УДК 004

А.Розиєв, магістр гр. КІ-24М,
Центральноукраїнський національний технічний університет

ДОСЛІДЖЕННЯ ТА ПРИНЦИПИ ПОБУДОВИ СИСТЕМИ БЕЗПЕЧНОЇ РОЗРОБКИ ПРИ ВИКОРИСТАННІ ТЕХНОЛОГІЇ AGILE

У статті розроблено програмне забезпечення, яке призначено для системи безпечної розробки при використанні технології Agile. Метою розробки є дослідження та принципи побудови системи безпечної розробки при використанні технології Agile. Об'єктом дослідження є процес безпечної розробки при використанні технології Agile. Предметом дослідження є методи безпечної розробки при використанні технології Agile. Методи дослідження базуються на методах теорії розробки програмного забезпечення, методах математичної статистики, методах розробки програмного забезпечення. Результат роботи – програмна реалізація системи безпечної розробки при використанні технології Agile. В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

методи безпечної розробки, Agile

Постановка проблеми. Популярність Web-застосунків як основного інструмента онлайн-бізнесу росте рік у рік, одночасно із цим усе більше залучаючи зловмисників. Тим часом, відповідно до досліджень Contrast Security, порядку 80% Web-застосунків у середньому містять до 45 уразливостей, з яких принаймні одна має високий рівень критичності. Така ситуація свідчить про недостатню увагу до питань безпеки в процесі розробки. Із впровадженням методологій прискореної розробки Agile положення ще більше збільшується.

Швидкість розробки збільшується рік у рік. Раніше новий реліз з'являвся один раз у квартал або один раз у рік, зараз же, в умовах зростаючої конкуренції, всі замовники прагнуть якнайшвидше реалізовувати новий функціонал. У зв'язку із цим доводиться скорочувати цикли розробки: релізи випускаються кілька разів у тиждень або навіть у день, так що стандартний підхід, коли про безпеку замислювалися лише в самому кінці, при такій моделі працює дуже погано.

Насамперед якщо не власники бізнесу, то власники застосунків повинні приділяти пріоритетну увагу безпеки й усвідомлювати, як наслідку зломів, компрометацій і витоку даних позначається на прибутку компанії. Крім того, необхідно переглянути роль фахівця з ІБ: з охоронця/наглядача, що заважає всім процесам і вставляє ціпка в колеса, як його часто сприймають, він повинен перетворитися в партнера й ІБ-наставника.

Аналіз останніх досліджень і публікацій. При аналізі останніх досліджень і публікацій [1-10] було виявлено певні прогалини у забезпеченні системи безпечної розробки при використанні технології Agile.

Мета й завдання дослідження. Метою роботи є дослідження та принципи побудови системи безпечної розробки при використанні технології Agile.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

– Огляд існуючих систем безпечної розробки при використанні технології Agile.
– Дослідження системи безпечної розробки при використанні технології Agile.
– Програмна реалізація системи безпечної розробки при використанні технології Agile.

Об'єктом дослідження є процес безпечної розробки при використанні технології Agile.

Предметом дослідження є методи безпечної розробки при використанні технології Agile.

Методи дослідження базуються на методах теорії розробки програмного забезпечення, методах математичної статистики, методах розробки програмного забезпечення.

Виклад основного матеріалу. Agile – це гнучка методологія розробки. яка являє собою концепцію, у рамках якої виконується розробка програмного забезпечення. У рамках даної концепції існує кілька методик, метою яких є мінімізація ризиків, досягається ця мета розробкою (проектуванням) короткими ітераціями.

Основні цінності Agile утримуються в "Маніфесті Agile":

- Люди і їхня взаємодія важливіше, ніж процеси й засоби.
- Працююча система важливіше, ніж вичерпна документація.
- Співробітництво із замовником важливіше, ніж обговорення умов контракту.
- Реагування на зміни важливіше, ніж проходження плану.

Scrum – це конкретна технологія (методи ведення проекту й ролі учасників процесу), що реалізує принципи Agile.

Дана технологія дозволяє в жорстко фіксовані й невеликі за часом ітерації, називані **спринтами (sprints)**, надавати замовникові працююче ПЗ з новими можливостями, для яких визначений найбільший пріоритет.

Вимоги до результатів спринту визначаються на початку спринту (планування спринту) і не можуть змінюватися в процесі спринту. Невелика тривалість спринту надає процесу передбачуваність і гнучкість.

