

**Лабораторна робота №1.1**

з дисципліни: «Супроводження програмного забезпечення»

на тему: «Технічні прийоми супроводження»

Виконав: студент ПІ-415

Воловін Є. О.

Прийняла: викладач

Корнієнко С. П.

# Зміст

1. Вступ…………………………………………………………………..……3
2. Визначення та термінологія………………………………….…….....…...5
3. Життє́вий цикл програ́много забезпе́чення…..……………………….….6
4. Каскадна модель…………………………...………………………...….….6
5. Спіральна модель………………….…………………………….……..…..7
6. Еволюційна модель.…………………………………………………..……8
7. Література…………………………………………………………...….…11

**Вступ**

**Супроводження програмного забезпечення** — процес покращення, оптимізації та виправлення дефектів у програмному забезпеченні після його [вводу до експлуатації](https://uk.wikipedia.org/wiki/%D0%92%D0%BF%D1%80%D0%BE%D0%B2%D0%B0%D0%B4%D0%B6%D0%B5%D0%BD%D0%BD%D1%8F_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%B7%D0%B0%D0%B1%D0%B5%D0%B7%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%BD%D1%8F).

Цей процес стандартизовано організацією ISO — [ISO/IEC 14764](https://uk.wikipedia.org/w/index.php?title=ISO/IEC_14764&action=edit&redlink=1).

Область знань «Супровід ПЗ (Software Maintenance)» складається з таких розділів:

* основні концепції (Basic Concepts),
* процес супроводження (Process Maintenance),
* ключові питання супроводу ПЗ (key Issue in Software Maintenance) ,
* техніки супроводу (Techniques for Maintenance).

Супровід розглядається з точки зору задоволення вимог споживача у готовому ПЗ, коректності його виконання, процесів навчання й оперативного обліку його процесу.

Основні концепції – це базові визначення і термінологія, підходи до еволюції і супроводу ПЗ, до оцінки вартості супроводу тощо. До основних концепцій можна віднести ЖЦ ПЗ (стандарт ISO/IEC 12207) і складання документації. Головне призначення цієї області знань полягає у виконанні готової програмної системи, фіксації помилок, що виникають при виконанні, дослідженні їх причин, аналізі необхідності модифікації системи з метою усунення помилок, оцінці вартості робіт із проведення змін функцій і системи в цілому. Розглядаються проблеми, пов'язані з ускладненістю продукту при великій кількості змін, і методи її подолання.

Процес супроводження містить у собі моделі процесу супроводу і планування діяльності людей, що проводять запуск ПЗ, перевірку правильності його виконання і внесення в нього змін. Цей процес згідно з стандартом ISO/IEC 14764 проводиться шляхом:

– коригування, вдосконалення продукту для усунення виявлених помилок або нереалізованих задач;

– адаптація, підлаштування продукту до умов експлуатації, що змінилися, або в новому середовищі виконання;

– поліпшення, еволюційна зміна продукту для підвищення продуктивності або рівня супроводу;

– перевірка ПЗ, пошук і виправлення помилок при експлуатації системи.

Ключові питання супроводу ПЗ – це управлінські, вимірювальні і вартісні. Суть управлінських питань – контроль ПЗ при модифікації й удосконалюванні функцій і недопущення зниження продуктивності системи. Питання вимірювання пов'язане з оцінкою характеристик системи після її модифікації, а також повторного тестування для оцінки показників якості. Вартісні питання пов'язані з оцінкою витрат на супровід залежно від його типу, кваліфікації персоналу, платформи й ін.

Техніка супроводу (цей розділ називають також еволюцією ПЗ). Відомий фахівець в області ПЗ Дж. Леман (1970 р.) запропонував розглядати супровід як еволюційну розробку програмних систем, оскільки здана в експлуатацію система не завжди цілком завершена, її треба змінювати протягом терміну експлуатації. Внаслідок змін система стає більш складною і погано керованою. У зв'язку з цим виникає проблема зменшення її складності. До технологій еволюції ПЗ відносять реінженерію, реверсну інженерію і рефакторинг.

Реінженерія – це удосконалення застарілого ПЗ шляхом його реорганізації або реструктуризації, а також перепрограмування окремих елементів або настроювання параметрів на іншу платформу, середовище виконання зі збереженням зручності його супроводу.

