УДК 004.9.005.95 DOI: https://doi.org/10.33099/2304-2745/2023-1-77/69-78

Рибидайло А. А., кандидат технічних наук, старший науковий співробітник (0000-0002-6156-469X)
Галаган В. І., кандидат військових наук, доцент (0000-0001-9578-0895)
Васюхно С. І. (0000-0002-0884-0405)
Мулявка А. С. (0000-0002-3113-0719)
Руденська Г. В. (0000-0002-4719-3765)

Центр воєнно-стратегічних досліджень Національного університету оборони України імені Івана Черняховського, Київ

Порядок організації створення спеціального програмного забезпечення для інформаційних систем військового призначення

Резюме. Розглянуто стратегії розроблення програмного забезпечення. Обґрунтовано підхід до порядку розроблення програмного забезпечення для інформаційних систем військового призначення з урахуванням умов ведення відповідного проєкту.

Ключові слова: програмне забезпечення; модель життєвого циклу; інформаційна система; каскадна, інкрементна, еволюційна стратегія; верифікація; валідація.

Постановка проблеми. Програмними документами України визначені завдання щодо створення єдиної інформаційної системи управління оборонними ресурсами, як інструментарію з розвитку та підвищення ефективності процесів планування та управління Збройними Силами України (ЗС України), їх технічного оснащення та всебічного забезпечення, підвищення рівнів підготовки та готовності до виконання покладених завдань, скорочення часу виконання завдань і розвитку спроможностей [1]. Питання створення Єдиної автоматизованої системи управління ЗС України (ЄАСУ ЗС України) та її окремих автоматизованих систем управління (АСУ), інформаційно-аналітичних систем (ІАС), програмних продуктів, що створюються в ході виконання програм автоматизації є актуальним.

Одним із критеріїв якісної роботи інформаційних систем ЄАСУ ЗС України є, по можливості, найповніша автоматизація процесів її функціонування. Це особливо важливо за необхідності обробляти величезні потоки даних. Динамічна зміна середовища безпеки диктує необхідність технологічних удосконалень. Тому інформаційні системи ЄАСУ не повинні відставати у питанні актуальності використовуваного програмного забезпечення (ПЗ). При цьому, зокрема, має забезпечуватись належна якість програмного продукту для конкретної АСУ (ІАС) при заощадливому використанні коштів бюджету Міністерства оборони України (МО України) на її розроблення, уникнення або зменшення проєктних ризиків. Одним із шляхів виконання означених вимог є створення спеціального програмного забезпечення (СПЗ) з використанням апробованих моделей життєвого циклу (ЖЦ), які є найбільш актуальними для умов виконання конкретного проєкту. Порядок створення СПЗ для ІС військового призначення має особливості, що притаманні галузі застосування означених ІС. Актуальним вбачається обґрунтування порядку створення СПЗ для ІС, які використовуються у Збройних Силах (ЗС) України.

Аналіз останніх досліджень і публікацій. Окремі питання застосування моделей життєвого циклу (ЖЦ) СПЗ розглядаються у багатьох джерелах, зокрема [2–6], де висвітлюються:

основні процеси життєвого циклу програмного забезпечення;

базові стратегії розробки програмних засобів та систем;

моделі життєвого циклу, що реалізують каскадну, інкрементну та еволюційну стратегії розробки програмних засобів;

основні переваги, недоліки і умови застосування каскадної інкрементної та еволюційної стратегій розробки програмних засобів.

Проте в означених джерелах систематизований аналіз моделей ЖЦ стосовно доцільності їх використання в конкретних проєктах інформатизації наведено фрагментарно. Крім того, надані рекомендації стосовно застосування СПЗ притаманні

бізнес-структурам, де доцільність обумовлена очікуваним у майбутньому прибутком.

Мета статті – обґрунтування рекомендацій щодо порядку створення спеціального програмного забезпечення для інформаційних систем військового призначення з урахуванням умов ведення проєкту.

Виклад основного матеріалу. *Програмне забезпечення* (ПЗ) – це сукупність програмних засобів і документації, що супроводжує їх, які дозволяють вирішувати на комп'ютері завдання різного призначення в економічній, управлінській та інших сферах діяльності, а також забезпечують функціонування апаратних засобів ЕОМ. Укрупнено ПЗ підрозділяється на системне (загальне) та прикладне (спеціальне).

Спеціальне ПЗ (СПЗ) представляє найбільший інтерес для користувача, тому що саме воно призначене для вирішення конкретних завдань АСУ (ІАС), що стосуються функціонування військового відомства: виробництво, накопичення та доставка оборонних ресурсів; розподіл вогневих засобів при веденні бою; підготовка звітних документів тощо. Яке прикладне забезпечення знадобиться конкретному користувачеві, залежить від його діяльності, поставлених завдань та посадових обов'язків.

Життєвий цикл програмного забезпечення – це умовна схема, що включає окремі етапи (стадії) процесу створення програмного забезпечення.

При цьому на кожному етапі циклу виконуються різні дії. Життєвий цикл програмного забезпечення (ЖЦ ПЗ) поділяється на впорядковані етапи, основні з яких наведені на рис. 1.

