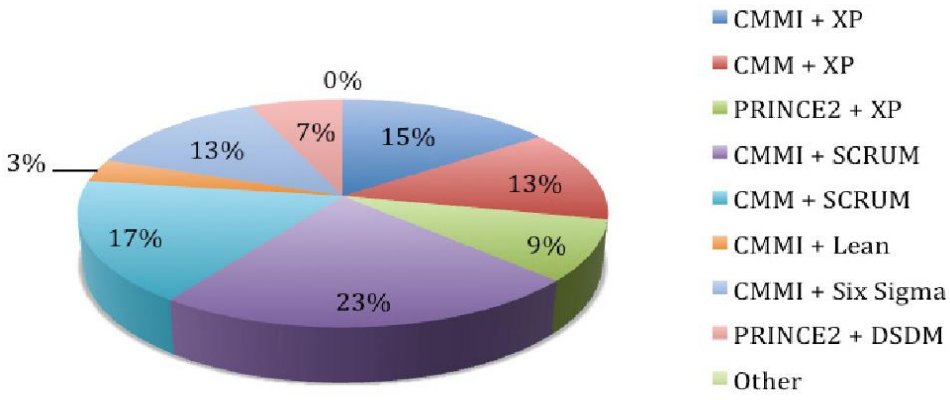

Рисунок 3.7 – найпоширеніші комбінації гнучких методологій з моделями SPI

На рис. 3.7 бачимо, що найпоширенішою комбінацією, на думку учасників, є CMMI та Scrum. Нижче наведено комбінації CMM і Scrum, CMMI і XP, CMMI і Six Sigma, а також CMM і XP.

Ще одне запитання стосується часу, протягом якого учасники використовували комбінацію гнучких методологій і моделей зрілості SPI. На рисунку 3.8 можна легко помітити, що найпоширенішими комбінаціями є CMM і Scrum, а також CMMI і Scrum. Слід звернути особливу увагу на те, що більшість учасників опитування ніколи не використовували жодної з найпоширеніших комбінацій.

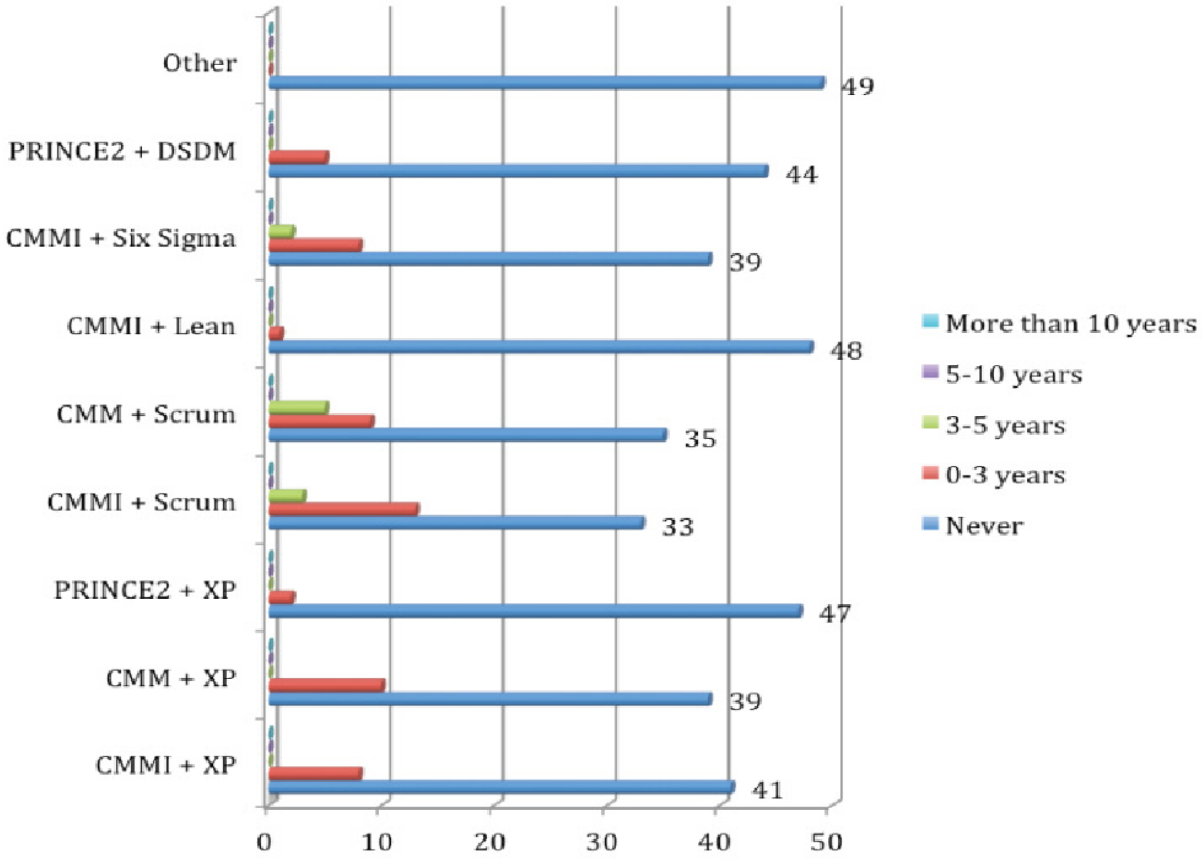

Рисунок 3.8 – Час застосування комбінації гнучких методологій та моделей зрілості SPI

Тим не менш, близько 8 учасників використовували комбінації CMMI і Six Sigma, CMM і Scrum, CMM і XP і CMMI і XP менше ніж 3 роки, тоді як 13 використовували CMMI і Scrum від декількох місяців до 3 років. Крім того,

комбінації CMMI і Six Sigma, CMM і Scrum, а також CMMI і Scrum використовувались терміном не менше 5 років.

Наступне запитання стосується можливих результатів, які можуть отримати комбінації гнучких методологій і моделей зрілості SPI. Як показано на рисунку 3.9, учасники стверджували, що комбінації покращують комунікацію між клієнтами, менеджерами та командою розробників; вони підвищують якість програмного продукту, розробка виконується в рамках часових і фінансових обмежень, а також забезпечується прозорість проєкту, управління та контроль процесу розробки.

