УДК 004.896
Самойленко Е.С., Проніна О.І.
Державний вищий навчальний заклад «Приазовський державний технічний університет»

МОДЕЛЮВАННЯ НЕЧІТКОГО ВИВЕДЕННЯ ДЛЯ ПЕРЕВІРКИ ЯКОСТІ ПРОГРАМНОГО ПРОДУКТУ

Самойленко Е.С., Проніна О.І. Моделювання нечіткого виведення для перевірки якості програмного продукту. У статті описані вимоги до перевірки якості програмного продукту перед передачею його в роботу. Представлені етапи побудови системи нечіткого виведення для визначення якості розроблюваного програмного продукту. Описана модель, вихідна лінгвістична змінна необхідна для побудови системи нечіткого виведення. Сформовано продукційні правила для системи нечіткого виведення. Проведено моделювання розробленої нечіткої моделі, показані результати моделювання нечіткого виведення в середовищі MatLabFuzzy.
Ключові слова: якість програмного продукту, тестування програмного продукту, нечітке моделювання, продукційні правила, система нечіткого виведення.

Самойленко Э.С., Пронина О.И. Моделирование нечеткого вывода для проверки качества программного продукта. В статье описаны требования к проверке качества программного продукта перед передачей его в работу. Представлены этапы построения системы нечеткого вывода для определения качества разрабатываемого программного продукта. Описана модель, выходная лингвистическая переменная необходимая для построения системы нечеткого вывода. Сформированы продукционные правила для системы нечеткого вывода. Проведено моделирование разработанной нечеткой модели, показаны результаты моделирования нечеткого вывода в среде MatLab Fuzzy.
Ключевые слова: качество программного продукта, тестирование программного продукта, нечеткое моделирование, продукционные правила, система нечеткого вывода.

Samoilenko E.S., Pronina O.I. Modeling fuzzy inference for checking the quality of a software product. The article describes the requirements for checking the quality of a software product before transferring it to work. The stages of building a fuzzy inference system for determining the quality of a software product being developed are presented. A model is described, the output linguistic variable necessary for constructing a fuzzy inference system. Production rules for the fuzzy inference system are generated. The developed fuzzy model is simulated, the results of fuzzy inference modeling in the MatLab Fuzzy environment are shown.
Keywords: soft ware product quality, software testing, fuzzy modeling, production rules, system of fuzzy inference.

Вступ. Процес розвитку сучасного інформаційного простору тягне за собою зростаючи потреби в системах автоматизації та інтелектуалізації. З причини стрімкого росту вимог до якості програмного продукту, виникають нові умови і вимоги до оптимізації та автоматизації процесу тестування програмних забезпечень. У нинішніх тенденціях розробки програмного забезпечення одна з найбільш популярних і виправданих моделей життєвого циклу є ітераційна. В її основу закладено послідовне виконання кожного етапу, один за іншим, від постановки завдання до експлуатації програмного забезпечення (рис. 1).

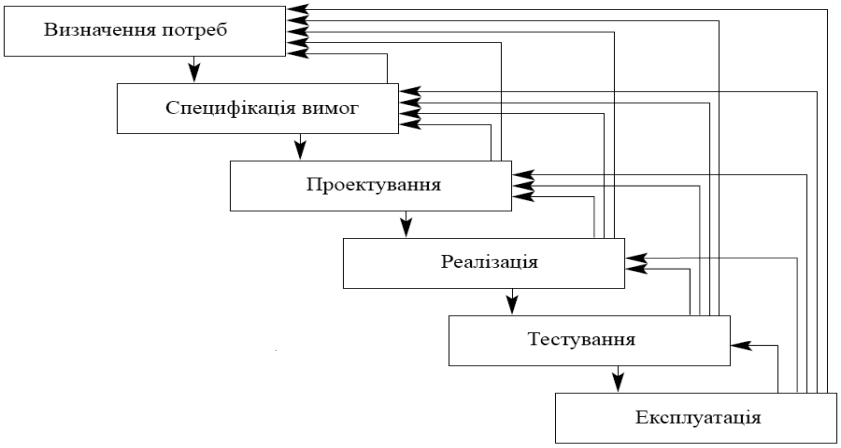


Рис. 1. Ітераційна модель життєвого циклу

Перевага такого підходу – послідовне виконання кожного етапу з фіксуванням кінцевого результату і оцінкою якості виконаної роботи, також, зниження ризиків при плануванні системи. Але, серед вагомих недоліків, порушення цілісності розробки при зміні пройденого етапу в рамках однієї ітерації, що призводить до зміщення поточного етапу, часу виконання циклу і додаткових трудовитрат.