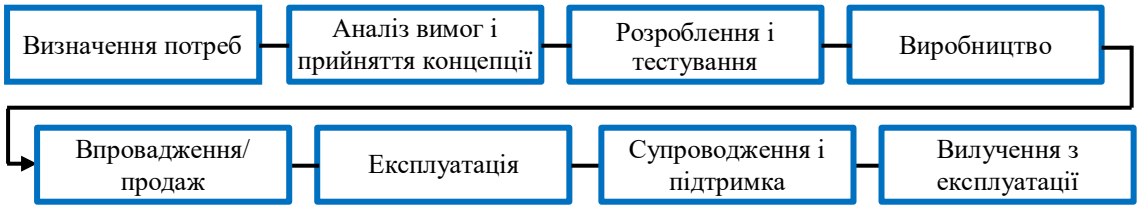

Рис. 1. Етапи життєвого циклу програмного забезпечення

Всередині кожного з етапів відбувається подальша деталізація по більш дрібним стадіям. Моделі життєвого циклу програмного забезпечення описують взаємозв'язки стадій. На рис. 2 наведені базові етапи життєвого циклу, які безпосередньо пов'язані із процесом розроблення ПЗ:

специфікація – визначення основних вимог.

розроблення – створення ПЗ відповідно до специфікацій.

тестування – перевірка ПЗ на відповідність вимогам Замовника.

супроводження/модернізація – розвиток ПЗ відповідно до змін потреб Замовника.

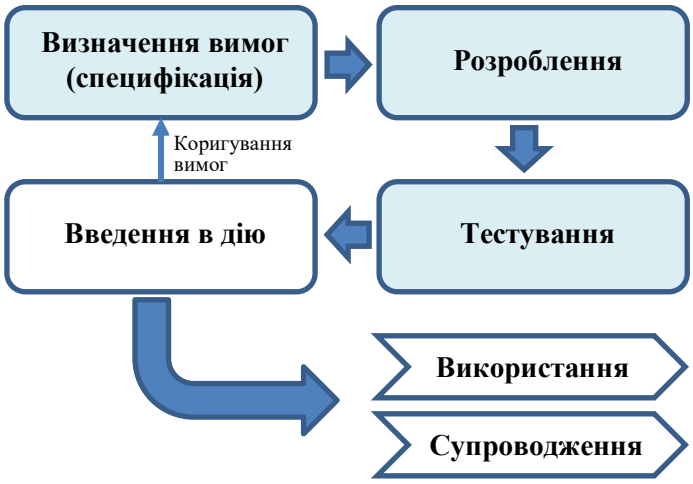

Рис. 2. Життєвий цикл програмного забезпечення

До теперішнього часу створені та широко використовуються три базові стратегії розробки ПЗ: каскадна; інкрементна; еволюційна. Деякі з цих стратегій розробки ПЗ та вимоги до них наведені в [8].

Каскадна стратегія являє собою одноразовий прохід етапів розробки.

Стандартна каскадна модель життєвого циклу відноситься до моделей послідовного виконання стадій ЖЦ ПЗ і передбачає, що кожна наступна фаза розпочинається тільки тоді, коли повністю завершене виконання попередньої фази. Кожна фаза каскадної моделі має свої певні критерії входу та виходу – вхідні та вихідні дані. Спроби оптимізації каскадної моделі ЖЦ ПЗ привели до виникнення інших циклів розроблення програмного забезпечення:

каскадна модель життєвого циклу із зворотнім зв'язком – розширює стандартну каскадну модель шляхом включення до неї циклів зворотного зв'язку із поверненням на попередню стадію при зміні вимог, проєкту і за результатами інспекції або дій по верифікації та валідації (V&V);

V-подібна модель життєвого циклу програмного забезпечення – існує можливість перевірки і оцінювання придатності вимог до тестування на ранніх стадіях розроблення та документування тестових вимог;

V-подібна модель життєвого циклу з організацією зворотних зв'язків між сусідніми етапами – базується на можливості паралельного розроблення програмних продуктів, що входять до складу системи, різними командами розробників, а також вибору програмних об'єктів з існуючих або розроблених раніше;

каскадна модель з прототипуванням – модифікація V-подібної моделі шляхом включення до неї прототипів для моделювання вимог і демонстрації проєкту (після демонстрації прототипи знищуються, а сама реалізація проєкту може виконуватись у іншому середовищі).

У Табл. 1 наведені переваги та недоліки моделей послідовного виконання стадій життєвого циклу ПЗ.