[Реверсна (зворотня) інженерія](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B2%D0%B5%D1%80%D1%81%D0%BD%D0%B0_%D1%96%D0%BD%D0%B6%D0%B5%D0%BD%D0%B5%D1%80%D1%96%D1%8F) полягає у відновленні специфікації (графів викликів, потоків даних і ін.) за отриманим кодом системи для її аналізу на більш високому рівні. Відновлюється ідентифікація компонентів і зв'язків між ними для забезпечення перепрограмування системи на нову платформу. Найчастіше реверсна інженерія застосовується після того, як у код ПЗ було внесено багато змін і воно стало некерованим або змінилася платформа комп'ютера.

Рефакторинг – це реорганізація коду для поліпшення характеристик і показників якості об’єктно-орієнтованих і компонентних програм без зміни їх поведінки. Цей процес реалізується шляхом поступової зміни окремих операцій над текстами, інтерфейсами, середовищем програмування і виконання ПЗ, а також настроювання або внесення змін в інструментальні засоби підтримки ПЗ. Якщо при зміні зберігається формат існуючої системи, то рефакторинг – один з варіантів реверсної інженерії.

**Визначення та термінологія**

Супровід програмного забезпечення визначається стандартом IEEE Standard for Software Maintenance (IEEE 1219) як модифікація програмного продукту після передачі в експлуатацію для усунення збоїв, поліпшення показників продуктивності і / або інших характеристик (атрибутів) продукту, або адаптації продукту для використання в модифікованому оточенні. Цікаво, що даний стандарт також стосується питань підготовки до супроводу до передачі системи в експлуатацію, проте, структурно це зроблено на рівні відповідного інформаційного додатку, включеного в стандарт.

У свою чергу, стандарт життєвого циклу 12207 (IEEE, ISO / IEC, ДСТУ ISO / IEC) позиціонує супровід як один з головних процесів життєвого циклу. Цей стандарт описує супровід як процес модифікації програмного продукту в частині його коду і документації для вирішення виникаючих проблем <при експлуатації> або реалізації потреб поліпшень <тих чи інших характеристик продукту>. Завдання полягає в модифікації продукту за умови збереження його цілісності. Міжнародний стандарт ISO / IEC 14764 (Standard for Software Engineering - Software Maintenance) визначає супровід програмного забезпечення в тих же термінах, що і стандарт 12207, надаючи особливого значення робіт з підготовки до діяльності по супроводу до передачі системи в реальну експлуатацію, наприклад, питань планування регламентів і операцій із супроводу.

**Життє́вий цикл програ́много забезпе́чення**

Життє́вий цикл програ́много забезпе́чення — сукупність окремих етапів робіт, що проводяться у заданому порядку протягом періоду часу, який починається з вирішення питання про розроблення програмного забезпечення і закінчується припиненням використання програмного забезпечення.

Модель життєвого циклу - це структура, що складається із процесів, робіт та задач, які включають в себе розробку, експлуатацію і супровід програмного продукту; охоплює життя системи від визначення вимог до неї до припинення її використання. На сьогодні найбільшого розповсюдження набули дві моделі:

* каскадна модель;
* спіральна модель.

**Каскадна модель**

Однією з перших з’явилась каскадна модель, в якій кожен етап роботи виконується лише раз. На кожному етапі робота виконується настільки ретельно, щоб потреби повертатись до попереднього не виникало. Результат виконання кожного етапу, перед передачею в наступний, піддається верифікації.

**Спіральна модель**

Розробка ітераціями відображає об'єктивно існуючий спіральний цикл створення системи. Неповне завершення робіт на кожному етапі дозволяє переходити на наступний етап, не чекаючи повного завершення роботи на поточному. При ітеративному способі розробки відсутню роботу можна буде виконати на наступній ітерації. Головне ж завдання - щонайшвидше показати користувачам системи працездатний продукт, тим самим активізуючи процес уточнення і доповнення вимог.

Виходячи з можливості внесення змін, як в процес, так і в проміжний продукт було створено спіральну модель ЖЦ.

Внесення змін орієнтоване на задоволення потреби користувачів одразу, як тільки буде встановлено, що створені артефакти або елементи документації не відповідають дійсному стану розробки.

Дана модель ЖЦ допускає аналіз продукту на витку розробки, його перевірку, оцінку правильності та прийняття рішення про перехід на наступний виток або повернення на попередній виток для доопрацювання на ньому проміжного продукту.

Відмінність цієї моделі від каскадної полягає в можливості багато разів повертатися до процесу формулювання вимог і до повторної розробки версії системи з будь-якого процесу моделі.