Ролі Scrum

1. **Scrum Master.** Бізнес-**а**налітик, керівник проекту. Проводить наради, стежить за дотриманням технології Scrum, знімає протиріччя й направляє команду. Основний обов'язок – забезпечення виконання технології Scrum.

2. **Product Owner.** Власник проекту, функціональний замовник. Представляє інтереси замовника (кінцевих користувачів).

3. **Development Team.** Команда фахівців (розроблювачів). Складається з фахівців різного профілю- аналітиків, архітекторів, програмістів, тестувальників. Команда відповідає за результат як єдине ціле.

Елементи Scrum

Sprints (Спринт)

Спринт – ітерація в процесі, у ході якої створюється новий результат. Спринт жорстко фіксований у часі – від 1 до 4 тижнів. Ніж коротше спринт, тим гнучкішим є процес розробки, оскільки після кожного спринту вимоги до системи можуть коректуватися на підставі зворотного зв'язка від замовника. Відповідно, знижуються ризики роботи в неправильному напрямку. З іншої сторони при більш тривалому спринті знижуються витрати на наради, і більше залишається часу на рішення завдань проекту.

Project backlog (журнал завдань проекту)

Project backlog – журнал побажань проекту. Це список вимог до системи, упорядкований за пріорітетом – важливості реалізації. Журнал побажань можуть доповнювати всі учасники процесу.

Sprint backlog (журнал завдань спринту)

Sprint backlog – журнал побажань спринту. Містить функціональність, відібрану власником проекту (**Product Owner**) для реалізації на поточному спринті.

Burndown chart (Діаграма згоряння завдань)

Burndown chart – діаграма згоряння завдань. Демонструє обсяг зробленої й роботи, що залишилася, щодо строку проекту. Діаграма актуалізується щодня. Передбачено два види діаграм:

– Діаграма згоряння для спринту – показує, скільки вже завдань зроблене й скільки ще залишається зробити в поточному спринті.

– Діаграма згоряння для проекту – показує, скільки вже завдань зроблене й скільки ще залишається зробити до завершення проекту.

Abnormal Termination (Зупинка спринту)

Abnormal Termination – зупинка спринту. Спринт може бути зупинений раніше його планового строку закінчення у виняткових ситуаціях. Наприклад, якщо завдання спринту не можуть бути досягнуті або якщо вони стали неактуальними. Рішення про зупинку приймається командою або Власником проекту. Після зупинки починається новий спринт.

Daily Scrum meeting (Щоденна нарада)

Daily Scrum meeting – щоденна нарада команди. Правила проведення наради:

– проводиться в те саме час, у тому самому місці;
– не більше 15 хвилин;
– кожному учасникові треба відповісти на 3 питання:
– Що я зробив учора
– Що я планую зробити сьогодні
– Що мені заважає

Scrum of Scrums (Скрам над скрамом)

Scrum of Scrums – Скрам над скрамом – нарада декількох Scrum-команд. Проводиться після щоденної скрам наради. Дозволяє декільком скрам командам обговорювати роботу, фокусуючись на загальних областях і взаємній інтеграції. Повістка та ж, що й на щоденному скрам нараді плюс наступні питання:

– Що кожна команда зробила з моменту попередньої щоденної наради?
– Що кожна команда зробить до наступної щоденної наради?
– Є чи проблеми, що заважають або сповільнюють роботу кожної команди?
– Потрібно чи іншій команді зробити щось із завдань вашої команди?

Sprint review meeting (огляд підсумків спринту)

Sprint review meeting – огляд підсумків спринту. Проводиться наприкінці спринту:

– Команда демонструє результати спринту всім зацікавленим особам. Залучається максимальна кількість глядачів.

– Всі члени команди беруть участь у демонстрації (одна людина на демонстрацію або кожного показує, що зробив за спринт).

– Не можна демонструвати незавершену функціональність.

– Обмежена чотирма годинами залежно від тривалості ітерації й обсягу результату.

У цей час про гнучкі технології виконання проектів Agile/Scrum говорять всі частіше. У цій статті я напишу про своє бачення плюсів і мінусів застосування Scrum при виконанні проектів з позиції замовника.

Agile, Scrum, Kanban – ці слова вже встигли примелькатися всім, хто має яке-небудь відношення до розробки продуктів, бізнес-процесам, організації праці. Багато хто вважають, що за ціми технологіями майбутнє. Але, що цікаво, деякі можуть пояснити розходження між ціми поняттями, наприклад, відповістити на запитання про Scrum і Agile: різниця існує й у чому вона полягає?».