Таблиця 1

Переваги та недоліки моделей послідовного виконання стадій життєвого циклу ПЗ

Назва моделі	Переваги	Недоліки
Стандартна каскадна модель	Розгорнуте документування кожного етапу розроблення. Чітке планування термінів і витрат. Прозорість усіх процесів розроблення для Замовника	Необхідність погодження повного обсягу вимог на першому етапі. За потреби внесення змін до вимог, перероблення всієї виконаної роботи. Збільшення обсягу витрат коштів і часу за необхідності зміни вимог
Каскадна модель із зворотнім зв'язком	Застосування перевірок дозволяє вчасно виявляти дефекти. Чіткі критерії початку і завершення стадій. Чіткі вимоги і цілі проєкту	Формальні перевірки по завершенні кожної стадії (інспекції, технічні огляди)
V-подібна модель	Забезпечує зворотний зв'язок з користувачем на ранніх стадіях ЖЦ. Покращує планування і розподіл витрат на тестування. Чіткі документовані цілі тестування	Важко вирішити завдання з паралельними подіями. Не враховуються ітерації поміж фазами. Не передбачено внесення вимог динамічних змін на етапах ЖЦ. Тестування відбувається надто пізно, що впливає на графік виконання проєкту
Каскадна модель з прототипуванням	Усунення проблем, пов'язаних з неповнотою і нечіткістю вимог	Потреба на завершальній стадії проведення всебічного тестування

Моделям послідовного виконання стадій ЖЦ ПЗ (каскадна стратегія) притаманні практично однакові недоліки, а саме: не повна зрозумілість вимог на початку проєктування, їх часта змінність, а також швидкий розвиток технологій не дозволяють масштабні проєкти реалізувати одразу. Крім того відсутня можливість демонстрації Замовнику проміжних результатів розроблення програмного продукту.

Інкрементна стратегія є багаторазовим проходом етапів розробки із запланованим поліпшенням результату. Ця стратегія заснована на повному визначенні всіх вимог до програмного засобу, що розробляється, на початку процесу розробки. Однак повний набір вимог реалізується поступово відповідно до плану у послідовних циклах розробки.

У разі використання інкрементних моделей здійснюється початкова часткова реалізація програмного засобу. При цьому насамперед реалізуються базові вимоги. За цим слідує повільне нарощування функціональних можливостей або характеристик якості програмного засобу в прототипах (інкрементах), що реалізуються послідовно. У кожній наступній версії програмного засобу додаються до попередньої версії певні програмні компоненти, що реалізують відповідні функціональні можливості, доки не будуть реалізовані всі

вимоги до програмного засобу. Кожна версія може здаватися в експлуатацію.

Відповідно до цих моделей програмний продукт розробляється ітераціями і кожна закінчується випуском працездатної версії програмного продукту. Основна відмінність від каскадних моделей – підхід до визначення вимог.

Інкрементна (ітераційна) розробка представляє собою процес часткової реалізації ПЗ з нарощуванням функціональних можливостей.

Пришвидшенню процесу створення функціонуючої системи сприяє застосовування принципу компонування стандартних блоків, завдяки чому забезпечується контроль процесу розробки змінних вимог.

У класичних варіантах інкрементна модель заснована на використанні заздалегідь сформованого набору вимог, що реалізуються послідовно у вигляді невеликих проєктів. Існують варіанти інкрементної моделі, що починаються з формулювання загальних цілей, які поступово уточнюються у процесі розроблення прототипів:

інкрементна модель з уточненням вимог на початкових етапах розробки – застосування зворотних зв'язків дозволяє проводити уточнення вимог до системи, виконувати проєктування її архітектури з урахуванням вимог, що змінилися, уточнювати вимоги до ПЗ;

інкрементна модель по ДСТ Р ІСО/МЕК ТО 15271–2002 – заснована на використанні повного заздалегідь сформованого набору вимог та їхньої поступової реалізації в окремих інкрементах;

інкрементна модель екстремального програмування.

При використанні інкрементна модель екстремального програмування:

на першому етапі розробки зусилля розробників витрачаються на ретельне проєктування архітектури програмного засобу (у подальшому уточнення чи зміна архітектури не передбачається);

на наступних етапах кожна нова функціональна вимога, що надійшла до поточного моменту від Замовника, оцінюється за вартістю та часом реалізації з урахуванням невизначеності даних та ризиків його реалізації;

за результатами виконаних оцінок Замовник обирає та затверджує нові вимоги, які будуть реалізовуватись у черговій версії ПЗ;

якщо в черговому варіанті ПЗ знайдені помилки, то даний варіант повертається на наступну ітерацію реалізації;

процес продовжується, поки результати чергових приймальних випробувань не будуть затверджені Замовником (затверджений варіант системи ПЗ надходить в експлуатацію як чергова версія).

Основні *переваги інкрементної стратегії*:

можливість одержання функціонального продукту після реалізації кожного інкременту;

коротка тривалість створення інкременту, що призводить до скорочення строків постачання програмного продукту;

запобігання реалізації громіздких специфікацій вимог та можливість урахування вимог, що змінилися;

зниження ризиків у порівнянні з каскадною стратегією;

включення в процес користувачів, що зрештою призводить до підвищення якості програмного продукту, зниження витрат та часу на його розробку.

До *основних недоліків інкрементної стратегії*, що виявляються внаслідок її невідповідного застосування, слід віднести:

необхідність повного функціонального визначення програмного засобу на початку ЖЦ для забезпечення планування інкрементів та управління проєктом;

можливість поточної зміни вимог до системи чи програмного засобу, які вже реалізовані у попередніх інкрементах;

складність планування та розподілу робіт;

прояв людського фактору, пов'язаного з тенденцією до відтягування вирішення важких проблем на пізні інкременти, що може порушити графік робіт або знизити якість програмного продукту.