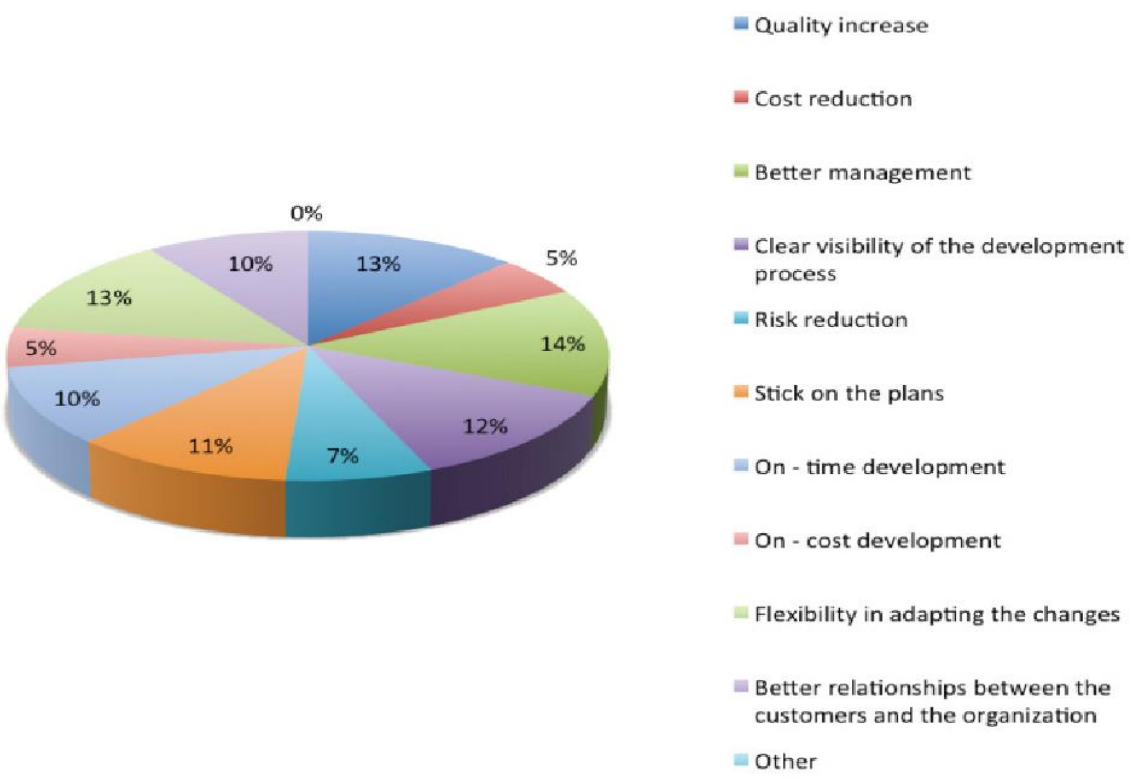

Рисунок 3.9 – Результати застосування комбінацій гнучких методологій і моделей зрілості SPI

Ці запитання ставили практиків, чи є у них якісь інші пропозиції щодо вдосконалення розробки програмного забезпечення та чи використовували вони будь-які інші методології, про які автори не згадують. Відповіді, які вони надали, такі:

• Спочатку потрібно визначити методологію, яку ви збираєтеся використовувати, і потім почати розробляти програмний код.

• Варто налагодити хорошу комунікацію між менеджерами, командою розробників і клієнтами. Крім того, вимоги до продукту мають бути зрозумілі всім залученим сторонам.

• Ранньому етапові розробки варто приділити найретельнішу увагу. Пізніше на цій основі забезпечується жорсткий контроль процесу.

• Не існує єдиної правильної методології, оскільки методів занадто багато. Різні комбінації можуть дати різні результати.

• На сьогоднішній день Agile вважається найкращою практикою, яка робить процес розробки програмного забезпечення швидким, гнучким щодо змін і дозволяє вчасно здати проєкт. Хоча в цьому випадку є ризик зниження якості, але якість можна покращити за допомогою стандартів CMMI або ISO.

• Варто організувати навчання команди розробників та їх мотивацію до навчання. Також вітається зміна обов'язків і ролей, щоб допомогти членам команди мати загальне уявлення про різні частини продукту.

3.2 Узагальнення опитування розробників

Таким чином було виявлено, що учасники знайомі з найпоширенішими гнучкими методологіями, такими як Scrum, XP, адаптивна розробка програмного забезпечення, розробка, керована функціями, і методологія динамічної розробки систем. Вони використовували Scrum, XP і ASD протягом часу від 3 до 5 років у різних проєктах. Крім того, вони використовували всі інші методології принаймні один раз, однак протягом менше 3 років.

З метою практичного впровадження було виявлено, що розробникам відомі основні моделі зрілості вдосконалення процесу програмного забезпечення, а саме респонденти знають CMM, CMMI, ISO / IEC 15504 і PRINCE2.

Стосовно поєднання методологій Agile та моделей зрілості SPI, то лише невелика частина опитаних знайома з основними комбінаціями, такими як CMM і XP, CMMI і XP, CMM і Scrum, CMMI і Scrum, CMMI і Six Sigma, PRINCE2 і DSDM і PRINCE2 і XP.

За словами практиків, переваги, які спостерігаються завдяки використанню цих комбінацій, полягають у підвищенні якості та зниження вартості розробки і затраченого часу. Крім того, покращується комунікація між клієнтами, компанією і командою розробників, а отже, підвищується задоволеність клієнтів. Нарешті, є чітке бачення і краще управління процесом розробки.

Варто зазначити, що для більшості комбінацій інформація, яка існує в літературі та в реальних прикладах по галузі, дуже обмежена. Більше того, комбінації стосуються більших компаній, і в результаті керівники проектів повинні були коригувати менеджмент, наприклад створити менші артефакти проєкту та спробувати спростити процеси, які вимагають комбінації. Крім того, необхідне навчання команди розробників, ліцензія на моделі зрілості SPI не завжди безкоштовна, і підприємству доводиться змінювати свою культуру.

На завершення респонденти зазначили, що для того, щоб підприємство було успішним, має бути налагоджена хороша комунікація між менеджерами та командою розробників. Крім того, зі свого досвіду вони помітили, що спочатку потрібно вирішити, яку методологію чи комбінацію вони збираються використовувати, а потім розпочати розробку коду. Вони підсумовують, що не існує правильної чи неправильної методології. Натомість для досягнення максимальних результатів підприємству доцільно дотримуватися методології або комбінації, з якою вони більше знайомі.