Щоб компенсувати втрати часу і трудовитрат, в реальних проектах, можуть внести корективи в якість продукту, що розробляється, створюючи свідомо погану логіку додатка в етапі реалізації з виправленням «на майбутнє» або скоротивши обсяг робіт на етапі тестування і налагодження за рахунок мінімізації процесу або пропустивши цей крок зовсім, що в корені неприйнятно. Процес тестування дозволяє з'ясувати відповідність між поставленим завданням і її реалізованим видом, а також зменшити число дефектів і поліпшити якість програмного забезпечення [1].

Тестування розділяється на велику кількість видів в різних напрямках, наприклад, за програмними цілями, з аналізу системи, по формальності та інше. В рамках розробки програмного забезпечення, використовується в повному обсязі види, основний, за ознакою поширеності, є тестування за метою, що включає в себе функціональне тестування (придатність, точність, взаємодія) і не функціональне (зручність, відмова і відновлення, конфігурація). На відміну від функціонального тестування, де виконується перевірка відповідності продукту, не функціональне розглядає продукт з точки зору виконуваного середовища і оточення.

Набір властивостей функціонального тестування формує якість програмного продукту і його готовність до етапу експлуатації. Відповідно до міжнародного стандарту [2], набір властивостей поділяється на:

1) Придатність (Suitability) - здатність програмного забезпечення до виконання своїх функціональних цілей.
2) Точність (Accuracy) – близькість дійсного результату до очікувану.
3) Здатність до взаємодії (Interoperability) – здатність програмного продукту виконувати набір функцій, визначених у його описі і відповідають указаним потребам користувачів.
4) Захищеність (Security) – стабільність роботи програмного забезпечення, прогнозування можливих помилок, дефектів і багів, відмов і помилкових ситуацій.
5) Відповідність (Functionality compliance) - здатність програмного забезпечення дотримуватися стандартів, угод або нормативних актів в законах і аналогічних приписах, що стосуються функціональності.

У [3], були внесені зміни в поточний набір функцій атрибутів і прибрав пункт Захищеність, що є неоднозначним рішенням, так як стабільність роботи програмного забезпечення залежить від багатьох чинників і має оцінюватися як окремий набір.

Проблема процесу тестування в сучасній розробці не нова і найбільш часто залежить від комерційного фактору, коли процес побудований на жорстких рамках виділеного часу на кожен етап ітераційної розробки. В рамках даної роботи, тестування розглядається не з економічного боку, а з боку процесу виконання, впливу чинників на кінцеву якість програмного забезпечення. З вище сказаного випливає, що проблема є актуальною і важливою в практичному застосуванні, цінність тестування полягає у виявленні проблем, що загрожують якості продукту. Більш докладний розбір даного етапу може бути корисний для оптимізації і зменшенні часу на виконання тестів, порівняльного аналізу

Метою даної роботи є моделювання нечіткої системи виведення для перевірки якості програмного продукту, що розробляється на основі виявлених основних критеріїв при тестуванні, а саме функціональна придатність, точність, здатність до взаємодії, відповідність, захищеність.

Аналіз останніх досліджень і публікацій. Згідно з дослідженнями Майкла Болтона (Michael Bolton), дослідник і засновник компанії Develop Sense [4], основна проблема тестування полягає в аналізі кінцевих результатом тестування, в їх обробці для прийняття рішень про готовність продукту, при цьому сама послідовність дій для їх отримання не важлива. В роботі [5] автор зазначає, що у процесі розробки програмного забезпечення його якість і надійність можуть змінюватися та аналізувати ці дані необхідно за кількома аналітиками та тестування в даному випадку виступає як техніка контролю якості, що перевіряє відповідність між реальною та очікуваною поведінкою, саме техніка.