Еволюційна стратегія – багаторазовий прохід етапів розробки. Ця стратегія заснована на частковому визначенні вимог до програмного засобу, що розробляється на початку процесу розробки. Вимоги поступово уточнюються у послідовних циклах розробки. Результат кожного циклу розробки зазвичай є черговою версією програмного засобу, що поставляється. Слід зазначити, що у випадку еволюційної стратегії характерно значно менше циклів розробки за більшої тривалості циклу проти інкрементної стратегії. При цьому результат кожного циклу розробки (чергова версія програмного засобу) набагато сильніше відрізняється від результату

попереднього циклу. Еволюційні моделі базуються на використанні прототипування:

класична еволюційна модель (ДСТ Р ІСО/МЭК ТО 15271–2002) – розробка кожної версії програмного засобу виконується на основі каскадної моделі, що містить чотири етапи: розробка вимог, проєктування, програмування та тестування, введення в дію та підтримка продукту;

структурна еволюційна модель швидкого прототипування – проміжні прототипи призначені лише для уточнення вимог; на останньому циклі (детальна розробка) використовуються етапи каскадної моделі з метою забезпечення якості ПЗ;

еволюційна модель прототипування ДСТ Р ІСО/МЭК ТО 15271–2002 – розробка кожного прототипу виконується в середовищі мови програмування четвертого покоління 4GL для швидкого проєктування та складання програмного засобу, а також його оперативного нарощування, зміни та уточнення;

спіральна модель Боема – для кожного циклу моделі аналізуються вимоги, альтернативні варіанти та обмеження; визначаються, скорочуються чи усуваються ризики; розробляється версія продукту цього циклу спіралі та підтверджується її правильність; планується наступний цикл та вибираються методи його здійснення;

спрощені варіанти спіральної моделі – модель Інституту якості SQI, модель Інституту управління проєктами PMI, модель “win-win”, спіральна модель Консорціуму з питань розробки програмного забезпечення – створені для зниження складності базової спіральної моделі Боема за рахунок усунення зайвої деталізації.

Основні *переваги еволюційної стратегії*:

можливість уточнення та внесення нових вимог у процесі розробки;

придатність проміжного продукту для використання;

можливість управління ризиками;

забезпечення широкої участі користувача у проєкті, починаючи з ранніх етапів, що мінімізує можливість розбіжностей та забезпечує створення продукту високої якості;

реалізація переваг каскадної та інкрементної стратегій.

Недоліки еволюційної стратегії:

невідомість точної кількості необхідних ітерацій та складність визначення критеріїв для продовження процесу розробки на наступній ітерації;

складність планування та управління проєктом;

необхідність активної участі користувачів у проєкті, що реально не завжди можливо;

необхідність у потужних інструментальних засобах та методах прототипування;

можливість відсунення вирішення складних проблем на наступні цикли, що може призвести до невідповідності отриманих продуктів вимогам Замовника.

Окремо доцільно виділити технологію розроблення додатків ПЗ RAD. *Модель швидкої розробки додатків* RAD (Rapid Application Development) з’явилася у 80-ті роки XX ст. у зв’язку з бурхливим розвитком потужних технологій та інструментальних засобів розробки програмних продуктів. Ця модель, виходячи з особливостей її реалізації та умов виконання проєкту, може використовуватись як самостійна ітераційна модель ЖЦ ПЗ , так і для підтримки інкрементної або еволюційної стратегій розробки ПЗ. Основою RAD-моделі є використання потужних інструментальних засобів розробки. У процесі швидкої розробки додатків основна увага приділяється не програмуванню та тестуванню, а аналізу вимог та проєктуванню. Використання потужних інструментальних засобів передбачає щільну комунікацію Розробника з Користувачем, що дає змогу дати оцінку продукту на всіх етапах його розробки. Характерною рисою RAD-моделі є короткий час переходу від аналізу вимог до створення повної системи або програмного засобу. На сьогодні апробовані такі різновиди RAD-моделей:

базова RAD-модель містить чотири етапи - може використовуватися в окремих ітераціях деякої еволюційної моделі та як незалежна модель у невеликих проєктах:

RAD-модель, заснована на моделюванні предметної галузі містить п’ять етапів – може вбудовуватися в інкрементні та еволюційні моделі як основа реалізації окремих ітерацій;

RAD-модель паралельної розробки додатків – підтримує паралельну розробку програмних компонентів різними групами розробників, що реалізують базові функції ПЗ, з подальшою інтеграцією компонентів у єдиний програмний засіб;

модель швидкої розробки додатків за ДСТ Р ІСО/МЕК ТО 15271-2002 – фактично

реалізує еволюційну стратегію розробки ПЗ (у кожному циклі розробки реалізується уточнення функціональних і не функціональних вимог, розробка функціонального прототипу, розробка версії програмного засобу, введення в дію та експлуатація цієї версії).

Наведений огляд моделей ЖЦ ПЗ дозволяє запропонувати їх класифікацію, яка наведена на рис. 3.

