ЛЕКЦІЯ 12. БЕЗПЕРЕРВНЕ ПОЛІПШЕННЯ ПРОЦЕСІВ ВИКОНАННЯ ПРОГРАМНИХ ПРОЕКТІВ

План лекції:

1. Основні положення інноваційного менеджменту при виконанні програмних проектів.
2. Інновації у сфері програмного забезпечення і їх вплив на параметри «залізного трикутника».
3. Безперервне поліпшення виконання програмних проектів як основна ціль проектного менеджменту.
4. Забезпечення конкурентноздатності середовища розробки.
5. Управління сторонніми компонентами при реалізації програмних проектів.

Питання, що виносяться на самостійне вивчення студентом (2 год.):

1. Модель вдосконалення процесу розробки програмного забезпечення на основі шести етапів [7, С. 1001].

12.1. Основні положення інноваційного менеджменту при виконанні програмних проектів

Основоположником теорії дослідження інновацій вважається відомий економіст Й. Шумпетер, який першим у економічній літературі провів межу між термінами «нововведення» та «інновації». Він стверджував, що «задача підприємців – реформувати і революціонізувати спосіб виробництва шляхом впровадження нових винаходів, а в більш загальному розумінні – за рахунок використання нових технологій для виробництва нових товарів, проте новим методом, завдяки відкриттю нового джерела сировини чи нового ринку готової продукції – в тому числі й до реорганізації існуючої та створення нової галузі промисловості».

Й. Шумпетер відзначав, що виробництво передбачає комбінування існуючих речей і сил, а виробництво чогось іншого чи іншим чином означає

створення нових комбінацій із речей та сил і, відповідно, виділив п’ять випадків здійснення нових комбінацій:

- виробництво нового, тобто ще невідомого споживачам, блага чи створення нової якості того чи іншого блага;
- впровадження нового, тобто ще практично невідомого методу (способу) виробництва, в основі якого не обов’язково лежить нове наукове відкриття і який може полягати у новому способі комерційного використання існуючого товару;
- отримання нового джерела сировини чи полуфабрикатів, незалежно від того, чи існувало це джерело раніше, чи просто не приймалося до уваги, не було доступним, чи його ще необхідно було створити;
- проведення відповідної реорганізації, наприклад, створення монопольного становища (шляхом створення тресту) чи підрив монопольного становища іншого підприємства;
- освоєння нового ринку збуту, тобто такого ринку, на якому дана галузь промисловості даної країни ще не була представлена, незалежно від того, існував цей ринок чи ні.

Інноваційний процес розглядається як «комплексний процес створення, розповсюдження і використання нового практичного засобу (нововведення) для нової (чи для кращого задоволення вже відомої) суспільної потреби, одночасно цей процес пов’язаний з впровадженням нових змін в тій соціальній і суспільній сфері, в якій здійснюється його життєвий цикл».

Подібне визначення інновацій і інноваційного процесу є досить широким і фактично охоплює будь-які зміни, спрямовані на покращення способу здійснення діяльності з задоволення потреб споживачів, що ускладнює уточнення предмету вивчення саме економічної науки.

Відповідно до міжнародних стандартів інновація визначається як «кінцевий результат інноваційної діяльності, що отримав втілення у вигляді нового чи вдосконаленого технологічного процесу, що використовується в практичній діяльності, чи у новому підході до соціальних послуг». Важлива

риса цього підходу до визначення інновацій полягає в тому, що акцент зроблено на фразі «кінцевий результат інноваційної діяльності», тобто зазначається, що впровадженню інновацій передує цілеспрямована діяльність, яка може охоплювати питання розробки чи пошуку нововведення і визначення економічної доцільності його впровадження у практику.

Інші підходи до визначення інновацій звертають увагу саме на основне джерело їх походження – науково-технічні розробки, зокрема інновація визначається як «використання результатів наукових досліджень і розробок, спрямованих на вдосконалення процесу діяльності виробництва, економічних, правових та соціальних відносин в області науки, культури, освіти і в інших сферах діяльності суспільства» чи «нововведення, пов'язане з науково-технічним прогресом (НТП) і, що полягає у відновленні основних фондів і технологій, в удосконаленні управління й економіки підприємства».