В роботі [6] на підставі аналізу структури традиційного звіту про дефекти програмних продуктів, виявлених при тестуванні, створена модифікована структура звіту. Тестування проводиться по BDD, TDD модулю, за умови що спочатку йде написання тестів, а лише потім основного функціоналу. Автор зазначає в своїй роботі важливість етапу тестування, оскілки результати тестування на пряму зв'язані з якість програмного продукту. Джеррі Вайнберг у своїй книзі [7], відзначає, що те, що ми отримуємо в якості результату – це перш за все інформація і не можна залишати поза оцінки потенційно значущі спостереження.

В роботі [8] авторами розглянуто процес тестування програмного забезпечення з боку повторення на принципі ітераційного циклу. На кожному кроці виконується повторення, щоб знайти нові помилки,

які виявляються і виправляються, при цьому вони не були знайдені раніше. Окрім цього відбувається прогнозування часу виконання, а також прогнозування затрати на процес тестування, враховуючи його циклічну повторюваність. Автори в роботі [9] розглядають тестування не як глобальний процес, а як надійність, стабільність роботи і прогнозування помилки, це все аналізується по модулях: щільність помилок, часу на виконання і інших. Для цього було розроблено нову математичну модель надійності програмного забезпечення з динамічним показником величини програмного проекту. Окрім цього в роботі наводиться порівняльний аналіз існуючих та розробленої моделей на реальних тестових прикладах.

Припустимо, результат може спотворюватися від людського фактора і тестування одного і того ж може бути представлено з різними похибками: неуважність тестувальника, його завантаженість в проекті, рутинне виконання однакових операцій. Зменшення впливу людського фактора дасть більш достовірні дані про пройдений етап тестування, а систематизація показників і побудови кінцевої оцінки на основі цих даних дозволить зменшити похибку і прискорити прийняття рішення.

Етап Тестування в ітераційної моделі життєвого циклу може бути представлений як оцінка відповідностей, відповідно до стандарту [3]. А кінцева оцінка, виходячи з оцінки критеріїв, буде узагальнювати отримані результати і представлена як Ступінь впевненості в якості програмного продукту.

Оцінити множину факторів можна за допомогою нечіткої логіки. Оцінкою параметрів буде ступінь впевненості в якості програмного продукту, тобто оцінка може бути частково правдива і частково помилкова. Побудови моделі виконується за допомогою математичних виразів. Де, збільшення числа змінних (атрибутів) і параметрів пропорційно впливає на складність системи [10].

На підставі обраних параметрів, що впливають на якість програмного продукту була побудована нечітка модель, представлена у вигляді(1).

$$L = <\{V\}_{i=1}^{5};\{R\}_{k=1}^{95};\{W\}_{j=1}^{1}>, \quad (1)$$

де {V} – множина вхідних змінних, $\{V\}_{i=1}^{5} = \{\beta_1, \beta_2, \beta_3, \beta_4, \beta_5\}$;

{R} – множина правил $\{R\}_{k=1}^{95} = \{R_1, R_2, ...R_{95}\}$;

{W} – множина вихідних змінних $\{W\}_{j=1}^{1} = \{\varpi_1\}$.

Кожне правило R_i представлено у вигляді нечіткої продукції виду (2).

$$ПРАВИЛО <\#>: ЯКЩО\ "\beta_1 \epsilon \alpha_1"\ TA\ "\beta_2 \epsilon \alpha_2"\ TA ..."\beta_m \epsilon \alpha_m"\ TO\ "\varpi_1 \epsilon y_1"\ TA ..."\varpi_S \epsilon y_S" \quad (2)$$

Аналізуючи предметну область було побудовано 95 продукційних правил, така кількість обумовлено специфікою предметної області і виключає взаємовиключні правила.

У якості схеми нечіткого висновку пропонується використовувати алгоритм Мамдані [10]: метод активації – min-активація, у всіх правилах у якості логічної зв'язки для умов застосовується нечітка кон'юнкція, в якості метода агрегування використовується min-кон'юнкція, для акумуляції висновків правил – метод max-диз'юнкції, метод деффазіфікаціі – метод центру тяжіння.