Для програмного продукту така модель не дуже підходить з декількох причин. По-перше, висловлення вимог замовником носить суб'єктивний характер, вимоги можуть багаторазово уточнюватися протягом розробки ПС і навіть після завершення та випробовування, і часом може з'ясуватися, що замовник «хотів зовсім інше». По-друге, змінюються обставини та умови використання системи, тому загальновизнаним законом програмної інженерії є закон еволюції, який сформулюємо так: кожна діюча ПС з часом потребує внесення змін або виводиться з експлуатації.

При необхідності внесення змін до системи на кожному витку з метою отримання нової версії системи обов'язково вносяться зміни в заздалегідь зафіксовані вимоги, після чого повертаються на попередній виток спіралі для продовження реалізації нової версії системи з урахуванням усіх змін.

**Еволюційна модель**

У разі еволюційної моделі система послідовно розробляється з блоків конструкцій. На відміну від інкрементної моделі в еволюційній моделі вимоги встановлюються частково і уточнюються в кожному наступному проміжному блоці структури системи.

Використання еволюційної моделі припускає проведення дослідження предметної області для вивчення потреб її замовника і аналізу можливості застосування цієї моделі для реалізації. Модель використовується для розробки нескладних і некритичних систем, де головною вимогою є реалізація функцій системи. При цьому вимоги не можуть бути визначені відразу і повністю. Тому розробка системи здійснюється ітераційним шляхом її еволюційного розвитку з отриманням деякого варіанта системи–прототипу, на якому перевіряється реалізація вимог. Іншими словами, такий процес за своєю суттю є ітераційним, з етапами розробки, що повторюються, починаючи від змінених вимог і закінчуючи отриманням готового продукту. В деякому розумінні до цього типу моделі можна віднести спіральну модель.

Розвитком цієї моделі є модель еволюційного прототипування в рамках усього ЖЦ розробки ПС .

У літературі вона часто називається моделлю швидкої розробки програм RAD (Rapid Application Development).

У даній моделі наведені дії, які пов'язані з аналізом її застосовності для конкретного виду системи, а також обстеженням замовника для визначення потреб користувача при розробці плану створення прототипу.

У моделі є дві головні ітерації розробки функціонального прототипу, проектування і реалізації системи з метою перевірки, чи задовольняє вона всі функціональні і нефункціональні вимоги. Основною ідеєю цієї моделі є моделювання окремих функцій системи в прототипі і поступове еволюційне його доопрацювання до виконання всіх заданих функціональних вимог.

Ітерацій з отримання проміжних варіантів прототипу може бути декілька, в кожній з яких додається функція і повторно моделюється робота прототипу. І так до тих пір, поки не будуть промодельовані всі функції, задані у вимогах до системи. Після цього виконується ще одна ітерація – остаточне програмування для отримання готової системи.

Ця модель застосовується для систем, в яких найбільш важливими є функціональні можливості, і які необхідно швидко продемонструвати на CASE-засобах.

Оскільки проміжні прототипи системи відповідають реалізації деяких функціональних вимог, їх можна перевіряти і під час супроводу і експлуатації, тобто разом з процесом розробки чергових прототипів системи. При цьому допоміжні і організаційні процеси можуть виконуватися разом з процесом розроблення і накопичувати відомості за даними кількісних і якісних оцінок на процесах розроблення.

При цьому враховуються такі чинники ризику:

– реалізація всіх функцій системи одночасно може призвести до громіздкості;

– обмежені людські ресурси зайняті розробкою протягом тривалого часу.

Переваги застосування даної моделі ЖЦ такі:

– швидка реалізація деяких функціональних можливостей системи і їх апробація;

– використання проміжного продукту в наступному прототипі;

– виділення окремих функціональних частин для реалізації їх у вигляді прототипу;

– можливість збільшення фінансування системи;

– зворотний зв'язок із замовником для уточнення функціональних вимог;

– спрощення внесення змін у зв'язку із заміною окремих функції.

Модель розвивається у напряму додавання нефункціональних вимог до системи, пов'язаних із захистом і безпекою даних, несанкціонованим доступом до них і ін.

**Література**

1. <http://webstore.iec.ch/Webstore/webstore.nsf/ArtNum_PK/41007!opendocument&preview=1>
2. Guide to the Software Engineering Body of Knowledge (SWEBOK). – California: IEEE Computer Society, 2001. – 219 p.
3. Grubb Penny. Software Maintenance: Concepts and Practice (2nd Edition) / [Penny Grubb, Takang A. Armstrong]. – Singapore: World Scientific, 2003. – 349 p.
4. Pigosky M. Thomas. Practical Software Maintenance – Best Practices for Managing Your Software Investment / Thomas M. Pigosky. – Canada: Wiley Computer Publishing, 1997. – 228 p.