Отже, спершу визначимося з методологічним апаратом:

Agile – клас гнучких методологій розробки; набір цінностей і базових принципів в області розробки ПЗ, основна суть яких зводиться до взаємодії функціональних крос-команд що самоорганізуються. У суть Agile закладені адаптивне планування, еволюційна розробка, максимально швидкий випуск нових версій продукту, постійна доробка й швидка відповідь на виникаючі складності. На сьогоднішній день Agile уже давно вийшов за рамки розробки програмного забезпечення, а його різні дочірні методології використовуються в різних сферах керування.

Scrum – одна з методологій у рамках Agile. В основі даної методології лежать «спринти» – короткі жорстко фіксовані за часом ітерації, у рамках яких як виконавець так і замовник виконуються певний набір завдань.

Kanban – система, що передбачає організацію розробки/ праці/ виробництва/ постачання відповідно до принципу Just In Time (або точно в строк). В основі Kanban лежить виробнича й постачальницька система Toyota, що у другій половині XX століття ледве було не знищила весь американський автопром.

Основними цінностями в концепції **Scrum** оголошені комунікація й готовність до змін, а не тверде проходження первісному плану. Коли ж **Scrum** може принести тільки користь, а коли – шкода?

Тверда підрядна технологія

Суть підряду дуже проста. Замовник замовляє й чітко визначає в специфікації до договору, що саме він хоче одержати**.** У договорі чітко фіксується вартість і строк. Замовник передає виконавцеві матеріали, придатні для обробки. Виконавець перевіряє матеріали, і якщо вони непридатні, то негайно сповіщає про це Замовникові й зупиняє роботи. Замовник зобов'язаний робити Виконавцеві сприяння в процесі робіт. Якщо Виконавець не уклався в строк, або надав настільки неякісний результат, що можна припустити недосягнення результату як такого – Замовник має право розірвати договір і зажадати повернення авансу в повному розмірі. Якщо ж результат досягнуть, то Замовник приймає роботи, підписує акт виконаних робіт, оплачує роботу.

При виконанні проектів по впровадженню ERP-систем звичайно полягають саме такі договори. У договорі визначаються фіксований бюджет і строк на кожний етап робіт, саме завдання (безпосередньо в договорі або окремо в ТЗ), обов'язку Замовника на кожному етапі, однозначні критерії результату. Зміни у вимогах у процесі робіт або не допускаються зовсім, або додатковий обсяг робіт оплачується додатково за окремою згодою.

Підкреслимо ключове положення цієї моделі взаємодії: у результаті оцінки Виконавцем поставленої споконвічно завдання в договорі встановлюється твердий строк і вартість етапу. Тому зміна завдання в процесі виконання робіт не допускається. Це правило діє, наприклад, і при ремонті квартири, і при впровадженні ERP-систем.

Якщо використовується ітераційний підхід, то після закінчення етапу робіт стають ясними завдання наступного етапу, а виходить, що випливають обсяги робіт. Відповідно, у додатковій угоді фіксується строк і вартість на наступний етап. Буває ще цікавіше – при впровадженні ERP-систем Виконавець (найчастіше щоб виграти тендер) фіксує бюджет на весь проект цілком, іноді на 1-2 роки вперед.

Замовники приймають бюджет, думаючи, що й строки, і бюджети будуть дотримуватися. При цьому, відповідно до договору, етапи робіт у кожному разі закриваються актами й оплачуються окремо, чи ледве не щомісячно. Не виключено, що через кілька місяців з'ясується, що бюджет необхідно збільшити в 2-3 рази (мов, вимоги до системи змінилися, а «ми про це не домовлялися, у ТЗ цього немає!»). Не хочете – до побачення.

Основний недолік твердої технології – істотні обмеження гнучкості у вимогах і рішеннях, в умовах виконання робіт, обмеження гнучкості участі Замовника в процесі виконання етапу. Така гнучкість у ході виконання робіт украй важлива, тому що на початку кожного етапу детальні, вичерпні, точні вимоги й організаційні умови сформулювати важко – високий ступінь невизначеності системи. Інтегратори-виконавці часто говорять: «Клієнт сам не знає, чого хоче»… Зовсім тупикові ситуації виникають, коли клієнт – який не знає, чого хоче – вимагає зробити систему «під ключ».