Гнучкі методології, такі як Scrum, XP, адаптивна розробка програмного забезпечення та розробка, керована функціями, добре знайомі учасникам. Окрім методології Agile, більшість учасників знайомі з основними методологіями SPI, такими як CMM, CMMI, SPICE та PRINCE2. Однак лише третина учасників знайома з комбінаціями Agile та SPI.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Синдром професійного вигорання в ІТ

У сучасних дослідженнях наголошується, що вигорання виникає у фахівця в контексті роботи і має негативні наслідки як для нього самого, так і для організації в цілому, психічного благополуччя всіх тих, з ким він взаємодіє в процесі професійної діяльності (клієнтів, учнів, пацієнтів, колег) [31]. Подальше вивчення даного феномену важливе для дослідження закономірностей і механізмів професійного становлення механізмів професійної адаптації; виявлення зв'язку між професійним стресом і соматичними розладами, та встановлення характеру цього зв'язку тощо.

Дослідниками з часу появи перших даних про вигорання як професійного феномену неухильно наголошується його поширеність. Якщо раніше він досліджувався у представників професій «людина – людина» – вчителів, викладачів коледжів, лікарів і медичних сестер, менеджерів, торгівельних агентів, соціальних працівників, представників інших професій, пов'язаних з безперервними комунікаціями з великим потоком людей, то тепер коло професіоналів, схильних до професійного вигорання, розширилося. Дослідниками отримані дані про наявність вигорання у представників управлінських структур і у представників професії «людина – знак», «людина – машина» – бухгалтерів, програмістів, пілотів, водіїв, операторів та ін. У зв'язку з цим, дослідження феномену професійного вигорання виступає одному з пріоритетних завдань сучасної науки [18].

Якісний і кількісний склад вигорання залежить від змісту професійної діяльності: у професіях «суб'єкт-суб'єктної» сфери вигорання має трикомпонентну структуру, а в «суб'єкт-об'єктній» сфері – близьку до двофакторної структури. Психічне вигорання набуває статусу загально професійного феномену і розглядається як компонент категоріального апарату психологічної науки. Даний феномен виявляється в різних сферах особистості –

емоційній, когнітивній, мотиваційній, сфері відношення людини до роботи – і найменше розроблена проблема впливу вигорання на поза професійні сфери життя людини [32].

Аналіз сучасних досліджень показав, що «психічне вигорання» ширше поняття, ніж «професійне вигорання», оскільки воно може бути викликане різними причинами – особовими, родинними, професійними. Психічне вигорання може виявлятися не лише в професійній сфері, а, наприклад,у сім'ї, в учбовій діяльності. Професійне вигорання становить один з варіантів психічного вигорання: перше частіше використовується в контексті трудової діяльності. Поняття «емоційне вигорання» вживається в тому разі, коли акцент робиться на емоційній складовій вигорання і йдеться про те, що вигоряє лише емоційна сфера психіки. Термін психічне вигорання використовується в разі, коли акцентується увага на тому, що вигорання зачіпає емоційну, інтелектуальну та мотиваційно-споживацьку сфери, а також вольовий механізм. На сучасному етапі вигорання вивчається одночасно як соціальний, професійний та особистісний феномени.

Отже, в контексті теоретичного вивчення проблем професійного вигорання виділяють основні етапи. Перший етап присвячений виявленню й опису окремих випадків вигорання. На другому етапі відбувається розуміння та узагальнення феномену вигорання. Третій період відзначається методологічною фазою дослідження вигорання. Зміст четвертого етапу полягає в аналітичному вивченні узагальненого концепту вигорання. Сучасний етап вивчення питання професійного вигорання характеризується появою нових напрямів досліджень, зокрема він характеризується різними сенсами, які зв'язуються з феноменом вигорання.

4.2 Створення і функціонування системи моніторингу довкілля з метою інтеграції екологічних інформаційних систем, що охоплюють певні території

Законом України "Про охорону навколишнього природного середовища" (ст.20, 22) передбачено створення державної системи моніторингу довкілля (ДСМД) та проведення спостережень за станом навколишнього природного середовища, рівнем його забруднення. Виконання цих функцій покладено на Мінприроди та інші центральні органи виконавчої влади, які є суб'єктами державної системи моніторингу довкілля, а також підприємства, установи та організації, діяльність яких призводить або може призвести до погіршення стану довкілля.

Основні принципи функціонування ДСМД визначені у постанови Кабінету Міністрів України від 30.03.1998 № 391 „Про затвердження Положення про державну систему моніторингу довкілля".

На даний час, у державній системі моніторингу довкілля (далі – ДСМД) функції і задачі спостережень та інформаційного забезпечення виконують 8 суб`єктів системи моніторингу: Мінприроди, МНС, МОЗ, Мінагрополітики, Мінжитлокомунгосп, Держводгосп, Держкомлісгосп, Держкомзем.

Кожний із суб`єктів ДСМД здійснює моніторинг тих об`єктів довкілля, що визначаються Положенням про державну систему моніторингу довкілля та порядками і положеннями про державний моніторинг окремих складових довкілля. Основні нормативні акти, що регламентують моніторинг об'єктів довкілля:

– постанова Кабінету Міністрів України від 09.03.1999 № 343 «Про затвердження Порядку організації та проведення моніторингу в галузі охорони атмосферного повітря»;

– постанова Кабінету Міністрів України від 20.07.1996 № 815 «Про затвердження Порядку здійснення державного моніторингу вод»;

– постанова Кабінету Міністрів України від 20.08.1993 № 661 «Про затвердження Положення про моніторинг земель»;

– постанова Кабінету Міністрів України від 26.02.2004 № 51 «Про затвердження Положення про моніторинг ґрунтів на землях сільськогосподарського призначення».

З метою координації діяльності міністерств та відомств, визначення основних принципів державної політики з питань розвитку системи моніторингу навколишнього середовища, забезпечення її функціонування на основі єдиного нормативно-методологічного забезпечення постановою Кабінету Міністрів України від 17.11.2001 № 1551 утворено Міжвідомчу комісію з питань моніторингу довкілля. Мінприроди здійснюється організаційно-технічне забезпечення роботи комісії та її профільних секцій.

Існуюча система моніторингу довкілля базується на виконанні розподілених функцій її суб'єктами і складається з підпорядкованих їм підсистем. Кожна підсистема на рівні окремих суб'єктів системи моніторингу має свою структурно-організаційну, науково-методичну та технічну бази.