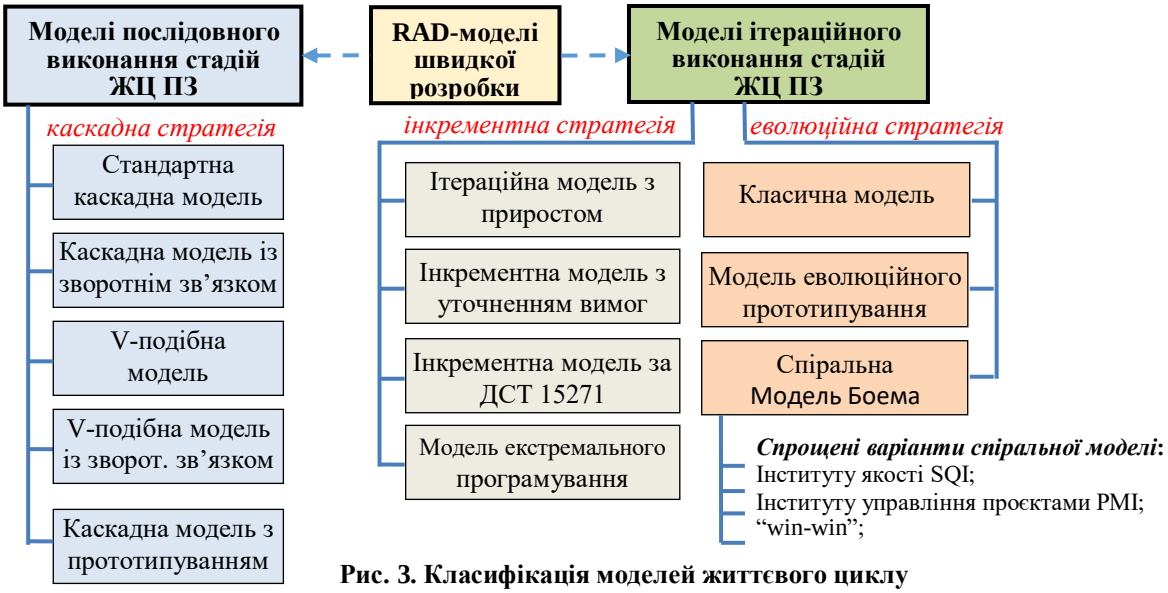

Рис. 3. Класифікація моделей життєвого циклу програмного забезпечення

Як видно з наведеного огляду моделей ЖЦ ПЗ Керівнику проєкту щодо створення СПЗ для ІС військового призначення необхідно обґрунтовано обрати ту чи іншу стратегію виконання проєкту, тобто найбільш прийнятну модель ЖЦ ПЗ.

Вибір моделі життєвого циклу ПЗ суттєво залежить від двох факторів:

A – можливість визначення повного набору функцій, які необхідно реалізувати в програмному продукті;

B – потреба постачання Замовнику програмного продукту, у якому одночасно реалізовані усі жадані функції.

На рис 4 наведено алгоритм вибору моделі життєвого циклу програмного забезпечення для створення продукту із врахуванням наявності (відсутності) факторів ***A*** і ***B***.

$A+B$ – каскадна модель $A+\overline{B}$ – ітераційна модель з приростом; $\overline{A}+B$ – спіральна модель;

$\overline{A}+\overline{B}$ – модель швидкої розробки.

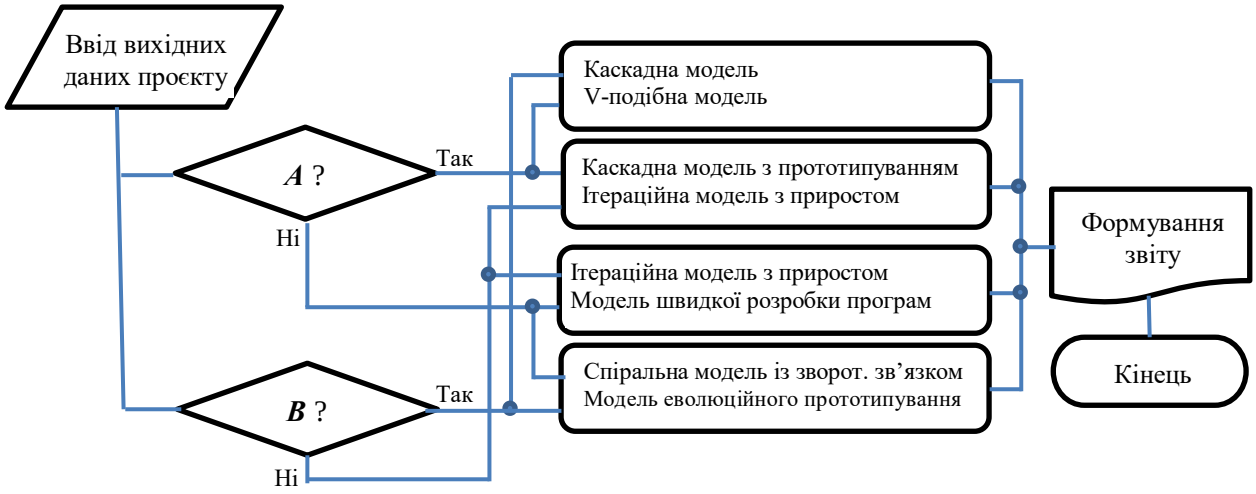

Рис. 4. Алгоритм вибору моделі життєвого циклу ПЗ

Наведений алгоритм дозволяє отримати альтернативи для вибору раціональної моделі ЖЦ ПЗ, проте врахування лише факторів ***A*** і ***B*** для однозначної відповіді недостатньо. Для остаточного рішення потрібно врахувати інші умови виконання проєкту, а саме:

наявний ресурс (розміри бюджету, кадровий потенціал, власні спроможності до

тестування програмного продукту та оцінювання його якості);

ризики при використанні обраної моделі ЖЦ ПЗ та спроможність управляти ними;

термін виконання проєкту.

