Лекція 4
Моделі життєвого циклу програмних систем

Життєвий цикл ПС визначається як «весь період існування системи від початку розробки до завершення її використання» (ДСТУ 2941-94. Розробка систем. Терміни і визначення)

ЖЦ поділяється на впорядковані стадії, основні з яких:

- Визначення потреб
- Аналіз вимог і оформлення концепції
- Розробка
- Виробництво
- Впровадження/продаж
- Експлуатація
- Супровід і підтримка
- Вилучення з експлуатації

Основні процеси ЖЦ, пов’язані з процесом розробки ПС:

- Аналіз вимог
- Проектування (попереднє і детальне)
- Реалізація
- Тестування

Найбільш відомі типи моделей ЖЦ:

- Послідовні
- Ітераційні

Призначення моделей розробки

Моделі ЖЦ можуть використовуватись для:

- Організації, планування. Розподілення ресурсів (затрат праці і часу) і керування проектом розробки
- Організації взаємодії з замовниками і визначення складу документів (робочих продуктів), які розроблюються на кожній стадії
- Аналізу і оцінювання розподілу ресурсів і затрат на протязі ЖЦ
- Наглядного опису або в якості основи для проведення фінансових розрахунків з замовниками
- Проведення емпіричних досліджень з метою визначення впливу моделей на ефективність розробки і загальну якість програмного продукту

ISO/IEC 12207 (Guide for ISO/IEC 12207 – Software life cycle processes).

Моделі послідовного виконання стадій

1. **Каскадна модель**

2. Каскадна модель із зворотнім зв’язком

Ця модель розширює стандартну модель включенням в неї циклів зворотного зв’язку для повернення на попередню стадію при зміні вимог, проекту і по результатам інспекцій або дій по V&V

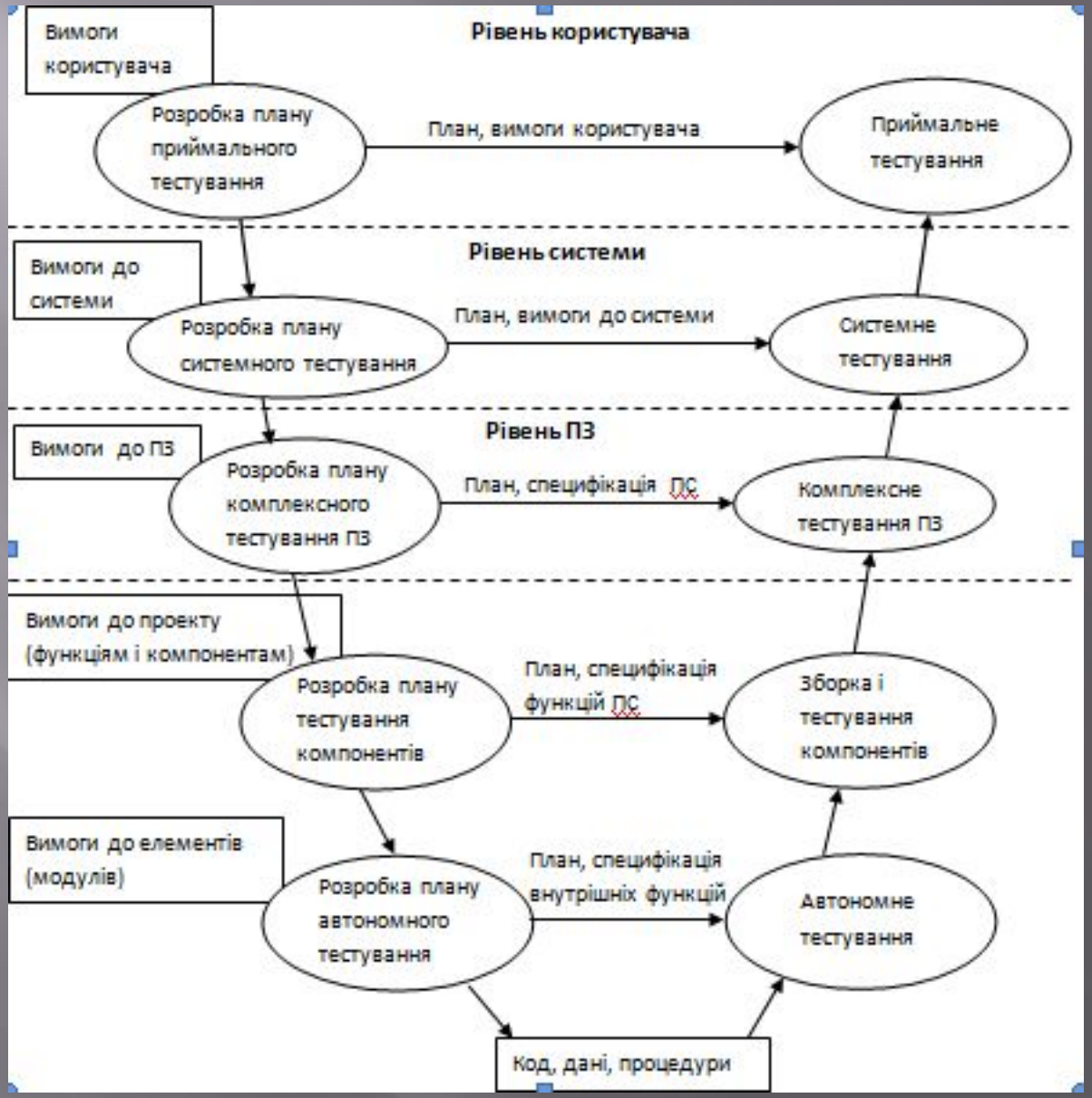
Характеристики каскадної моделі:

- Послідовне впорядкування стадій
- Формальні перевірки по завершенні кожної стадій (інспекції, технічні огляди)
- Наявність документованих вимог і проекту

Переваги каскадної моделі:

- Застосування формальних перевірок дозволяє вчасно виявляти дефекти
- Чіткі критерії початку і завершення стадій
- Чіткі вимоги і цілі проекту

3. V-подібна модель



3. V-подібна модель

В цій моделі тестування розглядається як неперервний процес, інтегрований в процес розробки ПС. Він включає два взаємопов’язаних підпроцесіса

- планування тестування в рамках процесів розробки системи (ліва гілка)
- проведення тестування відповідних об’єктів (права гілка)

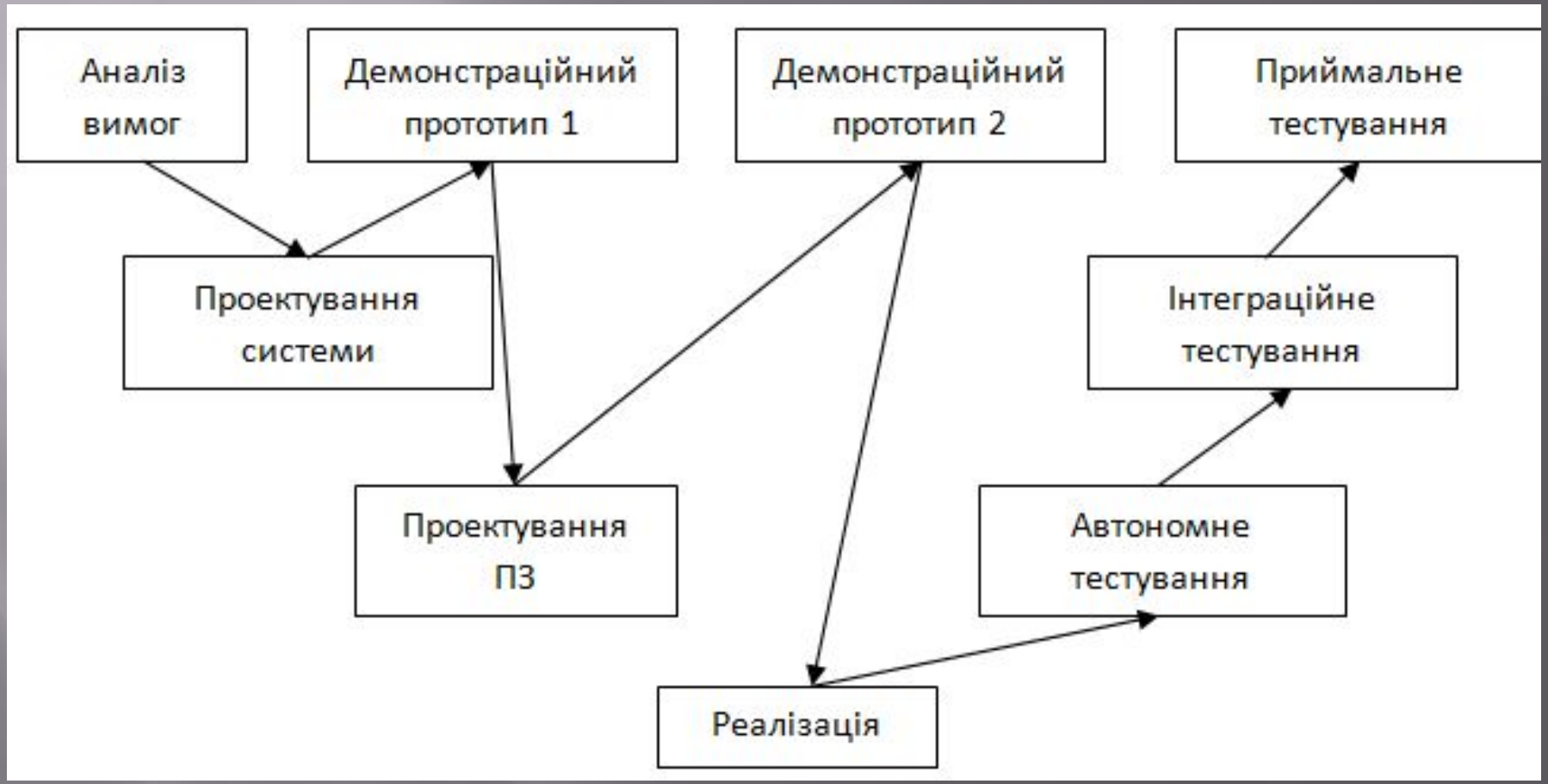
Характеристики V-подібної моделі:

- Перевірка і оцінка тестопридатності вимог на ранніх стадіях розробки (з допомогою аналізу, який виконується під час тестування)
- Наявність документованих тестових вимог

Переваги V-подібної моделі:

- Забезпечує зворотний зв’язок з користувачем на ранніх стадіях ЖЦ
- Покращує планування і розподіл затрат на тестування
- Чіткі документовані цілі тестування

4. Каскадна модель з прототипуванням (пилоподібна модель)



Прототипи слугують для демонстрації і після розробки проекту їх викидують, а реалізація проекту може виконуватись в іншому середовищі.

Характеристика

- для аналізу і моделювання проектних рішень застосовуються прототипи

Переваги

- усуває проблеми, пов’язані з неповнотою і нечіткістю вимог

Ризики застосування послідовних моделей:

- Вимоги не повністю зрозумілі
- Система занадто велика, щоб бути реалізованою одразу
- Швидкі зміні в технологіях
- Часта зміна вимог
- Користувач не може використовувати проміжні результати

Коли краще застосувати послідовну модель:

- Вимоги зрозумілі і не будуть суттєво мінятись
- Система має невеликий розмір і складність
- Усі можливості мають бути реалізовані одразу
- Нова система розробляється на заміну старої і необхідно повністю замінити стару систему

Ітераційні моделі

Ітераційні моделі загалом можна розділити на два класи:

- моделі з приростом (Incremental)
- еволюційні (Evolutionary)

У відповідності з цими моделями програмний продукт розроблюється ітераціями, і кожна ітерація закінчується випуском працездатної версії програмного продукту

1. **Ітераційні моделі з приростом**

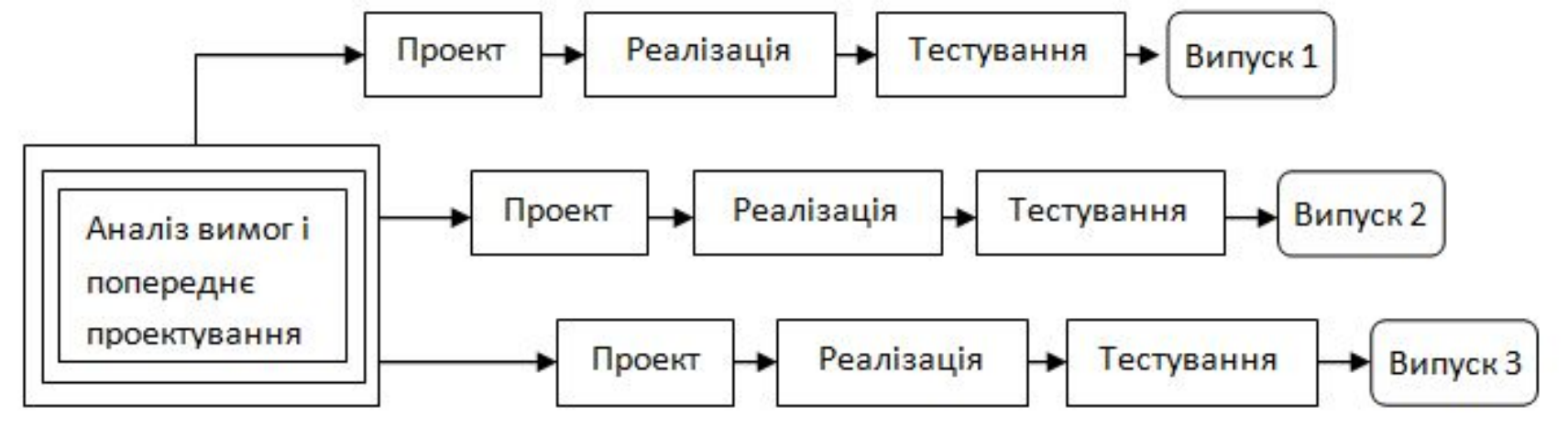
- спочатку визначаються усі вимоги до ПС, і можливо розроблюється попередній проект.

Подальша розробка ПС розбивається на ітерації.

- В першій ітерації реалізується набір основних вимог, які забезпечують базову функціональність.
- Інші ітерації реалізуються в порядку критичності вимог для кінцевого користувача

Ітераційна модель з перекриттям ітерацій

Ітераційні моделі з приростом широко використовуються для розробки комерційних програмних продуктів, які розвиваються на протязі довгого періоду часу або для яких зовнішні вимоги змінюються слабо.



Характеристики ітераційних моделей з приростом:

- Аналіз і проектування виконуються для усієї системи
- Базові функціональні вимоги реалізуються першими
- Інші вимоги реалізуються в наступних версіях
- Проміжні версії придатні для використання

Переваги ітераційних моделей з приростом:

- Критичні функції реалізуються в першу чергу
- Критичні функції тестуються більш ретельно
- Найменш критичні задачі реалізуються останніми, що мінімізує наслідки відмови із-за дефектів
- Завершення першої версії остаточно затверджує вимоги і проект
- Раннє планування виконання тестування
- Раннє виявлення дефектів користувачами

Ризики, пов’язані з вибором моделі:

- Вимоги не повністю зрозумілі
- Вимоги не стабільні
- Усі можливості мають бути реалізовані одразу
- Швидкі зміни в технології