Оскільки строки й бюджети визначаються на початку кожного етапу, будь-які коректування вимог до системи з боку Замовника, а також будь-які зміни умов у процесі виконання робіт на етапі (наприклад, обсяг участі й сприяння з боку Замовника) можуть створити додаткову трудомісткість для Виконавця, що зажадає перегляду бюджету й строків. Але перегляд бюджетів і строків – потенційно конфліктна ситуація!

Отже, Замовник зацікавлений у можливості зміни вимог у процесі виконання робіт, але підрядна технологія не дає йому цього робити, викликаючи конфлікт із Виконавцем.

Для підвищення гнучкості – у бюджет і строки заставляються ризики. Крім того, досвідчені Виконавці передбачають і прописують процедуру «Керування змінами», відповідно до якої в остаточному підсумку всі зміни первісних вимог проходять через процедури ініціювання й оцінки.

Вартість ризиків Виконавця, відбитих у бюджеті проекту, у подібних роботах може досягати більше 50% від вартості етапу. Інакше кажучи, Виконавець продає замовникові не тільки результат робіт, але й свої ризики. Якщо ризики не реалізуються, вартість робіт для Замовника виявляється завищеною щодо реальної трудомісткості цих робіт, а Виконавець дістає додатковий прибуток. Буває й навпаки – реалізовуються незаплановані ризики понад бюджет, у результаті чого Виконавець виявляється в збитку.

Таким чином, підрядна технологія обмежує гнучкість Замовника в зміні вимог, умов роботи й ступеня своєї участі, коли роботи вже початі, і «корабель відійшов від берега». Необхідна процедура керування змінами... але немає гарантії, що ця процедура зможе передбачити всі ситуації.

Більші витрати сторони несуть на формулюванні всіх умов робіт і вимог до результату при початку етапу, а також на формалізації правил керування змінами. І це при тім, що умови й вимоги можуть швидко втрачати актуальність у процесі виконання етапу! У результаті частина розробленого й оплаченого функціонала може виявитися незатребуваною в процесі експлуатації системи – нерідка ситуація при підрядному способі.

Чому ж Замовники, як правило, вимагають, щоб впровадження складних систем виконувалося по договорах підряду? Відповідь проста:

– Інші способи поділу відповідальності для Замовника менш комфортні. У договорі Замовник замовляє, а Виконавець – виконує, ця логіка породжує поділ відповідальності сторін за проект, зручне для Замовника.

– Інші (не підрядні) технології вимагають більшої відповідальності з боку замовника.

Чому Виконавці охоче погоджуються на підряд?

– Щоб забезпечити гнучкість у вимогах, Виконавці закладають у фіксовану вартість проекту ризики. Якщо ризики не реалізуються, Замовник переплачує.

– Вимоги до результату погодяться «на березі», а виходить, Замовник знову втрачає гнучкості, а Виконавець одержує можливість не робити потрібний Замовникові функціонал, посилаючись на те, що «у ТЗ цього немає». Так працювати спокійніше: зафіксував вимоги й “уперед”.

Гнучкість вимог у процесі виконання робіт

Через високу невизначеність системи на початку проекту й неможливості передбачити всі, замовники прагнуть міняти вимоги в процесі виконання робіт. З'являються нові ідеї, відбуваються зміни на підприємстві, з'ясовується, що замість інтеграції зі старою системою краще доробити нову систему й т.д.

Гнучкість вимог життєво необхідна для досягнення результату, дійсно потрібного підприємству.

Гнучка технологія (Scrum)

Зміст концепції Scrum у тому, що кожний етап робіт розбивають на короткі ітерації фіксованої тривалості (від 1-й тижня до місяця). На початку кожної ітерації визначають завдання сторін – Замовника й Виконавця – тільки на дану ітерацію. У процесі ітерації (у термінології Scrum – «**Спринт**») зміни в завданнях Сторін, вимогах і умовах виконання робіт не допускаються. По завершенні ітерації Сторони аналізують результати й ставлять завдання на наступну ітерацію; і при цьому вужі **допускається змінювати умови** виконання робіт.

Дуже важливо щільна й безперервна взаємодія Виконавця й Замовника. Якнайбільше спілкуватися, бути однією командою – от ключовий момент даного підходу!