Ф. Янсен, аналізуючи ідеї Й. Шумпетера, запропонував визначення, згідно з яким інноваціями є «одночасний прояв двох світів, зокрема світу техніки і світу бізнесу. Коли зміни відбуваються лише на рівні технології, Шумпетер називає їх нововведенням. И лише тоді, коли до змін підключається бізнес, вони стають інноваціями».

Управління інноваціями (інноваційний менеджмент) визначається як комплекс формальних та неформальних правил, принципів, норм, установок та ціннісних орієнтацій, що регулюють різні сфери інноваційної діяльності.

У галузі розробки ПЗ, яка є інноваційною за своєю природою, оскільки передбачає створення нового продукту, інноваційний менеджмент зосереджений на питаннях вдосконалення технологій і способів створення програмних продуктів, зокрема, це стосується середовища розробки, тобто програмного інструментарію та компонентів, які використовуються.

12.2. Інновації у сфері програмного забезпечення і їх вплив на параметри «залізного трикутника»

В цілому галузь розробки програмного забезпечення характеризується надзвичайно високою інтенсивністю науково-дослідницьких розробок (НДР).

Зокрема, згідно з проведеним аналізом показників 123-х глобальних компаній світу різних галузей промисловості та сфери послуг станом на 1998 р., галузь розробки програмного забезпечення разом з фармацевтикою займають лідируючі позиції за показником витрат на НДР по відношенню до обсягу продаж (12,1% і 12,6% відповідно), майже вдвічі перевищуючи за цим показником галузь, що йде наступною у рейтингу.

Висока інтенсивність НДР свідчить про високу інтенсивність інноваційної діяльності. На відміну від фармацевтики, де від початку наукових досліджень до виходу продукту на ринок можуть проходити десятки років, галузь розробки ПЗ відрізняється дуже коротким життєвим циклом продукту: рекомендований цикл зміни програмно-апаратної платформи становить близько 3-х років, відповідно цикл оновлення ПЗ має бути, як мінімум, не тривалішим за даний строк. З іншого боку, на практиці цей строк виявляється значно тривалішим, зокрема, окремі дослідження наводять середній строк сприйняття інновацій галуззю ПЗ в межах від 10 до 15 років, що свідчить про недостатній рівень інноваційного менеджменту у галузі ПЗ і вимагає суттєвого його підвищення.

Процесні інновації дозволяють перевести трикутник із одного стану в інший, змінивши положення однієї чи одночасно двох його вершин: ресурсів та процесу. Графічно ефект інновацій можна представити наступним чином (рис. 12.1).

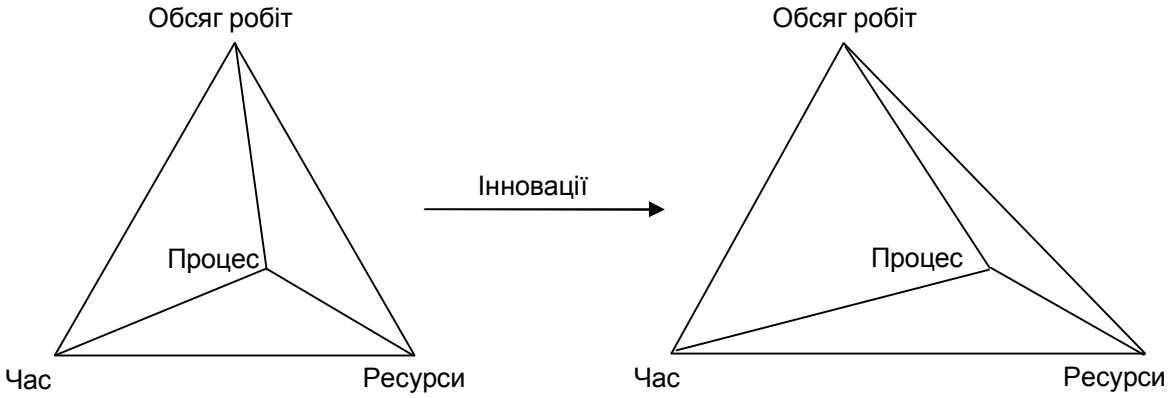

Рис. 12.1. Вплив процесних інновацій на зміну параметрів «залізного трикутника»

В результаті процесні інновації дозволяють змінити ті обмеження проекту з розробки ПЗ, які визначаються внутрішніми можливостями компанії-розробника.