Коли краще застосовувати модель:

- Вимагається швидка реалізація основних можливостей
- Якщо проект системи можна природним чином поділити на незалежні частини

2. Еволюційні моделі. Спіральна модель

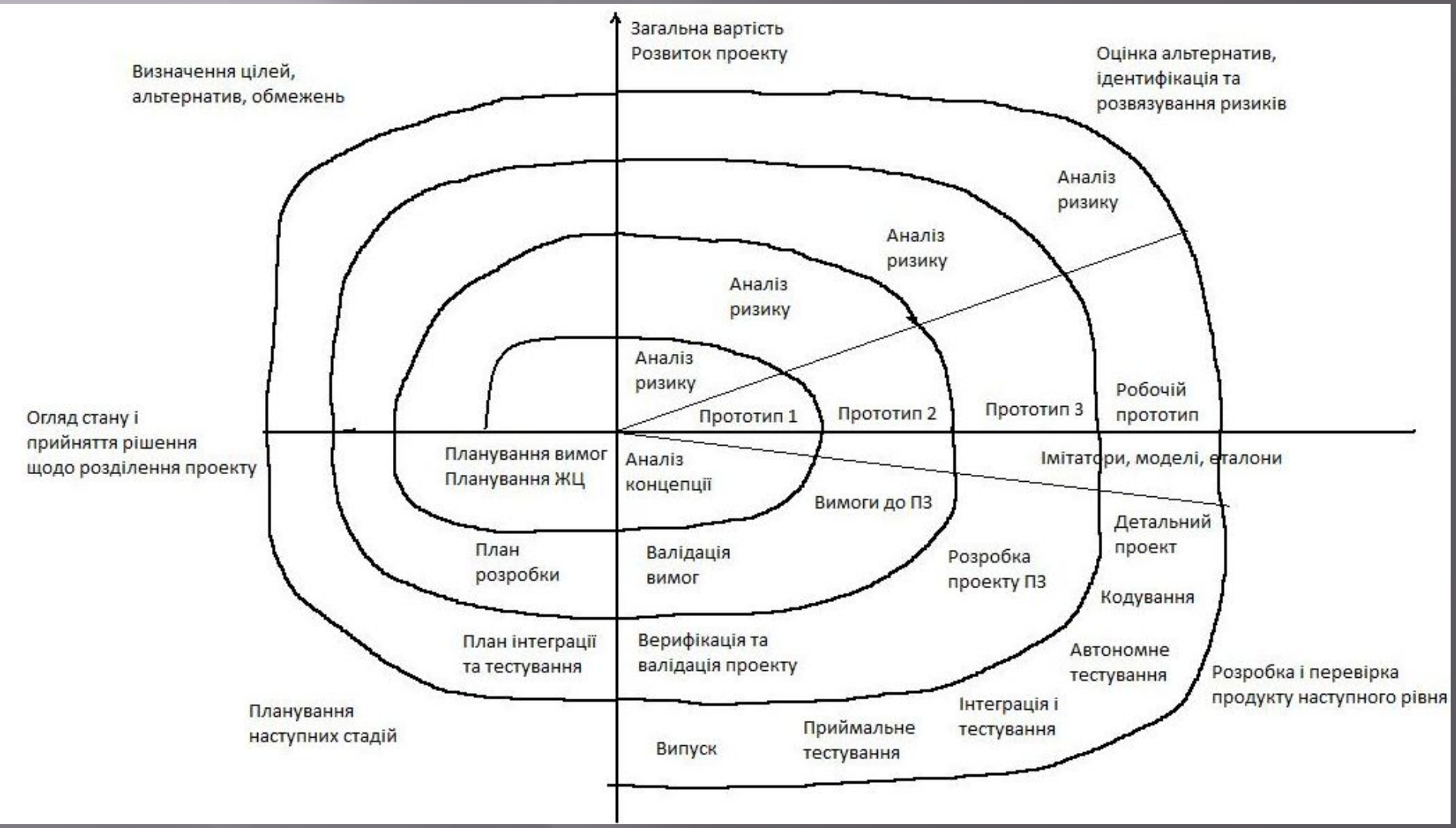
Розроблена Боемом. Відображає керований ризиком процес еволюції проекту від аналізу до готовності продукту.