Перевага гнучкої технології:

– За рахунок того, що ітерації набагато коротше, ніж весь етап робіт, досягається набагато більша гнучкість у прийнятті рішень і керуванні змінами в ході робіт. Між ітераціями Замовник не обмежений у прийнятті організаційних рішень, зміні рамок проекту й так далі. Очевидно, що сторони рухаються до результату більше коротким шляхом, чим при підряді, оскільки можуть безупинно вносити зміни в проект.

– Виключаються конфліктні ситуації через зміни, тому що в центр такої технології поставлена можливість зміни. Якщо при підряді зміна – це неприємне відхилення, що вимагає розбирань і з'ясування, хто кому повинен, то в Scrum зміни – це суть і складова процесу.

– Тарифікація ітерацій теж може бути гнучкої – або оплата виробляється по фактичній трудомісткості (відповідно до погодинної ставки роботи співробітника); або вартість ітерації фіксується на початку ітерації, виходячи із прогнозу її трудомісткості. У результаті, Замовник оплачує тільки фактичну трудомісткість і не переплачує за ризики.

Прогноз вартості кожної ітерації виконується досить просто по погодинному тарифі (якщо відомо кількість співробітників Виконавця, планованих на ітерацію з повним завантаженням).

Таким чином, при використанні гнучких технологій бюджет проекту на певний період будується на підставі прогнозу трудомісткості й кількості приваблюваних співробітників Виконавця.

Як правило, при використанні гнучких технологій вартість проекту не перевищує його вартості при використанні твердих технологій, оскільки кінцева вартість прямо залежить від сумарної трудомісткості. Трудомісткість гнучкої технології може виявитися навіть менше за рахунок того, що не розробляється незатребуваний функціонал, а виходить, менше буде й вартість проекту в цілому.

Однак технології Scrum властиві свої недоліки. Один з них – ризик несистемного підходу. Визначення всіх вимог до системи до початку робіт дозволяє продумати систему в цілому ще до її реалізації. У ході реалізації частини системи враховуються властивості всієї системи, всі вимоги до неї, а не тільки до одному з її сегментів. У цьому й полягає системний підхід.

При несистемному підході висока ймовірність, що замість системи вийде набір незв'язаних елементів – зовсім як у древній східній казці, у якій мудреці за хвостом, хоботом і ногами не побачили слона:

Тому якщо їсти можливість ретельно продумати функціонал і вимоги до початку реалізації, їй треба скористатися. Зрозуміло, якщо є вся необхідна інформація. Це ідеальний випадок, і таке буває рідко, тому найкращим варіантом у таких ситуаціях є Scrum, що дозволяє формувати вимоги до системи в процесі її реалізації й експлуатації. Якщо ж зміни не передбачаються, то Scrum – не кращий вибір.

Наприклад, при розробці типових рішень (не під конкретного замовника з його неочевидними «хотелками») систему краще продумувати відразу в повному обсязі, цілісно. Оскільки зовнішнього непередбаченого замовника немає – у цьому випадку замовником, по суті, є сама команда розроблювачів – вимоги до системи відомі споконвічно. Звичайно, якщо розроблювачі знають, за що беруться.

Отже, керування змінами при гнучкій технології Scrum відбувається найбільше ефективно. Варто врахувати, що в проектах з високим ступенем невизначеності системи на початку робіт керування змінами є ключовим чинником, що впливає як на кінцевий результат, так і на оптимізацію бюджету. У тому числі значно знижуються ризики розробки непотрібного функціонала, або функціонала, що не відповідає потребам бізнесу.

Договір підрядний, а працюємо по Scrum

Переваги Scrum всі частіше можна спостерігати, коли реалізується один із класичних проектних ризиків – "звалювання" проекту в Scrum по факту, хоча за договором робота виконується по підряду – із твердими строками й вартістю. Таке "звалювання" дуже вигідно активному Замовникові, тому що дозволяє досягти бажаного результату, відштовхуючись

щораз від фактичного результату проміжного етапу робіт. Замовникові набагато простіше зрозуміти, що він хоче одержати насправді, якщо він відштовхується від того, що вже зроблено.

Така ситуація характерна не тільки для IT-проектів. При ремонті квартири, будівництві будинку, і будь-якому проекті, коли складно чітко формалізувати остаточний результат, Замовник може почати контролювати безупинно кожний дрібний етап і вносити коректування в споконвічну постановку завдання. Це приводить Виконавця в замішання, тому що строки й трудомісткість роботи неконтрольована ростуть, а бюджет проекту не міняється. Виконавець починає думати, що «Замовник не знає, що хоче», це потенційно конфліктна ситуація. Насправді, Замовник і не може знати, що хоче – це цілком нормально. Що він хоче, він починає розуміти тільки після того як побачить, що виходить по факті – тільки після чергової ітерації робіт. Це Scrum.