С. Макконелл, порівнюючи сферу розробки ПЗ з іншими сферами підприємницької діяльності зазначає, що більшість не програмних проектів сфокусовані на оптимізації продуктових цілей, оскільки витрати власне на розробку є порівняно не важливими, однак велика вартість праці для проектів з розробки ПЗ переміщує фокус на проектні цілі, тобто на необхідність забезпечувати економічну доцільність розробки з урахуванням обмежень на якість і строки розробки продукту. Це твердження передбачає посилення уваги до вдосконалення внутрішніх процесів розробки ПЗ, відповідно зростає значення процесних інновацій по відношенню до продуктових.

12.3. Безперервне поліпшення виконання програмних проектів як основна ціль проектного менеджменту

Безперервне поліпшення виконання програмних проектів розглядається як основна ціль проектного менеджменту, повне досягнення якої можливе лише на високому рівні зрілості організації (наприклад, це основна характеристика найвищого рівня зрілості моделей SEI CMM/CMMI).

Підхід SPI (Software Process Improvement, Покращення процесу розробки ПЗ), сфокусований на питаннях покращення процесу розробки ПЗ, у тому числі й на основі інноваційних процесів, дуже часто базується на SW-CMM/CMMI. За оцінками фахівців вдале використання SPI може привести до отримання вигоди у співвідношенні з витратами від 2:1 до 10:1.

Інститут SEI розробив модель для практичної реалізації SPI, котра отримала назву IDEAL (Initiating, Diagnosing, Establishing, Acting & Learning). Модель IDEAL являє собою еталонну модель впровадження інновацій, яка дозволяє управляти процесами ініціації, планування та впровадження інновацій (рис. 12.2).

Рис. 12.2. Графічне представлення моделі IDEAL

Відповідно до рис. 12.2 модель IDEAL є циклічною і складається із п’яти фаз, які поділяються на окремі етапи.

Фаза «Ініціація» є початковою і містить наступні етапи:

– стимул до змін – внутрішня або зовнішня подія, що ініціює процес;
– задати контекст – визначити, яким чином інновація відповідає стратегії бізнесу організації;
– організувати підтримку – отримати підтримку зі сторони керівництва і відповідальних осіб як у вигляді ресурсів, так і управлінського впливу на виконавців процесу;
– зафрахтувати інфраструктуру – визначити і зафіксувати у вигляді письмової угоди чітко визначені очікування від впровадження інновації, вигоди та відповідальність.

Фаза «Діагностика» містить наступні етапи:

– охарактеризувати нинішній і цільовий стан – необхідно визначити нинішній стан проблеми і окреслити, що саме буде досягнуто при її вирішенні;
– розробити рекомендації – визначити ключові рекомендації щодо впровадження інновації. Рекомендації мають бути розроблені фахівцями предметної області і можуть бути використані керівництвом при прийнятті рішень.

Фаза «Створення» містить наступні етапи:

– встановити пріоритети – необхідно визначити пріоритети дій з урахуванням багатьох факторів, зокрема обмеженості ресурсів;
– розробити підхід – скласти стратегію виконання роботи з урахуванням доступних ресурсів. Мають бути розглянути технічні фактори (специфіка впровадження нової технології та необхідні для її використання знання і навики), а також нетехнічні (організаційна культура, ймовірні джерела опору, рівні підтримки та ринкові сили);
– спланувати дії – необхідно розробити детальний план, що має включати графік, завдання, віхи, точки прийняття рішень, ресурси, відповідальність, систему вимірювання досягнень, механізми

відслідковування, ризики та стратегії міграції та будь-які інші елементи, які можуть вимагатися організацією.