Функціонування ДСМД здійснюється на трьох рівнях, що розподіляються за територіальним принципом:

– загальнодержавний рівень, що охоплює пріоритетні напрямки та завдання моніторингу в масштабах всієї країни;

– регіональний рівень, що охоплює пріоритетні напрямки та завдання в масштабах територіального регіону;

– локальний рівень, що охоплює пріоритетні напрямки та завдання моніторингу в масштабах окремих територій з підвищеним антропогенним навантаженням.

Моніторинг якості повітря.

Державною гідрометеорологічною службою (МНС) здійснюються спостереження за забрудненням атмосферного повітря у 53 містах України на 162 стаціонарних, двох маршрутних постах спостережень та двох станціях

транскордонного переносу. Ведуться спостереження за хімічним складом атмосферних опадів та за кислотністю опадів.

Програма обов`язкового моніторингу якості атмосферного повітря включає сім забруднюючих речовин: пил, двоокис азоту (NO_2), двоокис сірки (SO_2), оксид вуглецю, формальдегід (H_2CO), свинець та бенз(а)пірен. Деякі станції здійснюють спостереження за додатковими забруднюючими речовинами. Проводиться аналіз наявності забруднюючих речовин в опадах та сніговому покриві.

Державна екологічна інспекція (Мінприроди) здійснює вибірковий відбір проб на джерелах викидів. Вимірюється понад 65 параметрів.

Санітарно-епідеміологічна служба (МОЗ) здійснює спостереження за якістю атмосферного повітря у житловій та рекреаційній зонах, зокрема поблизу основних доріг, санітарно-захисних зон та житлових будинків, на території шкіл, дошкільних установ та медичних закладів в містах та в робочий зоні. Крім того, здійснюється аналіз якості повітря у житловій зоні за скаргами мешканців.

Моніторинг стану вод суші.

Державна гідрометеорологічна служба (МНС) проводить моніторинг гідрохімічного стану вод на 151 водному об`єкті, а також здійснює гідробіологічні спостереження на 45 водних об`єктах. Отримуються дані по 46 параметрах, що дають можливість оцінити хімічний склад вод, біогенні параметри, наявність зважених часток та органічних речовин, основних забруднюючих речовин, важких металів та пестицидів. На 8 водних об`єктах проводяться спостереження за хронічною токсичністю води. Визначаються показники радіоактивного забруднення поверхневих вод.

Державна екологічна інспекція (Мінприроди) відбирає проби води та отримує дані по 60 вимірюваних параметрах.

Державний комітет по водному господарству проводить моніторинг річок, водосховищ, каналів, зрошувальних систем і водойм у межах водогосподарських систем комплексного призначення, систем водопостачання, транскордонних водотоків та водойм у зонах впливу атомних електростанцій.

Контроль якості води за фізичними та хімічними показниками здійснюється на 72 водосховищах, 164 річках, 14 зрошувальних системах, 1 лимані та 5 каналах комплексного призначення. Крім того, у рамках радіаційного моніторингу вод водогосподарськими організаціями здійснюється контроль вмісту радіонуклідів у поверхневих водах.

Санітарно-епідеміологічна служба (МОЗ) проводить спостереження за джерелами централізованого та децентралізованого постачання питної води, а також місцями відпочинку вздовж річок та водосховищ.

Підприємствами Державної геологічної служби (Мінприроди) здійснюється моніторинг стану підземних вод. У місцях моніторингу проводиться оцінка рівня залягання підземних вод (наявність), їх природного геохімічного складу. Проводяться визначення 22 параметрів, в тому числі концентрації важких металів та пестицидів.

Санітарно-епідеміологічна служба (МОЗ) здійснює хімічний аналіз підземних вод, які призначаються для питного споживання.

Моніторинг прибережних вод.

Державна гідрометеорологічна служба (МНС) управляє мережею моніторингу стану прибережних вод, яка складається з станцій моніторингу у місцях скиду стічних вод та науково-дослідних станцій, що розташовані на прибережних територіях Чорного та Азовського морів. На існуючих станціях проводяться вимірювання від 16 до 26 гідрохімічних параметрів вод та донних відкладів.

Державні інспекції охорони Чорного та Азовського морів (Мінприроди) мають власні системи спостережень. До їх повноважень відносяться щомісячні відбори проб та аналіз впливу джерел забруднення, які розташовані на узбережжі; моніторинг скидів з кораблів; забруднення від діяльності з пошуку та видобування нафти, газу і будівельних матеріалів на морському шельфі; нагляд за використанням живих ресурсів моря.

Державна санітарно-епідеміологічна служба (МОЗ) здійснює моніторинг якості морської води в зонах рекреаційного та оздоровчого водокористування.

Моніторинг стану ґрунтів.

Державна гідрометеорологічна служба (МНС) здійснює моніторинг забруднення ґрунтів сільськогосподарських земель пестицидами та важкими металами у населених пунктах. Проби відбираються раз у п`ять років, проби на важкі метали у містах Костянтинівка та Маріуполь відбираються щороку.

Державна екологічна інспекція (Мінприроди) здійснює відбір проб на промислових майданчиках в межах країни. Загальна кількість параметрів, що вимірюються – 27.

Установи МОЗ здійснюють моніторинг стану ґрунтів на територіях їх можливого негативного впливу на здоров`я населення. Найбільше охоплені території вирощення сільськогосподарської продукції, території в місцях застосування пестицидів, ґрунти в зоні житлових масивів, дитячих майданчиків та закладів. Досліджуються проби ґрунту в місцях зберігання токсичних відходів на території підприємств та поза територією підприємств у місцях їх складування або захоронення.

Мінагрополітики здійснює спостереження за ґрунтами сільськогосподарського використання. Здійснюються радіологічні, агрохімічні та токсикологічні визначення, залишкова кількість пестицидів, агрохімікатів і важких металів.

Моніторинг показників біологічного різноманіття.

Через обмежене бюджетне фінансування моніторинг здійснюється тільки за видами, які представляють промисловий інтерес (дерева, риба, дичина).

Підприємства Держкомлісгоспу проводять моніторинг лісової рослинності у 24 областях країни. Здійснюється оцінка біомаси, пошкодження її біотичними та абіотичними чинниками; мисливської фауни, біорізноманіття; радіологічні визначення.