На кожному витку спіралі (стадії) виконуються наступні дії:

1. Визначаються цілі стадії. Розглядаються альтернативні рішення для досягнення цих цілей
2. Проводиться оцінювання цих рішень. Ідентифікуються ризики завершення стадії і виконується їх аналіз. Приймаються рішення про продовження або завершення стадії
3. Розробляються робочі продукти стадії та план для наступної стадії
4. Останній виток спіралі може мати структуру каскадної моделі.

Види розглядуваних ризиків – ризики, які стосуються технічних аспектів розробки, фінансові ризики, ризики експлуатації.

Спіральна модель застосовується для складних проектів або


Загальна вартість
Розвиток проекту
Визначення цілей, альтернатив, обмежень
Оцінка альтернатив, ідентифікація та розвязування ризиків
Аналіз ризику
Аналіз ризику
Аналіз ризику
Аналіз ризику
Прототип 1
Прототип 2
Прототип 3
Робочій прототип
Огляд стану і прийняття рішення щодо розділення проекту
Планування вимог
Планування ЖЦ
Аналіз концепції
Імітатори, моделі, еталони
Вимоги до ПЗ
Детальний проект
План розробки
Валідація вимог
Розробка проекту ПЗ
Кодування
План інтеграції та тестування
Верифікація та валідація проекту
Автономне тестування
Розробка і перевірка продукту наступного рівня
Планування наступних стадій
Випуск
Приймальне тестування
Інтеграція і тестування

Характеристики спіральної моделі:

- Перший прототип моделює концепцію. Результатом є план вимог. Перед переходом до розробки наступного прототипу виконується аналіз ризику
- Другий прототип моделює вимоги до ПЗ. Результатом є план розробки. Виконується аналіз ризику
- Третій прототип моделює проект. В результаті створюється інтегрований і протестований прототип. Виконується аналіз ризику.
- Останній прототип (робочій) використовується як основа для детального проектування, кодування та тестування.

Ризики пов’язані з вибором моделі:

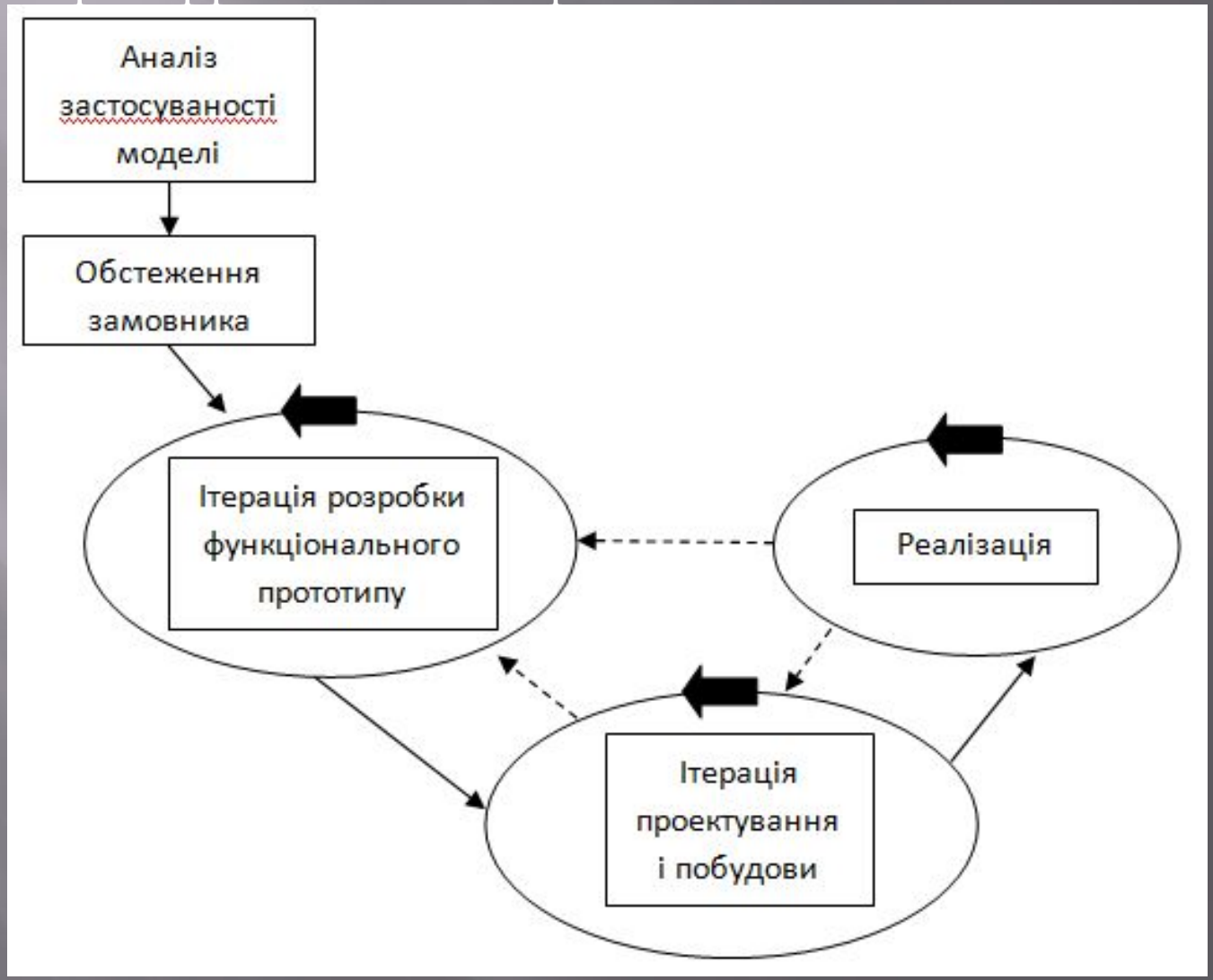
- Усі можливості мають бути реалізовані одразу
- Проект неможна природним чином розділити на незалежні частини

Коли краще застосовувати модель:

- Проект крупний, складний і вимоги не можуть бути визначені одразу
- Нова технологія і вимагається її вивчення
- Користувачі не можуть чітко сформулювати вимоги
- Вимагається рання демонстрація можливостей

Модель еволюційного прототипування.

RAD – Rapid application development



Характеристики моделі:

- Гнучкість. Можливість швидко реагувати на зміни і розширення вимог
- Пріоритети функціональних характеристик перед технічними (якості)

Переваги моделі:

- Раннє виявлення дефектів в інтерфейсі
- Швидка демонстрація функціональних можливостей

Ризики пов’язані з вибором моделі:

- Від розробника вимагається хороше володіння CASE-засобами та інструментами
- Програма не має бути критичною
- Вимагається наявність потужних CASE-засобів

Коли краще застосовувати модель

- Користувачі не можуть чітко сформулювати вимоги
- Вимагається рання демонстрація можливостей

Вибір моделі

Вибір моделі суттєво залежить від двох факторів:

- А) чи можна спочатку визначити практично повний набір функцій, які необхідно реалізувати в програмному продукті
- Б) чи мають усі жадані функції поставлятись замовнику одночасно
- Якщо А і Б, то вибираємо каскадні моделі
- А і не Б – вибирається ітераційна модель з прирощуваннями
- Не А і Б, а також бажана розробка прототипів для моделювання вимог – спіральна модель
- Не А і не Б – модель швидкої розробки програм, при умові, що строки розробки не будуть чітко встановлені.