Інститутом якості програмного забезпечення SQI (Software Quality Institute, США) спеціально для вибору моделі ЖЦ запропоновано схему класифікації проєктів з розробки ПС та систем [8]. Основу даної класифікації становлять чотири категорії критеріїв: характеристики вимог до проєкту; характеристики команди розробників; характеристики користувачів (замовників); характеристики типів проєктів та ризиків. За кожним критерієм проєкти поділяються на два альтернативні класи: можна чи не можна використовувати. Процедура базується на застосування чотирьох таблиць питань. Приклади питань наведено у Табл. 2–5.

Таблиця 2

Вибір моделі життєвого циклу з урахуванням характеристик вимог

№	Умови використання (вимоги)	МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ ПЗ				
		Каскадні	RAD	Інкрементні	Швидкого прототипування	Еволюційні
1	Чи є вимоги до проєкту легко визначимими та реалізованими?	Так	Так	Ні	Ні	Ні
2	Чи можуть бути вимоги сформульовані на початку ЖЦ?	Так	Так	Так	Ні	Ні
3	Чи часто будуть змінюватися вимоги протягом ЖЦ?	Ні	Ні	Ні	Так	Так
4	Чи потрібно демонструвати вимоги з метою їхнього визначення?	Ні	Так	Ні	Так	Так
5	Чи потрібна перевірка концепції ПЗ?	Ні	Так	Ні	Так	Так
6	Чи змінюватимуться вимоги зі зростанням складності ПЗ у ЖЦ?	Ні	Ні	Так	Так	Так
7	Чи потрібно реалізувати основні вимоги на ранніх етапах розробки?	Ні	Так	Так	Так	Так

Таблиця 3

Вибір моделі життєвого циклу на основі характеристик команди розробників

№	Умови використання (вимоги)	МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ ПЗ				
		Каскадні	RAD	Інкрементні	Швидкого прототипування	Еволюційні
1	Чи є проблеми предметної галузі проєкту новими для більшості розробників?	Ні	Ні	Ні	Так	Так
2	Чи є інструментальні засоби, що використовуються у проєкті, новими для більшості розробників?	Так	Ні	Ні	Ні	Так
3	Чи змінюються ролі учасників проєкту протягом ЖЦ?	Ні	Ні	Так	Так	Так
4	Чи є структура процесу розробки більш значущою для розробників, ніж гнучкість?	Так	Ні	Так	Ні	Ні
5	Чи важлива легкість розподілу людських ресурсів проєкту?	Так	Так	Так	Ні	Ні
6	Чи приймає команда розробників оцінки, перевірки, стадії розробки?	Так	Ні	Так	Так	Так

Таблиця 4

Вибір моделі життєвого циклу на основі характеристик колективу користувачів

№	Умови використання (вимоги)	МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ ПЗ				
		Каскадні	RAD	Інкрементні	Швидкого прототипування	Еволюційні
1	Чи буде присутність користувачів обмежена у ЖЦ розробки?	Так	Ні	Так	Ні	Так
2	Чи оцінюватимуть користувачі поточний стан ПЗ у процесі розробки?	Ні	Ні	Так	Так	Так
3	Чи будуть користувачі залучені	Ні	Так	Ні	Так	Ні

№	Умови використання (вимоги)	МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ ПЗ				
		Каскадні	RAD	Інкрементні	Швидкого прототипування	Еволюційні
	до всіх фаз ЖЦ?					
4	Чи Замовник відстежуватиме хід виконання проєкту?	Ні	Ні	Ні	Так	Так

Таблиця 5

Вибір моделі життєвого циклу на основі характеристик типу проєкту та ризиків

№	Умови використання (вимоги)	МОДЕЛІ ЖИТТЄВОГО ЦИКЛУ ПЗ				
		Каскадні	RAD	Інкрементні	Швидкого прототипування	Еволюційні
1	Чи розробляється у проєкті продукт нового для організації напряму?	Ні	Ні	Так	Так	Так
2	Чи буде проєкт розширенням існуючої системи?	Так	Так	Так	Ні	Ні
3	Чи буде проєкт крупно- чи середньомасштабним?	Ні	Ні	Так	Так	Так
4	Чи очікується тривала експлуатація ПЗ?	Так	Ні	Так	Ні	Так
5	Чи потрібний високий рівень надійності ПЗ?	Ні	Ні	Так	Ні	Так
6	Чи передбачається еволюція ПЗ протягом ЖЦ?	Ні	Ні	Так	Так	Так
7	Чи велика ймовірність зміни ПЗ на етапі супроводу?	Ні	Ні	Так	Так	Так
8	Чи є графік стислим?	Ні	Так	Так	Так	Так
9	Чи передбачається повторне використання компонентів?	Ні	Так	Так	Так	Так
10	Чи є достатні ресурси (час, гроші, інструменти, персонал)?	Ні	Ні	Ні	Так	Так

Процедура вибору моделі ЖЦ ПС SQI базується на застосуванні чотирьох таблиць питань, які відповідають запропонованій класифікації проєктів. По кожному з питань необхідно виділити відповіді, що відповідають конкретному проєкту, та вибрати модель із найбільшою кількістю позитивних відповідей.