Деякі дослідження здійснюються через надання міжнародної допомоги, або в рамках міжнародних програм.

Моніторинг радіаційного випромінювання.

Державна гідрометеорологічна служба (МНС) здійснює спостереження за радіоактивним забрудненням атмосфери шляхом щоденних замірів доз гамма-радіаційної експозиції (ГРЕ), осідання радіоактивних частинок з атмосфери та вмісту радіоактивного аерозолю в повітрі. Здійснюються заміри радіоактивного забруднення поверхневих вод на 8 водних об`єктах. Поблизу атомних електростанцій Державна гідрометеорологічна служба здійснює заміри радіоактивного забруднення поверхневих вод цезієм-137 у та забруднення ґрунтів.

Лабораторії моніторингу Мінагрополітики проводять контроль у місцях концентрації радіоактивних речовин у ґрунтах та харчових продуктах. МНС здійснює моніторинг доз ГРЕ на 10 автоматизованих пунктах поблизу атомних електростанцій. У межах 30-кілометрової зони навколо Чорнобильської АЕС (зони відчуження), МНС здійснює спостереження за концентрацією радіонуклідів; радіонуклідами в атмосферних опадах, а також концентрацією «гарячих» частинок у повітрі. Міжнародна радіоекологічна лабораторія Чорнобильського центру атомної безпеки, радіоактивних відходів та радіоекології у Славутичі, здійснює моніторинг впливу радіації на біоту у зоні відчуження.

Суб`єктами ДСМД створені, або розробляються відомчі бази даних моніторингової інформації. Існуюча система інформаційної взаємодії відомчих підсистем моніторингу довкілля передбачає обмін інформацією на загальнодержавному та регіональному рівнях. Організаційна інтеграція суб`єктів моніторингу довкілля на всіх рівнях здійснюється Мінприроди та його територіальними органами.

Для упорядкування процесу обміну інформацією за показниками та термінами надання екологічної інформації між Мінприроди та суб’єктами ДСМД укладено двохсторонні угоди про співробітництво у сфері моніторингу навколишнього природного середовища, до яких розроблені відповідні регламенти обміну екологічною інформацією.

Оперативна моніторингова інформація передається територіальними органами суб'єктів ДСМД до регіональних центрів моніторингу довкілля, або державних управлінь охорони навколишнього природного середовища в регіонах.

Узагальнена аналітична інформація надається міністерствами та відомствами-суб'єктами ДСМД Мінприроди.

Отримані дані передаються до Інформаційно - аналітичного центру Мінприроди та накопичується у банках екологічних даних.

На основі отриманої щомісячної та щоквартальної інформації Мінприроди видається інформаційно – аналітичний огляд „Стан довкілля в Україні ”, який розповсюджується серед заінтересованих користувачів.

Функціонування Інформаційно-аналітичного центру Мінприроди забезпечує інформаційний обмін з регіональними центрами моніторингу довкілля, суб’єктами державної системи моніторингу довкілля, створення уніфікованого банку екологічних даних, проведення комплексного аналізу стану довкілля, тощо.

Постановою Кабінету Міністрів України від 05.12.2007 № 1376 затверджено Державну цільову екологічну програму проведення моніторингу навколишнього природного середовища. З метою забезпечення інтеграції інформаційних ресурсів суб`єктів системи моніторингу довкілля передбачено створення та забезпечення функціонування єдиної автоматизованої підсистеми збору, оброблення, аналізу і збереження даних та інформації, отриманих в результаті здійснення моніторингу.

В межах Державної цільової екологічної програми проведення моніторингу навколишнього природного середовища, у тому числі, передбачено розширення мережі автоматизованих постів спостережень за забрудненням атмосферного повітря в екологічно небезпечних містах.

ВИСНОВКИ

Загальна мета магістерської роботи полягала в тому, щоб зрозуміти, як поєднання гнучких методологій і моделей зрілості SPI можна застосувати в малих та середніх компаніях, які можливі переваги та недоліки, які можуть виникнути, і чи вони дійсно застосовуються на підприємствах, як описано в літературі. Основна мета поєднань полягає в тому, щоб вони намагалися застосувати найкращі частини кожної методології Agile та моделі зрілості SPI, щоб створити нову методологію, яка містить усі ці ключові концепції. Більше того, кожна методологія має бути трансформована, щоб задовольнити потреби кожного підприємства.

Щоб зрозуміти все вищезгадане, в роботі дано відповіді на наступні задачі дослідження.

RQ1: «Які існують гнучкі методики в поєднанні з моделями зрілості покращення процесів розробки програмного забезпечення, створеними для малих і середніх підприємств?»

Щоб дати відповіді на це питання дослідження, було проведено огляд літератури відповідної тематики. Завдяки цьому дослідженню виявлено наступні найпоширеніші комбінації:

- CMMI і XP,
- CMM і XP,
- PRINCE2 і XP,
- CMMI та Scrum,
- CMM і Scrum,
- CMMI та Lean,
- CMMI і шість сигм,
- PRINCE2 і DSDM.

RQ2: «У яких ситуаціях і як можна застосувати ці методології? Які результати, переваги та потенційні недоліки може мати кожна методологія? Коли слід застосовувати кожну методологію? Які причини невдачі?»

Кожна комбінація може бути застосована різними способами та має різні переваги та недоліки. Виявлено, що підприємства зазвичай спочатку дотримуються методології Agile, а потім переходять до прийняття моделі зрілості SPI.

Загальні переваги, які можуть виникнути майже в кожній комбінації, це підвищення якості проекту програмного забезпечення, краще управління та прозорість проєкту в процесі розробки.

Тим не менш, моделі зрілості SPI не завжди безкоштовні, і вони адресовані більшим організаціям. Крім того, потрібен кваліфікований персонал. Проте було доведено, що якщо МСП внесуть невеликі коригування, наприклад, створять менші артефакти або спростять процеси, яких вимагає методологія, вони зможуть прийняти ці комбінації.

RQ3: «Чи справді ці методології можна застосовувати в малих та середніх підприємствах?»

Результати дослідження показують, що хоча малі та середні підприємства стикаються з серйозними труднощами під час впровадження цих методологій, шляхом виконання невеликих коригувань вони зможуть прийняти комбінації, згадані вище.

Комбінації, які дійсно застосовуються, це ті, які включають CMM, CMMI та Scrum. Зокрема, CMMI та XP, CMM та XP, CMM та Scrum, CMMI та Scrum, CMMI та Six Sigma.