Для побудованої нечіткої моделі доречні функції приналежності сігмоїдної, на крайніх значеннях і дзвін для значень між. Обрані дані функції, так як вони нелінійні і не мають різких перепадів, що підходить для вхідних значень. У якість параметрів, що впливають на якість програмного продукту відповідно до стандарту [3], були обрані основні критерії оцінки тестування і шкала оцінок:

– «придатність» діапазон універсуму [0; 100];
– «точність» діапазон універсуму [0; 10];
– «взаємодія» діапазон універсуму [0; 100];
– «захищеність» діапазон універсуму [0; 10];
– «відповідність» діапазон універсуму [0; 100].

Згідно особливостям предметної області ключові параметри представлені у вигляді лінгвістичних змінних.

Вихідна лінгвістична змінна ω_1 визначається кортежем $<\varpi_1, T(\varpi_1), X>$, де ϖ_1=«ступінь впевненості в якості програмного продукту», T(ϖ_1)={DL, DA, DH}, X=[0; 1]. Дана змінна є вихідний в нечіткої моделі (1), формується на підставі 5 вхідних лінгвістичних змінних і відображає ступінь впевненості протестованого продукту в його якості, для подальшого переходу до експлуатації.

Критерії змінної представлені в таблиці 1. Параметри функцій належності були обрані емпіричним шляхом.

Таблиця 1. Параметри термів лінгвістичної змінної ω_1

Позначення терму	Назва терму	Тип функції приналежності	Параметри			Вихідний універсум
			a	b	c	
DL	низька	сигмоїдна	-13,7	-	0,2394	[0; 0.35]
DA	середня	узагальнений дзвін	0,13	3,0	0,5081	[0,35; 0,7]
DH	висока	сигмоїдна	13,7	3,50	0,8	[0,7; 1]

Моделювання нечіткої системи вибору оптимальної поїздки в середовищі Matlab.

Для перевірки розробленої нечіткої моделі було проведено моделювання, що дозволяє оцінити вплив вхідних лінгвістичних змінних на значення вихідної змінної і проводити аналіз залежностей. Побудови моделі нечіткої логіки було виконано за допомогою інструменту Matlab R2009b в пакеті Fuzzy Logic Toolbox. На рис.2 представлена структурна схема у вікні редактора системи нечіткого виведення.

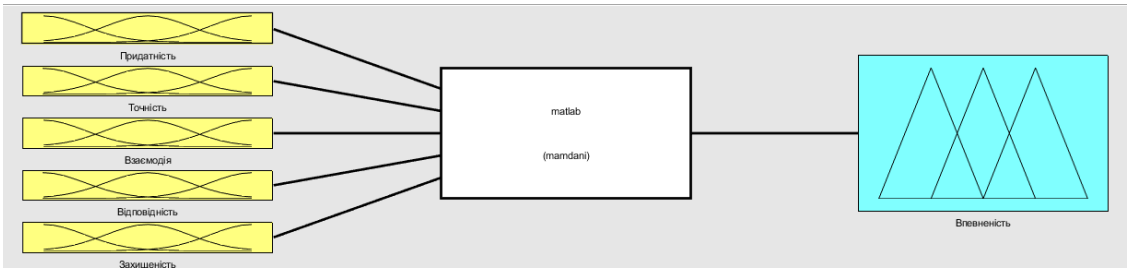
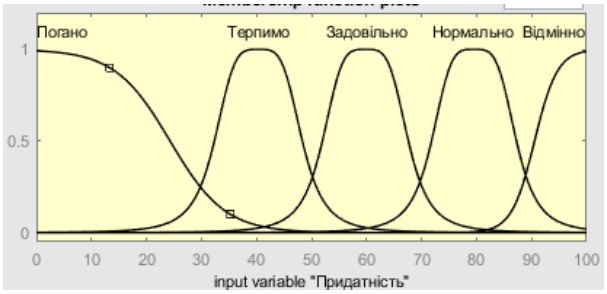
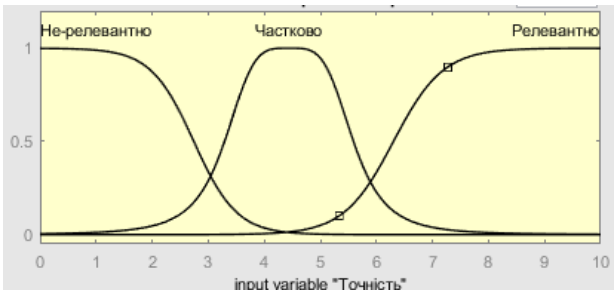