Фаза «Дії» містить наступні етапи:

– створити рішення – на основі поєднання усіх ключових елементів необхідно розробити рішення, що має відповідати потребам організації, визначеним на попередніх етапах. Рішення має бути комплексним і якомога більш детальним, враховувати усі необхідні дії і ресурси, у тому числі й передбачати зовнішню допомогу;
– перевірити рішення – рішення має бути перевірене, наприклад, з використанням пілотного проекту;
– уточнити рішення – на основі досвіду, отриманого на попередньому етапі необхідно внести корективи у рішення, якщо це необхідно;
– реалізувати рішення – на основі розробленого плану необхідно реалізувати рішення на практиці.

Фаза «Навчання» містить наступні етапи:

– аналіз і перевірка – зібрати і проаналізувати інформацію стосовно реалізованого рішення. Інформація має бути зібрана відповідально, отримані результати мають бути зіставлені з цілями, які ставилися на фазі ініціації і задокументовані;
– скласти пропозиції щодо подальших дій – зробити висновки і оформити їх у вигляді звіту, який може бути використаний керівництвом для прийняття рішень стосовно подальших дій.

Важливим досягненням моделі IDEAL є орієнтація на безперервність процесу і необхідність цілеспрямовано збирати інформацію про досвід реалізації інноваційного проекту, яка в подальшому може бути використана у наступних ітераціях чи окремих проектах.

12.4. Забезпечення конкурентноздатності середовища розробки

Статистика щодо випуску нових версій для 12 довільно обраних популярних бібліотек коду, які використовуються розробниками на

високорівневій мові програмування Borland Delphi, за період в один календарний рік (з 01.06.2003 р. по 01.06.2004 р.) представлена в табл. 12.1.

Відповідно до даних табл. 12.1, можна зробити висновок, що випуск нових версій для сторонніх бібліотек коду відбувається досить часто. Серед розглянутих бібліотек коду мінімальна кількість випусків нових версій становила три за рік, в той час як максимальна – двадцять дев'ять. Середнє значення в місяць на рівні 1,06 свідчить, що нові версії бібліотек коду в середньому виходять не рідше, ніж один раз на місяць.

В тому, що бібліотеки коду постійно обновляються, немає нічого негативного, скоріше навпаки: це свідчить про постійну роботу їх розробників над вдосконаленням бібліотек, підвищенням їх стабільності та функціональності. Саме для найпопулярніших бібліотек практика частих випусків нових версій є досить поширеною.

Таблиця 12.1

Вихід нових версій бібліотек коду, що використовуються розробниками на Borland Delphi за період з 01.06.2003 р. по 01.06.2004 р.

Назва бібліотеки коду	***Кількість нових версій***	
	всього за період	***в середньому в місяць***
Absolute Database	17	*1,42*
BusinessSkinForm	29	*2,42*
EurekaLog	12	*1*
FastReport VCL	3	*0,25*
FIBPlus	5	*0,42*
IBPhoenix OpenSource ODBC Driver	20	*1,67*
IntraWeb	19	*1,58*
KOL & MCK	17	*1,42*
PGP Components	3	*0,25*
SQLDirect	8	*0,67*
TPlanner	4	*0,33*
Unified Interbase	16	*1,33*
Середнє значення	***12,75***	***1,06***

Однак з погляду розробників програмного забезпечення постійний випуск нових версій використовуваних компонентів та інструментарію несе за собою наступні актуальні проблеми:

- по-перше, процес стеження за новими версіями потребує значних витрат часу, він вимагає періодичного ознайомлення з анонсами компаній-розробників бібліотек компонент та інструментарію, стрічками новин спеціалізованих сайтів та списків розсилки;
- по-друге, після того, як було помічено вихід нової версії сторонньої бібліотеки чи інструментарію, її необхідно отримати, що може бути пов’язане не лише зі значними витратами часу (посилання запиту на отримання нової версії, очікування ключів, доставка та ін.), а й з суттєвими матеріальними витратами (придбання нової версії, завантаження дорогими каналами зв’язку, пересилка поштою чи кур’єрською службою зі сплатою мита та митних зборів залежно від вимог законодавства);
- по-третє, чи не найскладнішою проблемою є процес переходу на нову версію, що пов’язаний з надзвичайно трудомісткими процесами забезпечення сумісності розробленого коду проекту з новими версіями бібліотек чи інструментарію, а також здійснення регресивного тестування проекту на предмет коректної поведінки з урахуванням поновлення версій бібліотек та інструментарію;
- по-четверте, необхідно забезпечувати сумісність для «унаслідуваних» проектів, тобто тих, для яких не відбувається розширення функціональності, а забезпечується лише супровід з виправленням виявлених помилок; для цих проектів досить часто взагалі здійснювати перехід на нові версії бібліотек та інструментарію робити економічно недоцільно, а тому розробники вимушені витрачати додатковий час на перехід з нових версій бібліотек та інструментарію на попередні для обслуговування даних

проектів, і повернення на актуальні версії для роботи з основними проектами;

– по-п’яте, досить часто виявляється, що нова версія містить критичні помилки які роблять її використання неможливим до того часу, доки вони не будуть виправлені її розробником, у даному разі слід здійснити процес повернення на попередню версію, який нерідко виявляється нетривіальним;
– по-шосте, якщо компанія-розробник програмного забезпечення однозначно прийняла рішення про поновлення версій використовуваних бібліотек та інструментарію, то слід здійснити даний процес дуже відповідально, виключивши випадки, коли деякі з розробників на робочих місцях мають різні версії сторонніх бібліотек чи інструментарію, в противному разі це може привести до помилок у проекті, джерело яких надзвичайно складно виявити.

Наведений перелік не є вичерпним, однак лише вказаних проблем достатньо, для того, щоб зрозуміти, що до такого, інноваційного за своєю природою процесу, як перехід на нові версії використовуваного інструментарію та бібліотек коду, слід відноситися з такою ж відповідальністю, як і до будь-якого іншого виду інноваційної діяльності.

Однак, зважаючи на свою відносну новизну, проблема управління впровадженням нових версій бібліотек та інструментарію поки що не має добре відомих прикладів успішного і ефективного вирішення, а тому компанії-розробники програмного забезпечення вимушені самостійно, на свій страх і ризик вживати заходи, які насправді дуже часто негативним чином впливають на успішність проектів з розробки програмного забезпечення.

Серед таких, на нашу думку, надзвичайно пагубних рішень слід виділити повну або часткову відмову від поновлення версій сторонніх бібліотек та інструментарію, що використовується. Якщо компанія обрала подібний шлях, то вона свідомо йде на сумнівний компроміс: незручності з

відмовою від проблем з поновленням версій компенсується економією часу та ресурсів на проведення подібних поновлень.

Проте на практиці виявляється, що подібне рішення найчастіше приймається адміністративно і дуже часто суперечить точці зору багатьох розробників, що працюють над проектом і вимушені безпосередньо стикатися з різноманітними проблемами попередніх версій, а тому природно зацікавлених у їх поновленні. Адміністративний тиск, спрямований лише на забезпечення виконання проекту в строк, без врахування актуальних для розробників проблем, однією з головних серед яких є забезпечення потреби працювати лише з сучасним інструментарієм та досконалими бібліотеками компонентів, деморалізує роботу колективу, призводить до виникнення конфліктів і в кінцевому випадку негативно позначається на успіху проекту.

Іноді ініціатива про відмову від поновлень версій йде безпосередньо від розробників, що працюють над проектом. У більшості випадків це свідчить про низький рівень організації праці у такому колективі, відсутність зацікавленості учасників проекту у кінцевому результаті, їх відмову від сприйняття інновацій, що взагалі неприпустимо для галузі розробки програмного забезпечення. Проект, що розробляється з такої позиції не може бути успішним.

У будь-якому разі відмова від використання сучасного інструментарію та бібліотек коду призводить до зниження якості кінцевого продукту, його низької конкурентноздатності. А тому подібний шлях, на нашу думку, є однозначно неприйнятним.