На IT проекті "звалювання" проявляється в такий спосіб.

Як правило це відбувається на етапі реалізації (кодування). Як тільки Замовник просить показувати проміжні результати, і починає їх приймати, не чекаючи пред'явлення всього обсягу робіт на випробування, чекайте перехід по факті в Scrum. Приймання Замовником уже першої ітерації робіт швидше за все спричинить частковий перегляд Замовником усього технічного завдання, а це вже Scrum. Зрозуміло, для кінцевого результату й задоволення Замовника це благо. А перегляд ТЗ – це зміна трудомісткості, звичайно в більшу сторону. І якщо не укласти додаткову угоду на збільшення вартості, то ущемляються інтереси Виконавця. А це конфліктна ситуація.

Виникає питання – чи не простіше споконвічно домовитися працювати по Scrum? Адже справедлива оплата робіт повинна відповідати фактичній трудомісткості, що при "звалюванні" в Scrum не піддається точному розрахунку!

Що ж виходить? Замовник може заперечувати Scrum при укладанні договору, але потім фактично змусити Виконавця працювати по Scrum. У рамках фіксованого строку й бюджету. І не забуваємо, що прострочення строку здачі робіт (через постійне внесення Замовником дрібних коректувань у завдання) загрожує Виконавцеві серйозною відповідальністю, тому що Виконавець відповідає за строк здачі робіт. Невиконання строку надає право Замовникові висувати обґрунтовані претензії. Більше того, Замовник може дорікати Виконавцю в недостатньому професіоналізмі, тому що Замовник схильний часто свої коректування в постановку Завдання розцінювати як свої вимоги виправити недоліки.

На рисунку 1 зображена структурна схема програмного забезпечення, яке реалізує процес безпечної розробки при використанні технології Agile з застосуванням обфускації коду. Розглянемо цю схему більш детально.

Процес безпечної розробки при використанні технології Agile з застосуванням обфускації може бути здійснений над кожним з вище перерахованих видів подання програмного коду, тому прийнято виділяти наступні рівні процесу безпечної розробки при використанні технології Agile з застосуванням обфускації:

– нижчий рівень, коли процес безпечної розробки при використанні технології Agile з застосуванням обфускації здійснюється над ассемблерним кодом програми, або навіть безпосередньо над двійковим файлом програми, що зберігає машинний код;

– вищий рівень, коли процес безпечної розробки при використанні технології Agile з застосуванням обфускації здійснюється над вихідним кодом програми написаному мовою високого рівня.

Здійснення безпечної розробки при використанні технології Agile з застосуванням обфускації на нижчому рівні вважається менш комплексним процесом, але при цьому більш важко реалізованим з ряду причин. Одна із цих причин полягає в тому, що повинні бути враховані особливості роботи більшості процесорів, так як спосіб безпечної розробки при використанні технології Agile з застосуванням обфускації, прийнятний на одній архітектурі, може виявитися неприйнятним на іншій.