Рис. 2. Побудова системи у Matlab

При створенні моделі проходить кілька етапів, основні особливості яких наведено нижче. Після завдання лінгвістичних змінних на етапі фазифікації значення вхідних змінних приводиться до відповідного нечіткого значення. Лінгвістична оцінка значення для змінної «функціональна придатність», проводиться за допомогою 5 термів: («погано», «терпимо», «задовільно», «нормально», «відмінно») рис. 3а. Лінгвістична оцінка значення для змінної «точність», проводиться за допомогою 3 термів: («не релевантно», «частково», «релевантно»), рис. 3б.



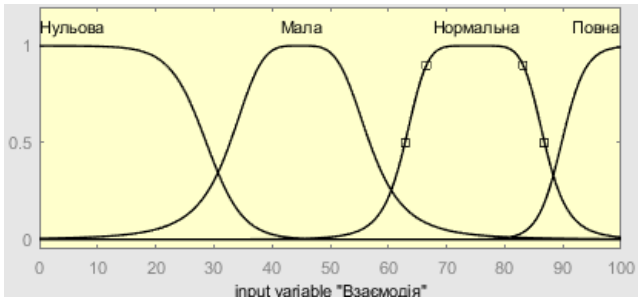
а) змінна функціональна придатність



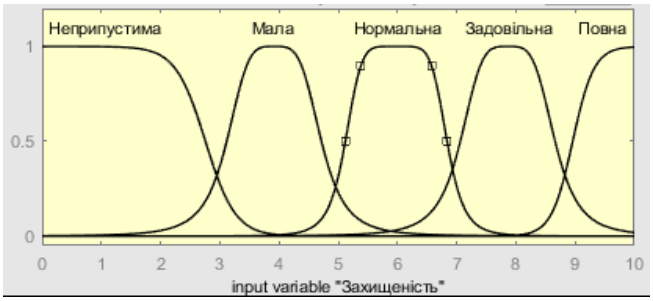
б) змінна точність

Рис. 3. Завдання лінгвістичної оцінки вхідних сигналів

Лінгвістична оцінка значення для змінної «здатність до взаємодія», проводиться за допомогою 3 термів: («нульова», «мала», «нормальна», «повна») рис. 4а. Лінгвістична оцінка значення для змінної «захищеність», проводиться за допомогою 3 термів: («неприпустима», «мала», «нормальна», «задовільна», «повна»), рис. 4б.



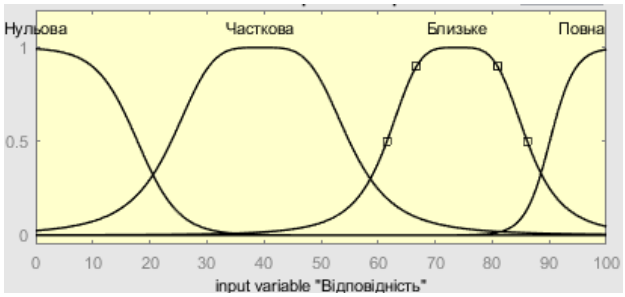
а) змінна здатність до взаємодії



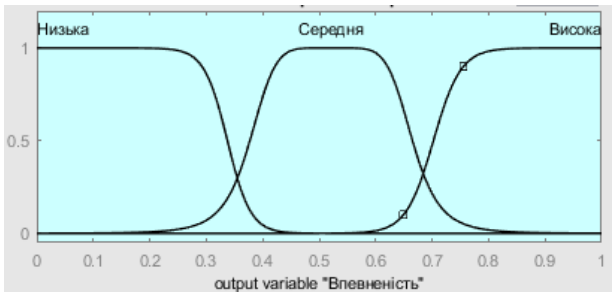
б) змінна захищеність

Рис. 4. Завдання лінгвістичної оцінки вхідних сигналів

Лінгвістична оцінка значення для змінної «відповідність»,проводиться за допомогою 3 термів: («нульова», «часткова», «близька», «повна»), рис. 5а. Лінгвістична оцінка значення для вихідної змінної «ступінь впевненості у якості програмного забеспечення», проводиться за допомогою 3 термів: («низька», «середня», «висока»), рис. 4б.