Проведений аналіз підтверджує те, що моделі ЖЦ ПЗ можуть використовуватись для: планування; розподілення ресурсів (затрат праці і часу) керування проєктом розробки; організації взаємодії Замовника і Виконавця – визначення складу робочих документів, які розробляються на кожній стадії; аналізу і оцінювання розподілу ресурсів і затрат протягом усього життєвого циклу; проведення емпіричних досліджень з метою визначення впливу моделей на ефективність розробки і загальну якість програмного продукту.

Порядок організації проєкту щодо створення СПЗ військового призначення має містити наступні кроки.

1. Оцінка розміру проєкту та обсяг робіт, що потрібно виконати. Це можна зробити за допомогою методів оцінки розміру проєкту, таких як метод *COCOMO* [9] або із застосуванням підходу, який наведено у [10].

Довідка *COCOMO* (COnstructive COst MOdel) – це алгоритмічна модель оцінки вартості розробки програмного забезпечення, розроблена Баррі Боемом. Модель використовує просту формулу регресії з параметрами, визначеними з даних, зібраних із низки проєктів.

У методиці розрахунку трудомісткості робіт із розробки програмного забезпечення [10] використовують формулу:

$$T_{ПЗ} = T_{оп} + T_{досл} + T_{бс} + T_{прог} + T_{налаг} + T_{док},$$

де $T_{ПЗ}$ – трудомісткість створення програмного забезпечення (людино-години);

$T_{оп}$ – витрати, необхідні опису завдання;

$T_{досл}$ – витрати, необхідні для дослідження галузі розробки;

$T_{бс}$ – витрати, необхідні опису блок-схеми;

$T_{прог}$ – витрати на процедуру програмування;

$T_{налаг}$ – витрати на налагодження ПЗ;

$T_{док}$ – витрати на відпрацювання документації.

2. Визначення терміну виконання проєкту та наявного ресурсу.

3. Визначення вимог до якості програмного продукту. Це можуть бути

функціональні та не функціональні вимоги, такі як швидкість роботи, безпека, масштабованість та ін.

Враховуючи зовнішньополітичні умови, оцінювання повинно враховувати, що до СПЗ АСУ військового відомства висуваються більш суворі вимоги, зокрема щодо точності, надійності, вірогідності ризиків. СПЗ має бути максимально стійким не лише до негативних чинників, пов'язаних з програмним середовищем або діями користувачів, але й функціонувати за таких несприятливих умов, як:

низька якість зв'язку або його тимчасова відсутність, що зумовлено фізичним та радіоелектронним впливом противника, особливістю рельєфу місцевості, виконанням заходів безпеки операцій (приховування);

кібернетичний вплив на АСУ;

застосування СПЗ на транспортному засобі під час руху;

функціонування на технічних (транспортних) засобах зі слабким зарядом акумуляторних батарей;

недостатня (у сутінках) або навпаки надмірна (за ясної погоди) освітленість екрану технічного засобу, де встановлене СПЗ.

Крім того, інтерфейс СПЗ має враховувати потребу зменшення демаскувальних властивостей екрану у темну пору доби.

Зазначенні та інші аспекти має враховуватися при плануванні тестування та практичному його проведенні.

4. Аналіз моделей ЖЦ ПЗ. Кожна з них має свої переваги та недоліки, тому важливо вибрати ту, що найбільше відповідає потребам проєкту. Підхід до вибору моделі ЖЦ ПЗ наведено у даній статті. Проте питання, які наведені у Табл. 2–5 можна доповнити, виходячи з особливостей функціонування ІС військового призначення. Крім того процедуру аналізу за Табл. 2–5 доцільно автоматизувати, що є предметом окремих досліджень.

5. Оцінка можливих ризиків та проблем, які можуть виникнути при використанні певної моделі. Наприклад, деякі моделі можуть бути менш ефективними для великих проєктів, тоді як інші можуть бути складними для управління. З метою зменшення ризиків і підвищення ефективності розробки, необхідно вжити наступні *заходи по управлінню ризиками*:

ідентифікація ризиків на початку процесу розробки – необхідно проаналізувати всі можливі ризики, які можуть виникнути під час процесу розробки, щоб уникнути більш складних проблем в майбутньому;

аналіз ризиків на кожному етапі ЖЦ (ітерації) – детальний аналіз з точки зору ризиків допоможе виявити можливі проблеми та відповідні заходи, щоб їх запобігти;

планування заходів з управління ризиками – резервні плани, зміни у розкладі, більш детальна специфікація вимог тощо.