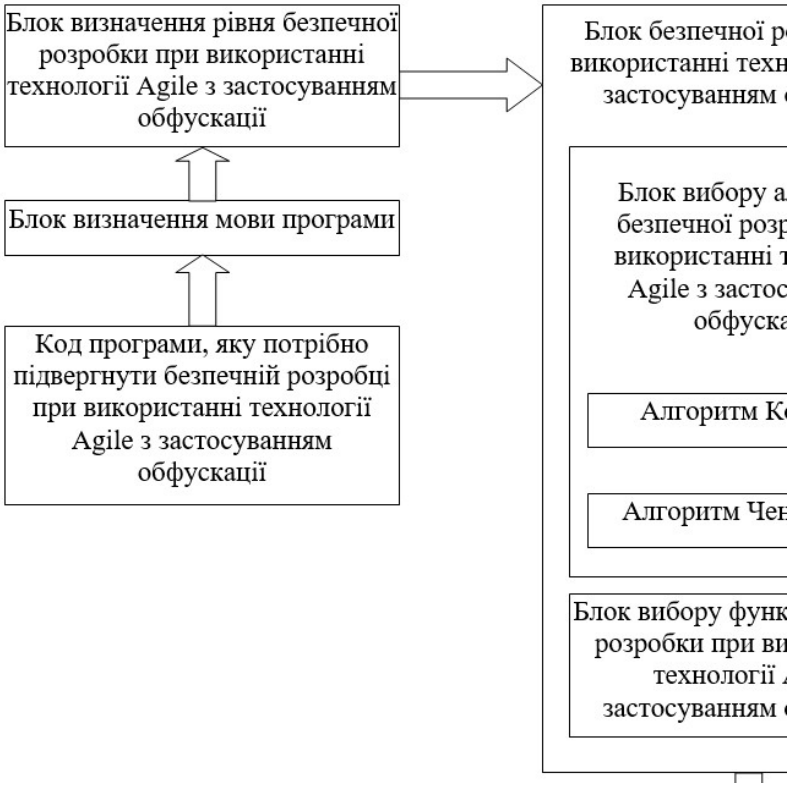

Рисунок 1 – Структурна схема системи

Висновки. У статті наведені теоретичне узагальнення й рішення наукового завдання дослідження методів безпечної розробки при використанні технології Agile. Рішення даного завдання полягало у вирішенні наступних задач:

– Був проведений огляд існуючих систем безпечної розробки при використанні технології Agile.

– Досліджена система безпечної розробки при використанні технології Agile.

– На основі отриманих результатів досліджень створена програмна реалізація системи безпечної розробки при використанні технології Agile.

Розроблені алгоритми дозволяють успішно вирішувати завдання безпечної розробки при використанні технології Agile. Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Список літератури

1. Смірнова, Т.В. «Дослідження методів, моделей та сучасних ІТ-рішень для підтримки технологічних процесів у критичній інфраструктурі держави». Кібербезпека: освіта, наука, техніка. 2025. Том 2 № 30. С.195-208, 2025.
2. Вінтенко Б., Смірнов О., Миронець І., Смірнова Т., Смірнов С. «Імітаційна модель шляхів вхідних даних комп’ютерної інтелектуальної системи підтримки оператора енергоблоку АЕС». Комбінаторні конфігурації та їхні застосування: Матеріали XXVII Міжнародного науково-практичного семінару, присвяченого 125-річчю Національного університету «Запорізька політехніка» (Запоріжжя-Кропивницький-Київ, 4-6 червня 2025 р.). Запоріжжя: НУ «Запорізька політехніка», 2025. С.82-91.
3. Al-Azzeh, J., Ayyoub, B., Mesleh, A., Smirnova, T., Gnatyuk, S., Drieiev, O., Smirnov, O., Dorenskyi, O. «Cloud-Based Information System for Evaluating Caverns in the Process of Blasting Metal Surfaces of Details». International Review on Modelling and Simulations 18 (1), 2025. pp. 32-42.
4. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В., Коваленко О.В., Мацуй А.М. «Модель шляхів отримання вхідних даних комп’ютерної інтелектуальної системи підтримки оперативного персоналу АЕС». Центральноукраїнський науковий вісник. Технічні науки. 2025. Вип. 11(42), ч. II. С.52-62.
5. Вінтенко Б.Ю., Смірнов О.А., Миронець І.В., Смірнова Т.В. «Методи забезпечення відмовостійкості

інтелектуальних систем підтримки оператора». VIII міжнародна науково-практична конференція “Інформаційна безпека та комп’ютерні технології”, м. Кропивницький. 24-25 квітня 2025 р. – Кропивницький: ЦНТУ. – 2025. – С. 44-46.