а) вхідні змінна відповідності



б) вихідна змінна ступінь впевненості

Рис. 5. Завдання лінгвістичної оцінки вхідних і вихідних змінних

Далі відбувається завдання правил для системи нечіткого виведення. Всього було створено 95 правил, які повністю описують всілякі поєднання перебору вхідних даних і відповідних вихідних даних. Вид графічного інтерфейсу редактора правил після завдання всіх правил зображений на рис. 6.

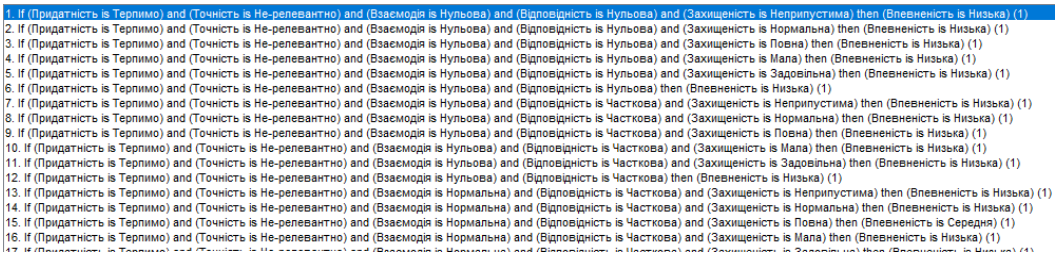


Рис. 6. Правила залежностей вихідної змінної від вхідних

На основі цих даних, можна побудувати правила залежностей вихідної змінної від вхідних та виконати оцінку побудованої системи нечіткого виведення для задачі визначення ступеня впевненості в якості програмного забезпечення. Для цього відкриємо вікно перегляду (рис. 7) і введемо значення вхідних змінних для окремого випадку: «функціональна придатність» – 41, «точність» – 4,17, «здатність до взаємодія» – 77,8, «відповідність» – 47, «захищеність» – 3,65. Процедура нечіткого виведення, реалізована в системі MatLab Fuzzy для розробленої нечіткої моделі, видає результат 0,334, що інтерпретується в якість програмного продукту - «Ступінь впевненості у якості програмного продукту», терм «низька» тим самим підтверджує її адекватність, в рамках даної моделі.

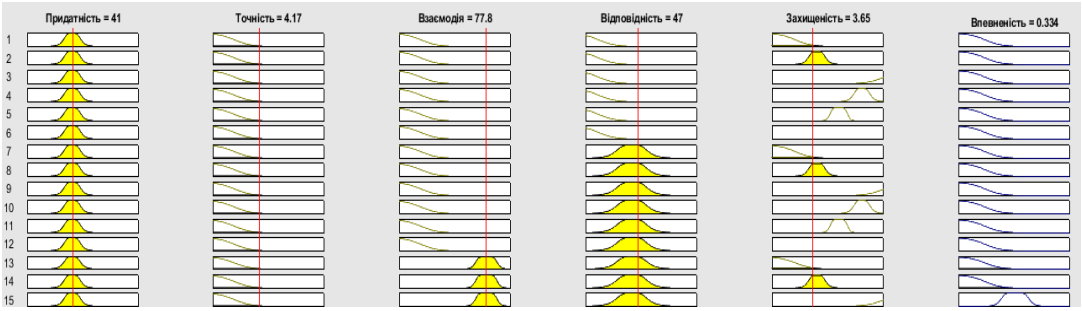


Рис. 7. Інтерфейс перегляду правил системи нечіткого виводу

Процес дослідження і аналізу розробленої нечіткої моделі складається з тестового виконання нечітких висновків для різних значень вхідних змінних і оцінки отриманих результатів з метою внесення необхідних коригувань у разі неузгодженості окремих результатів. Окрім цього на етапі моделювання можна встановити залежність значення вихідної змінної «ступінь впевненості у якості програмного забезпечення» від значень вхідних змінних «функціональна придатність», «точність», «здатність до взаємодія», «захищеність».