моніторинг ризиків на кожному етапі для внесення змін в плани дій при необхідності;

планування проведення регулярних оглядів та аудитів з метою перевірки відповідності розробленого ПЗ вимогам та стандартам.

6. Забезпечення безпеки – необхідно враховувати вимоги до безпеки та захисту даних.

7. Забезпечення управління командою та комунікації між учасниками проєкту. Добре підібрана команда та ефективна комунікація можуть допомогти забезпечити успішне виконання проєкту незалежно від вибраної моделі життєвого циклу.

Висновок. У статті обґрунтовані рекомендації щодо порядку організації створення спеціального програмного забезпечення для інформаційних систем військового призначення з урахуванням умов ведення проєкту. Основна увага приділена процедурі вибору моделі життєвого циклу. Вибір моделі життєвого циклу програмного забезпечення є важливим етапом у створенні програмного продукту. Для визначення раціональної моделі необхідно врахувати низку факторів та зосередитися на розробленні продукту згідно з вибраною моделлю. Періодичне оцінювання прогресу та ризиків дасть змогу вносити необхідні зміни та забезпечити успішне завершення проєкту.

Викладений матеріал може бути використаний під час розроблення Методичних рекомендацій щодо ведення і супроводження проєктів інформатизації стосовно створення СПЗ для ІС військового призначення.

Перспективи подальших досліджень. Оскільки *метою використання* моделі життєвого циклу програмного забезпечення є створення ефективного, економічно вигідного і якісного програмного продукту у визначений термін, подальші дослідження доцільно зосередити на розробленні часткових методик, зокрема: оцінки якості програмного продукту, проведення його тестування, верифікації і валідації.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Стратегічний оборонний бюлетень України : затв. Указом Президента України від 17.09.2021 р. № 473/2021
2. Якість програмного забезпечення та тестування: базовий курс : навч. посіб. / за ред. С. Я. Крепич , І. Я. Співак ; для бакалаврів галузі знань 12 "Інформаційні технології" спеціальності 121 "Інженерія програмного забезпечення". Тернопіль : ФОП Паляниця В. А., 2020. 478с.
3. Розробка інформаційних ресурсів та систем : електронне навчальне видання : конспект лекцій / Л. С. Глоба, Т. М. Кот. Київ : НН ІТС НТУУ "КПІ", 2014. 320 с.
4. Сидоров М. О. Вступ до інженерії програмного забезпечення : курс лекцій. Київ : НАУ-друк, 2010. 112 с.
5. Бахтизин В. В., Глухова Л. А. Б30 Технология разработки программного обеспечения : учеб. пособ. Минск : БГУИР, 2010. 267 с.
6. Коцюба И. Ю., Чунаев А. В., Шиков А. Н. Методы оценки и измерения характеристик информационных систем : учеб. пособ. Санкт-Петербург : Университет ИТМО, 2015. 264 с.
7. ГОСТ Р ИСО/МЭК ТО 15271 – 2002. Информационная технология. Руководство по применению ГОСТ Р ИСО/МЭК 12207 (Процессы жизненного цикла программных средств). Введ. 2002–06–05. Москва : Изд-во стандартов, 2002.
8. Управление программными проектами: достижение оптимального качества при минимуме затрат / Р. Фатрелл, Д. Шафер, Л. Шафер. Москва : Вильямс, 2003.
9. Філіпчик А. А. Методи оцінки трудомісткості програмних проєктів за методикою COCOMO II. URL: http://www.konferenciaonline.org.ua/ua/article/id-76/ (дата звернення: 29.05.2023).
10. Терешина Н. П., Летягин В. Г., Шобанов А. В. Технико-экономическое обоснование вариантов создания совместного предприятия : методические указания к курсовой работе. Mjcrdf : МИИТ, 2001. 20 с.

Стаття надійшла до редакційної колегії 29.05.2023

The procedure of organizing the development of special software for military information systems

Annotation

One of the criteria for the high-quality work of information systems of the Armed Forces of Ukraine is the full automation of the processes of its functioning. This is especially important when you need to process huge data streams. The rapidly changing security environment requires a constant need for technological improvements. Therefore, the information systems of the Armed Forces of Ukraine should not lag behind in the relevance of the software used. The procedure for creating special software for military IS certainly has certain features when it is written. The substantiation of the procedure for creating special software for information systems that are used in the Armed Forces of Ukraine is relevant.

The purpose of the article is to substantiate recommendations on the procedure for creating special software for military information systems, taking into account the conditions of the project.

The article substantiates recommendations on the procedure for organizing the creation of special software for military information systems, taking into account the conditions of the project.

The main attention is paid to the procedure for choosing a life cycle model, as this is an important step in creating a quality software product. To determine a rational model, it is necessary to take into account a number of factors and focus on developing a product according to the chosen model. Periodic assessments of progress and risks will enable necessary changes to be made and ensure the successful completion of the project.

The presented material can be used in the development of Guidelines for the management and maintenance of informatization projects to create special software for military information systems.

Keywords: software; life cycle model; Information system; cascade, incremental, evolutionary strategies; verification; validation.