З усіх досліджень, які були використані для цієї кваліфікаційної роботи, можна сказати, що застосування комбінації різних методологій принесе користь процесу розробки програмного забезпечення. Оскільки гнучкі методології та моделі зрілості SPI застосовуються по-різному, залежно від середовища, де вони застосовуються, отримані переваги від застосування комбінації відрізнятимуться від одного випадку до іншого. Тим не менш, можна бути впевненим, що будуть отримані позитивні результати, оскільки ідея поєднання полягає в застосуванні практик, які можуть бути «цікавими» для надання додаткової цінності процесу створення програмного забезпечення.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Strutynska, I., Kozbur, H., Dmytrotsa, L., Bodnarchuk, I., & Hlado, O. (2019, October). Small and medium business structures clustering method based on their digital maturity. In 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T) (pp. 278-282). IEEE.

2. B. Kitchenham and S. Charters, “Guidelines for performing systematic literature reviews in software engineering,” Guid. Perform. Syst. Lit. Rev. Softw. Eng., 2007.

3. F. L. Ribeiro and M. T. Fernandes, “Exploring agile methods in construction small and medium enterprises: a case study,” Journal of Enterprise Information Management, vol. 23, no. 2, pp. 161–80, 2010.

4. M. Y. al-Tarawneh, M. S. Abdullah, and A. B. M. Ali, “A proposed methodology for establishing software process development improvement for small software development firms,” Procedia Computer Science, vol. 3, pp. 893–897, 2011.

5. “Software process improvement in small and medium software enterprises: a systematic review - Springer.”

6. H. Kaindl, S. Brinkkemper, J. A. Bubenko Jr, B. Farbey, S. J. Greenspan, C. L. Heitmeyer, J. at al., “Requirements Engineering and Technology Transfer: Obstacles, Incentives and Improvement Agenda,” Requirements Engineering, vol. 7, no. 3, pp. 113–123, Sep. 2002.

7. S. Jantunen, “Exploring software engineering practices in small and medium-sized organizations,” in Proceedings of the 2010 ICSE Workshop on Cooperative and Human Aspects of Software Engineering, New York, NY, USA, 2010, pp. 96–101.

8. A. Simon, R. Alain, H. Naji: “OWPL: A Gradual Approach for Software Process Im- provement In SMEs”. In: Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO-SEAA 2006) (2006).

9. E. Kabaale and J. Nabukenya, “A Systematic Approach to Requirements Engineering Process Improvement in Small and Medium Enterprises: An Exploratory

Study," in Product-Focused Software Process Improvement. 12th International Conference, PROFES 2011, 20-22 June 2011, 2011, pp. 262–75.

10. A. Fuggetta, "Software Process: A Roadmap," Proc. Conf. Future of Software Eng., pp. 25-34, 2000.

11. N. Ehsan, A. Perwaiz, J. Arif, E. Mirza, and A. Ishaque, "CMMI / SPICE based process improvement," in 2010 IEEE International Conference on Management of Innovation and Technology (ICMIT), 2010, pp. 859–862.

12. SEI, CMMI for I, Version 1.2. Technical Report CMU/SEI-2006-TR-008., 2006. http://www.sei.cmu.edu/cmmi/

13. S. Garcia, S. Cepeda, J.M. Staley, G. Miluk, "Summary of CMMI For Small Business Pilot Study in Huntsville. Alabama, USA", Presented at Info Seminar, June 2004, Carnegie Mellon. Software Engineering Institute, Pittsburgh, USA

14. PRINCE2 Maturity Model (P2MM) // Available at https://www.prince2.com/eur/prince2-maturity-models

15. Z. Al-Zoabi, "Introducing Discipline to XP: Applying PRINCE2 on XP Projects," in 3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008, 2008, pp. 1–7.

16. A. Omran, "AGILE CMMI from SMEs perspective," in 3rd International Conference on Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008, 2008, pp. 1 –8.

17. K. Beck, A. Cockburn, R. Jeffries, and J. Highsmith. Agile Manifesto. http://www.agilemanifesto.org . 2001. 12-4-2002.

18. North American and European Enterprise Software and Services Survey, Business Technographics Ed., 2005.

19. D. Turk, R. France, and B. Rumpe, "Assumptions underlying agile software-development processes," J. Datab. Manage., vol. 16, 2005.

20. D. Cohen, M. Lindvall, and P. Costa, "Agile software development," Data & Analysis Center for Software (DACS), New York, 2003.

21. B. J. Hicks, and J. Matthews, 2010. "The barriers to realizing sustainable process improvement: a root cause analysis of paradigms for manufacturing systems

improvement", International Journal of Computer Integrated Manufacturing, 23 (7), 585–602.

22. ISO/IEC 15504 Information technology – Process assessment, (Software Process Improvement and Capability Determination – SPICE)

23. ISO/IEC 33001:2015 - Information technology – Process assessment – Concepts and terminology

24. T. Dyba and T. Dingsoyr, "Empirical studies of agile software development: A systematic review," Information and Software Technology, vol. 50, no. 9–10, pp. 833–859, Aug. 2008.

25. M. Pikkarainen and A. Mäntyniemi, "An approach for using CMMI in agile software development assessments: experiences from three case studies," in SPICE 2006 conference, Luxemburg, 2006, pp. 4–5.

26. K. C. Dangle, P. Larsen, M. Shaw, and M. V. Zelkowitz, "Software process improvement in small organizations: a case study," IEEE Software, vol. 22, no. 6, pp. 68 – 75, Dec. 2005.

27. M. I. Khan, M. A. Qureshi, and Q. Abbas, "Agile methodology in software development (SMEs) of Pakistan software industry for successful software projects (CMM framework)" in 2010 International Conference on Educational and Network Technology (ICENT), 2010, pp. 576–580.

28. Z. Lina and S. Dan, "Research on Combining Scrum with CMMI in Small and Medium Organizations," in 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE), 2012, vol. 1, pp. 554–557.

29. A. S. C. Marcal, B. C. C. de Freitas, F. S. Furtado Soares, and A. D. Belchior, "Mapping CMMI Project Management Process Areas to SCRUM Practices," in 31st IEEE Software Engineering Workshop, 2007, pp. 13–22.

30. C. R. Jakobsen and T. Poppendieck, "Lean as a Scrum Troubleshooter," in Agile Conference (AGILE), 2011, 2011, pp. 168 – 174.