Висновки. Була розроблена нечітка модель визначення ступеня впевненості в якості програмного продукту, що розробляється. Дана модель була змодельована за допомогою Matlab в пакета Fuzzy, на основі системи нечіткої логіки, яка дає можливість аналізувати вплив вхідних параметрів на вихідний. В рамках сфери застосування, можна уявити етап тестування, як набір вхідних лінгвістичних змінних, саме функціональна придатність, точність, здатність до взаємодії, відповідність, захищеність, де від оцінки кожної змінної залежить ступінь впевненості в якості програмного продукту. Розроблена нечітка модель застосовувалася для моделі ітераційного життєвого циклу програмного продукту. Даний підхід універсальний і може бути застосований для моделей іншої категорії, наприклад, каскадної за рахунок більш м'якого повторення етапу тестування.

Дані знання та модель можна застосувати в області автоматизації процесу тестування. Такий підхід збільшить в рази швидкість проходу даного етапу, зменшить вплив людського чиннику, мінімізує можливу помилку і оптимізує процес розробки.

Список бібліографічних посилань

1. Програмна інженерія. Підручник для ядра знань програмної інженерії (ISO 19759: 2016, IDT) : ДСТУ ISO 19759: 2016. – [Чинний від 2016–10–07] . Київ: Держ. стандарт України, 2016. – 336 с. (Національний стандарт України).
2. Програмна інженерія. Якість продукту. Внутрішні метрики (ISO9126:2012, IDT) : ДСТУ ISO9126:2012. – [Чинний від 2012–11–28] . Київ : Держ. стандарт України, 2012. – 46 с. (Національний стандарт України).
3. Інженерія систем і програмних засобів. Вимоги до якості систем і програмних засобів та її оцінювання (ISO 25010:2016, IDT) : ДСТУ ISO 25010:2016. – [Чинний від 2016–12–27] . – Київ :Держ. Стандарт України, 2016. – 36 с. (Національний стандарт України).
4. Heusser M. How to Reduce the Costof Software Testing / Matthew Heusser,Govind Kulkarni – United States of America, 2011. – 340 p.
5. Трофімук А. В. Оцінка якості програмного забезпечення за показниками надійності / А. В. Трофімук; наук. кер. С. С. Федін // Наукові розробки молоді на сучасному етапі : тези доповідей XVII Всеукраїнської наукової конференції молодих вчених та студентів (26-27 квітня 2018 р., Київ). - Київ : КНУТД, 2018. – Т. 2 : Мехатронні системи і комп'ютерні технології. Ресурсозбереження та охорона навколишнього середовища. – С. 369 – 370.
6. Щербаков О. В. Оцінка ефективності тестування програмного забезпечення на основі аналізу кількості та критичності знайдених дефектів / О.В. Щербаков, Є.С. Луценко - Системи обробки інформації, 2011, випуск 3 (93). – С. 88 – 92.
7. Weinberg G. Perfect Software And Other Illusions about Testing / GeraldWeinberg – United States of America, 2011. – 200 p.
8. Дубовой В.М. Моделювання процесу тестування програмного забезпечення як розгалуженоциклічного технологічного процесу / В. М Дубовой, І. В. Пилипенко // Автоматизація технологічних і бізнес-процесів, 2015. – № 7 (4). – С. 55 – 64.
9. Критерій достатності процесу тестування програмного забезпечення / В. Яковина, М. Сенів, Я. Чабанюк, Д. Федасюк, У. Хімка //Комп'ютерні науки та інформаційні технології : [зб. наук. пр.] / відп. ред. Ю.М. Рашкевич. – Л. : Вид-во Львів. політехніки, 2010. – С. 346 – 358.
10. Леоненков А. В. Нечеткое моделирование в среде MATLAB и fuzzy TECH / А. В. Леоненков – СПб. : БХВ-Петербург, 2005. – 736 с.