6. Смірнов, О.А., Константинова, Л.В., Конопліцька-Слободенюк, О.К., Козірова, Н.В, Якименко, Н.М., Доренський, О.П., Буравченко, К.О. «Дослідження інструментів штучного інтелекту для роботи з базами даних та аналізу даних». Кібербезпека: освіта, наука, техніка. 2025. №3(27), С. 429–448.)
7. Smirnov O., Fedorov E., Neskorodieva A., Neskorodieva T. «Intellectual Classification method of Gymnastic Elements Based on Combinations of Descriptive and Generative Approache». CEUR Workshop Proceedings Volume 3664, 2024, Pages 11-23.
8. Вінтенко, Б., Миронець, І., Смірнов, О., Коваленко, А., Конопліцька-Слободенюк, О., Смірнова, Т., Константинова, Л. «Дослідження застосування систем підтримки оперативного персоналу об’єкту критичної інфраструктури при керуванні енергоблоком АЕС з реактором типу ВВЕР-1000». Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка», 2024. № 2(26), С. 6-26.
9. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Кібербезпека: освіта, наука, техніка. 2024. №3(23), С. 111-131.
10. Kuznetsov O., Ilchenko O., Kryvinska N., Buravchenko K., Smirnov O., Savchenko Iu. «An Empirical Assessment of Leading Blockchain Financial Services». 2023 IEEE 1st Ukrainian Distributed Ledger Technology Forum (UADLTF), Kyiv, Ukraine, 2023, pp. 1-6,
11. Kuznetsov, O., Kryvinska, N., Ilchenko, O., Smirnova, T., Ulianovska, Y. «Comparative Analysis of Cryptocurrency Trading Platforms Using the Analytic Hierarchy Process». CEUR Workshop Proceedings, 2023, 3628, pp. 106-115.
12. Malyukov V., Bebeshko B., Lakhno V., Smirnov O., Malyukova I., Mohylnyi H. «Managing the Purchase-Sale Process of Digital Currencies Under Fuzzy Conditions». Lecture Notes in Networks and Systems, 2023, 729 LNNS, pp. 104–112.
13. Al-Mudhafar Aqeel, A.M., Smirnova, T., Buravchenko, K., Smirnov, O. «The method of assessing and improving the user experience of subscribers in software-configured networks based on the use of machine learning». Advanced Information Systems, 2023, 7(2), pp. 49-56.
14. Smirnov, O., Sydorenko, V., Aleksander, M., Zhyharevych, O., Yenchev, S. «Simulation of the cloud IoT-based monitoring system for critical infrastructures». CEUR Workshop Proceedings, Volume 3530, 2023, pp. 256-265.
15. Smirnov, O., Odarchenko, R., Smirnova, T., Bondar, S., Volosheniuk, D. «Optimal Structure Construction of Private 5G Network for the Needs of Enterprises». Lecture Notes on Data Engineering and Communications Technologies, 2023, 178, pp. 208–223.
16. Вінтенко Б.Ю., Смірнов О.А., Коваленко А.С., Смірнов С.А., Буравченко К.О. «Дослідження вимог міжнародних стандартів IEC60880 та IEC62138 з розробки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Системи управління, навігації та зв’язку, 2023, вип. 3(73), С. 155-166.
17. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Кібербезпека: освіта, наука, техніка. 2024. №3(23), С. 111-131.
18. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп’ютерних систем управління АЕС, важливих для безпеки». Системи управління, навігації та зв’язку, 2023, вип. 2(72), С. 170-178.
19. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». Сучасні інформаційні системи, 2023, том 7, № 2, С. 49-56.
20. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А. «Дослідження нормативної документації та стандартів розробки програмного забезпечення комп’ютерних систем управління АЕС, важливих для безпеки». VI міжнародна науково-практична конференція “Інформаційна безпека та комп’ютерні технології”, м. Кропивницький. 20-21 квітня 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 35-36.
21. Smirnov, O., Karapetyan, A., Fedorov, E., «Creating Neural Network and Single Solution Human-Based Metaheuristic Methods of Solving the Traveling Salesman Problem». CEUR Workshop Proceedings, Volume 3312, 2022, pp. 47-58.
22. Smirnov O., Kuznetsov A., Kryvinska N., Kiian A., Kuznetsova K. «Full Non-Binary Constant-Weight Codes». SN Computer Science, Vol 2, 337, 2021. https://doi.org/10.1007/s42979-021-00739-w.
23. Smirnov O., Kovalenko O., Kovalenko A., Kavun S. «Quantitative Risk Assessment Method Development in the Context of the SDLC-model». 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T), 2021, pp. 203-208, doi: 10.1109/PICST54195.2021.9772143
24. Smirnova T., Gnatyuk S., Berdibayev R., Avkurova Zh., Iavich M. «Cloud-Based Cyber Incidents Response System and Software Tools». Communications in Computer and Information Science, 2021, vol 1486. Springer, Cham. pp 169-184.
25. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering

functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.
26. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114.
27. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.
28. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». International Journal of Computer Network and Information Security (IJCNIS). Vol. 12, No. 3, 2020. PP.33-43.
29. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645.
30. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019; Odessa; Ukraine; 9-13 September 2019. P.22-28.
31. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.