31. R. Jha and A. K. Saini, "ERP Redefined: Optimizing Parameters with Lean Six Sigma for Small & Medium Enterprises," in 2011 International Conference on Communication Systems and Network Technologies (CSNT), 2011, pp. 683 – 687.

32. W. C. de Souza Carvalho, P. F. Rosa, M. dos Santos Soares, M. A. Teixeira da Cunha Junior, and L. C. Buiatte, “A Comparative Analysis of the Agile and Traditional Software Development Processes Productivity,” in Computer Science Society (SCCC), 2011. 30th International Conference of the Chilean, 2011, pp. 74–82.

33. DSDM with PRINCE2 Task Group. “Using DSDM with PRINCE2,” DSDM consortium 2000.

34. Koutsoumpos, V., & Marinelarena, I. (2013). Agile methodologies and software process improvement maturity models, current state of practice in small and medium enterprises. Blekinge Institute of Technology, Department of Software Engineering. Unpublished Master’s Thesis.

35. Вовк О. В. Особливості синдрому професійного вигорання в працівників сфери інформаційних технологій. [Електронний ресурс]. – Режим доступу: http://maup.com.ua/assets/files/psihologz/2019-1/02.pdf

36. Назарук Н. Каузально-телеологічний формат профілактики «професійного вигорання» вчителя / Н. Назарук // Психологія особистості. 2012.- No 1 (3). – С. 119–128.

37. Вдосконалення охорони праці в ІТ-індустрії. // Харківський національний дорожний університет. [Електронний ресурс]. – Режим доступу: https://www.khadi.kharkov.ua/fileadmin/P_vcheniy_secretar/%D0%9E%D0%A5%D0%9E%D0%A0%D0%9E%D0%9D%D0%90_%D0%9F%D0%A0%D0%90%D0%A6%D0%86/R_IT-INDUSTRIA.pdf

ДОДАТКИ

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

М А Т Е Р І А Л И

X НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

7–8 грудня 2022 року

ТЕРНОПІЛЬ
2022

УДК 004.41
В. Волович, Б. Береженко, І. Боднарчук
(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ЗАДАЧА ПРОЄКТУВАННЯ ПРОГРАМНОЇ АРХІТЕКТУРИ В ПРОЦЕСАХ ЗАБЕЗПЕЧЕННЯ ЯКОСТІ

UDC 004.41
V. Volovych, B. Berezhenko, I. Bodnarchuk

THE PROBLEM OF SOFTWARE ARCHITECTURE DESIGN IN THE PROCESSES OF QUALITY ASSURANCE

При проектуванні програмних систем (ПС) широко застосовується компонентна технологія, яка базується на використанні компонентів, взятих з раніше виконуваних проектів (компоненти повторного використання.

Архітектура в цій технології проектується шляхом вибору каркасу, на основі вимог до ПС, і заповненням його необхідними компонентами, взятими з репозиторію. Оскільки існує декілька компонентів, які реалізують одну і ту ж функціональність, то отримаємо множину альтернативних архітектур ПС. Для вибору найбільш прийнятного рішення необхідно або проранжувати альтернативи відносно значень критеріїв якості ПС, або використати деякий інтегральний критерій, значення якого обчислюється для кожної альтернативи.

На практиці використовується декілька методів оцінювання якості програмної архітектури. Найбільш відомими з них є методи, які базуються на розробці сценаріїв використання та перевірки, чи задовольняє даний варіант архітектури вимозі по певному критерію якості. Найбільш відомими з них є методи SAAM, ATAM і CBAM [1].

Спільним недоліком перерахованих методів є те, що для їх реалізації необхідно створювати та аналізувати експертам достатньо велику кількість сценаріїв використання, що робить їх трудомісткими, вартісними і складними для формалізації. Тому поява робіт, в яких для рішення цієї задачі було використано метод аналізу ієрархій (МАІ), дозволила значно покращити процедуру вибору архітектури і формалізувати її для автоматизації процесів прийняття рішень [2].

Суттєвим недоліком застосування МАІ є обмежена кількість альтернатив, які можна оцінювати одночасно ($n \leq 7 \pm 2$), що породжується неузгодженістю елементів матриць парних порівнянь, яка збільшується при збільшенні кількості альтернатив [3].

Для вирішення цієї проблеми О. Павловим в [4] запропонована модифікація МАІ, в якій вагові множники альтернатив визначаються з умови мінімізації неузгодженості матриці парних порівнянь, що приводить вихідну задачу до задачі математичного програмування. В роботах [5], [6] розглянуті питання застосування модифікованого МАІ (ММАІ) до задачі оцінювання альтернативних варіантів архітектури програмних систем при великій кількості альтернатив.

Остаточний вибір варіанта архітектури з врахуванням сукупності критеріїв частіше всього виконується заміною задачі багатокритерійного вибору архітектури однокритерійною, з використанням лінійної адитивної згортки частинних критеріїв. Її використання є обґрунтованим лише у невеликому околі базових точок.

Також проблематичним є призначення ваг частинних критеріїв у скалярній згортці експертним методом який є неформалізованим, носить суб'єктивний характер і служить додатковим джерелом похибок Для вирішення вказаних проблем необхідно вибрати прийнятну структуру скалярної згортки і застосувати формалізовані методи визначення ваг частинних критеріїв.

Для цього пропонується використовувати універсальну скалярну згортку запропоновану в [7]. В ній оптимізується цільова функція, яка залежить від міри напруженості ситуації, котра визначається близькістю значень критеріїв до своїх обмежень. Схема задачі оцінювання та багатокритеріального вибору архітектури ПС з множини альтернатив зображено на рис. 1.

Тут використовуються такі позначення: $K_j^1, j = \overline{1,p}$ критерії якості ПС, визначені у відповідності з вимогами в термінах стандарту з якості ISO/IEC 25010; $K_i^2, i = \overline{1,n}$ – критерії якості архітектури, визначені з множини $K_j^1, j = \overline{1,m}$ методом SQFD (Software Quality Function Deployment) або методом парних порівнянь [5]; K^0 – інтегральний критерій якості ПС; $R_i, i = \overline{1,n}$ – задані обмеження на показники якості архітектури; $A_i, i = \overline{1,m}$ – альтернативні архітектурні рішення. Оскільки множина критеріїв $\{K_i^2\}$ отримана з множини $\{K_j^1\}$, то можна рівень показників якості ПС виключити з розгляду.