12.5. Управління сторонніми компонентами при реалізації програмних проектів

Управління впливом сторонніх компонентів на процес розробки ПЗ має здійснюватися у рамках єдиної системи управління інноваціями, що має бути органічно інтегрованою у систему управління конфігураціями.

На рис. 12.3 представлено схематичне зображення основних складових системи управління впливом сторонніх компонентів на процес розробки ПЗ.

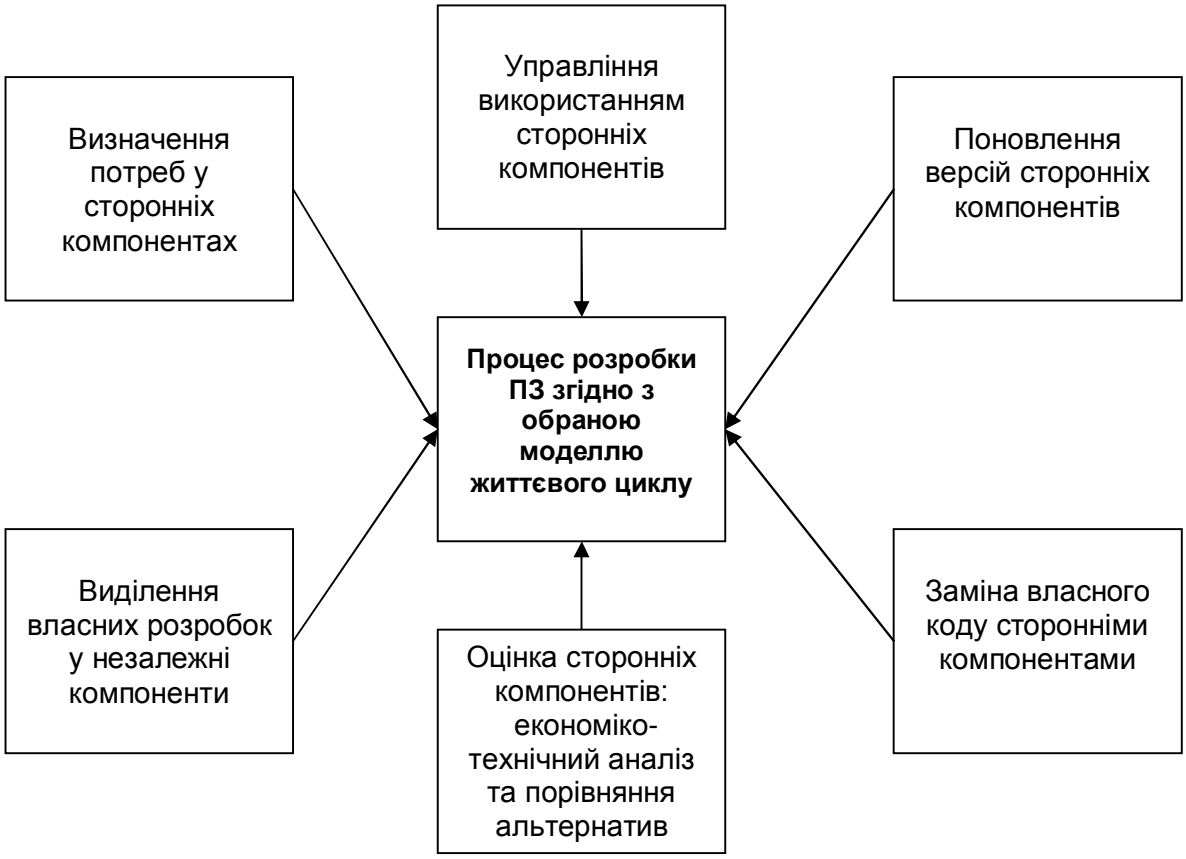

Рис. 12.3. Основні складові системи управління впливом сторонніх компонентів на процес розробки ПЗ

Відповідно до рис. 2.7, на нашу думку, доцільно виділити такі складові системи управління впливом сторонніх компонентів на процес розробки ПЗ:

– управління використанням сторонніх компонентів – має передбачати комплекс заходів, спрямованих на популяризацію використання і забезпечення ефективності використання сторонніх компонентів, що може бути здійснено як за рахунок навчання персоналу, ознайомлення його зі структурою та особливостями використання бібліотек коду, так і за рахунок певних організаційних заходів, що обмежують створення розробниками власних рішень, якщо для цього можна використати готові бібліотеки коду. Подібним заходам має бути приділена особлива увага, оскільки розробники як по причині недостатньої інформованості стосовно

особливостей використання готових бібліотек коду, так і по причині недостатньої дисциплінованості, намагаються уникати готових рішень, розроблюючи власні. Звичайно подібна практика призводить до збільшення витрат на розробку і супровід продукту, оскільки необґрунтовано зростає його розмір. Крім того, самостійна розробка рішення замість використання готового коду звичайно призводить до необхідності більше ресурсів витрачати на тестування подібного рішення у порівнянні з уже розробленим і, відповідно, перевіреним;

– визначення потреб у сторонніх компонентах – у процесі розробки програмного продукту, особливо на ранніх етапах життєвого циклу, необхідно періодично здійснювати ідентифікацію потреб у сторонніх компонентах та бібліотеках готового коду. Своєчасне виявлення можливості для відмови від власних розробок на користь уже готових компонентів може суттєво скоротити як вартість, так і тривалість розробки, а також у більшості випадків може привести до підвищення якості готового продукту;
– виділення власних розробок у незалежні компоненти – задача виділення власних розробок у незалежні компоненти має розглядатися як з позицій підвищення долі повторного використання програмного коду, так і з погляду виявлення нових джерел доходів за рахунок розповсюдження створених бібліотек коду як окремих продуктів. Обидва напрямки дозволяють підвищити дохідність капіталовкладень у розробку проекту, у першому випадку це здійснюється за рахунок скорочення витрат на розробку за рахунок повторного використання створеного коду, у другому випадку – за рахунок зростання обсягу грошових потоків в результаті реалізації проекту;
– оцінка сторонніх компонентів: економіко-технічний аналіз та порівняння альтернатив – оцінку сторонніх компонентів та

здійснення порівняння з альтернативними варіантами необхідно провадити надзвичайно відповідально з урахуванням значної кількості важливих факторів як технічного, так і економічного характеру. Крім того, в процесі здійснення оцінки необхідно здійснювати юридичний аналіз ліцензій, на умовах яких пропонуються до використання сторонні бібліотеки коду, а також враховувати такі фактори, які складно піддаються кількісній оцінці, наприклад, перспективність технологій, рівень сприйняття бібліотек персоналом та ін.;

– заміна власного коду сторонніми компонентами – в процесі розробки необхідно періодично перевіряти створених програмний код на предмет доцільності його заміни сторонніми компонентами. Інспекція програмного коду з цією метою має здійснюватися постійно, оскільки готові бібліотеки стороннього коду могли з'явитися на ринку пізніше проведення початкового проектування програмного продукту і, відповідно, не були враховані;
– поновлення версій сторонніх компонентів – необхідно забезпечити прозорий і безперервний процес здійснення оцінки доцільності і переходу на нові версії сторонніх бібліотек компонентів. Слід відзначити, що сторонні компоненти і бібліотеки коду розвиваються незалежно від ходу проекту компанії-розробника, і у процесі розвитку можуть набувати нових якісних властивостей, які позивним чином здатні вплинути на характеристики продукту, що розробляється з їх використанням. Подібний розвиток звичайно здійснюється без суттєвих витрат зі сторони розробника – звичайно нові версії можна отримати безкоштовно, чи зі значними знижками, що, в свою чергу, є вагомим джерелом для підвищення дохідності капіталовкладень у процес розробки ПЗ.

Слід зазначити, що збільшуючи кількість сторонніх компонентів і бібліотек коду, які використовуються у проекті з розробки програмного

забезпечення, компанія-розробник в певній мірі ставить себе в залежність від постачальника сторонніх компонентів, що, в свою чергу, може привести до зростання ризиків проекту.

Однак, за умови виваженого підходу до підбору сторонніх компонентів, можна не лише уникнути невиправданого зростання сумарних ризиків проекту, а і досягти їх зменшення.