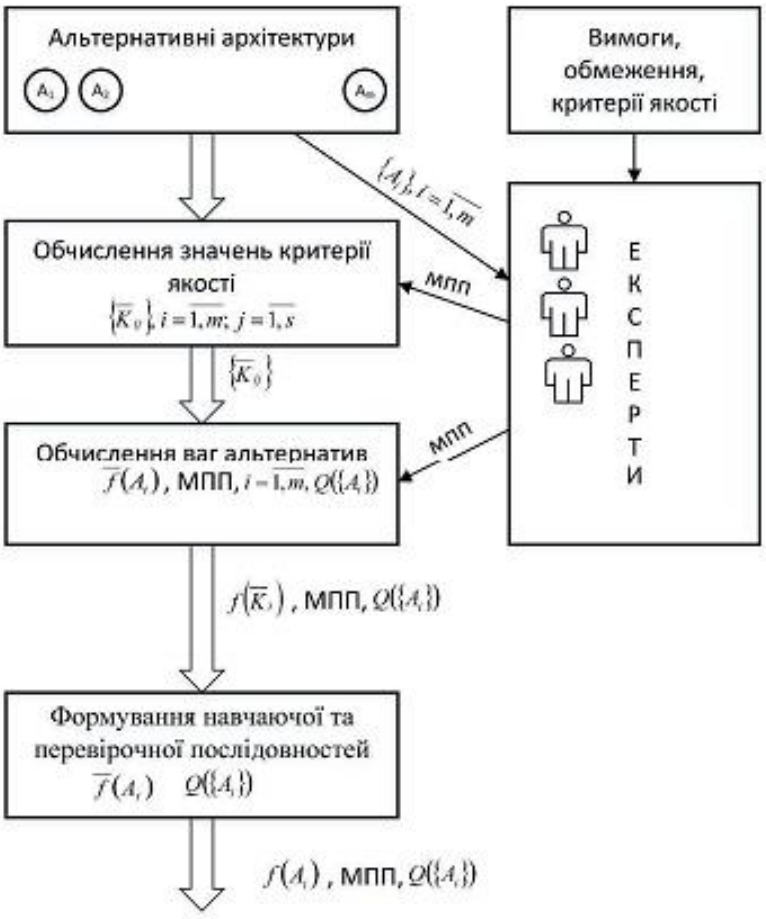

Рисунок 1. Структурна схема процедури вибору програмної архітектури

Отримати порівняльні оцінки альтернатив $\{A_i\}$ по кожному критерію $K_i^2, i = \overline{1,n}$ можна методом аналізу ієрархій Саати (МАІ), або модифікованим методом (ММАІ), використання яких детально описано в роботах [2], [5]. Відмінність ММАІ від МАІ полягає в тому, що в ньому оцінки альтернатив по реалізації критеріїв якості шукаються з умови мінімуму міри неузгодженості матриці парних порівнянь, що дозволило розширити межі застосування методу на більшу кількість альтернатив (критеріїв) ($n \leq 30$) [6].

При обчисленні інтегрального критерія якості альтернатив з використанням скалярної згортки ваги критеріїв визначають експертним методом.

В експертному оцінюванні, як правило, беруть участь декілька груп фахівців, які мають різне бачення стосовно впливу кожного з атрибутів якості на загальну якість ПА. Для підвищення достовірності результуючої оцінки і досягнення компромісу призначаються показники компетентності кожній з груп $(\beta_1, \beta_2, \ldots, \beta_r)$, $\sum_{i=1}^{r} \beta_i = 1$, $\beta_1 \geq 0$.

Потім кожна з груп формує матриці парних порівнянь для критеріїв якості і застосуванням MAI обчислюються ваги критеріїв $\{\alpha_i^s\}$, i=1, n , s=1, r – номер групи експертів. Компромісне рішення можна знаходити, як середнє геометричне $\alpha_i = \sqrt[r]{\alpha_i^1 \cdot \alpha_i^2 \cdots \alpha_i^r}$, або як усереднене, з врахуванням показника компетентності груп експертів $\alpha_i = \alpha_{i1}^{\beta_1} \alpha_{i2}^{\beta_2} \cdots \alpha_{ir}^{\beta_r}$, $i=\overline{1,n}$. Однак, при великих розбіжностях в оцінках, таке усереднення може не привести до компромісу інтересів. Так, як слідує з даних, взятих з [6], отримані за результатами опитування різних груп фахівців, відрізняються більше, ніж в два рази.

В цьому випадку, використання усереднених значень оцінок ваг критеріїв може не забезпечити компроміс інтересів експертів. Тоді використання лінійної згортки для оцінки альтернатив ПА і вибору найкращої з них може бути некоректним. Також при виборі ПА потрібно враховувати можливість зміни вимог до ПА в процесі проектування, а відповідно і ваг критеріїв якості.

Література

1. Len Bass, Paul Clements, Rick Kazman Software architecture in practice. 2nd ed., Addison-Wesley, 2003, 575 p.
2. M. Svahnberg, C. Wholin, and L. Lundberg. A Quality-Driven Decision-Support Method for Identifying Software Architecture Candidates. Int. Journal of Software Engineering and Knowledge Engineering. 2003. 13 (5). P. 547–573.
3. Saaty T., Vargas L. Decision Making with the Analytic Network Process. N.Y.: Springer, 2006. 278 p.
4. Павлов А. А., Лищук Е. И., Кут В. И. Математические модели оптимизации для нахождения весов объектов в методе парных сравнений. Системні дослідження та інформаційні технології. К.: ІПСА, 2007. № 2. С. 13–21.
6. Харченко О. Г., Боднарчук І. О., Галай І. О. Метод багатокритеріальної оптимізації програмної архітектури на основі аналізу компромісів. Інженерія програмного забезпечення. К.: НАУ, 2012. № 3–4 (11–12). С. 5–11.
7. Kharchenko A., Bodnarchuk I., Yatcyshyn V. The method for comparative evaluation of software architecture with accounting of trade-offs. American Journal of Information Systems. Vol. 2. No. 1. 2014. P. 20–25. URL: http://pubs.sciepub.com/ajis/2/1/5.
8. Воронин А. Н., Зиатдинов Ю. К. Теория и практика многокритериальных решений: Модели, методы, реализация. LambertAcademicPublishing, 2013, 305 p.