Спіральна модель життєвого циклу програмного продукту

Спіральна модель була вперше сформульована Барі Босмом в 1988 році. Відмінною

особливістю цієї моделі є підвищена увага до ризиків, що впливають на організацію

"життєвого циклу. Найбільш поширені ризики при розробці програмного продукту

1. Дефіцит спеціалістів

2. Нереалістичні терміни та бюджети

3. Реалізація невідповідної функціональності

4. Розробка неправильного інтерфейсу користувача

5. Непотрібна оптимізація

6. Безперервний потік змін

7. Нестача інформації про зовнішні компоненти, що визначають оточення, втягнене в

8. Недоліки в роботі, що виконується зовнішніми (по відношенню до проекту)

9, Недоліки в продуктивності отриманої системи

10. "Розрив" в кваліфікації спеціалістів різних областей знань

Спіральна модель включає в себе переваги каскадної моделі. Крім того в неї включені

аналіз ризиків, управління ризиками, а також процеси підтримки та менеджменту. Тут також

передбачена розробка програмного продукту при використанні методу прототипування чи

швидкої розробки програм.

Модель відображає базову концепцію, яка полягає в тому. що кожний цикл представля:

собою набір операцій, якому відповідає така ж кількість стадій як її в моделі каскалного

процесу. Причому приймається до уваги кожна складова частина проекту, її кожний рівень

складності, починаючи з загального формулювання вимог і закінчуючи кодуванням кожної

окремої програми.

Розглянемо квадранти спіральної моделі:

«визначення цілей, альтернативних варіантів та обмежень. Виконується

визначення цілей, таких як робоча характеристика, виконувані функції, можливість внесення

змін, найважливіших факторів досягнення успіху та апаратного/програмного інтерфейсу

Визначаються альтернативні способи реалізації цієї частини продукту (конструювання,

повторне використання, покупка и т.д.), визначаються обмеження, що накладаються на

використання альтернативних варіантів (затрати, графік виконання, інтерфейс, обмеження

середовища тощо). Створюється документація, що підтверджує ризики, пов'язані з нестачею

досвіду в даній сфері, використанням нової технології, жорстким графіком, погано

організованими процесами тощо.

Оцінка альтернативних варіантів, ідентифікація та вирішення ризиків.

Виконується оцінка альтеративних варіантів відносно цілей та обмежень. виконується

визначення та вирішення ризиків (менеджмент ризиків, методика економічно вигідного

вибору шляхів вирішення, оцінка інших пов'язаних з ризиком ситуацій, коли можуть бути

фінансові втрати через продовження розробки системи (рішення про припинення/

продовження робіт над проектом тощо)

Розробка продукту наступного рівня. Типові дії, що виконуються в цьому квадранті

можуть включати в себе створення проекту, розробку коду, перевірку коду. тестування та

компонування продукту. Перша створена версія продукту базується на тому. що потрапляє в

поле зору замовника. Потім починається фаза планування: програма повертається в

початковий стан з ціллю врахування реакції клієнту, Кожна наступна версія більш точно

відображає вимоги замовника. Кількість внесених змін від однієї версії програми до іншої

зменьшується, що в кінці призводить до отримання функціональної системи.

Планування наступної фази. Тнпові дії в цьому квадранті можуть включати в себе

розробку плану проекту, розробку плану менеджменту конфігурацій, розробку плану

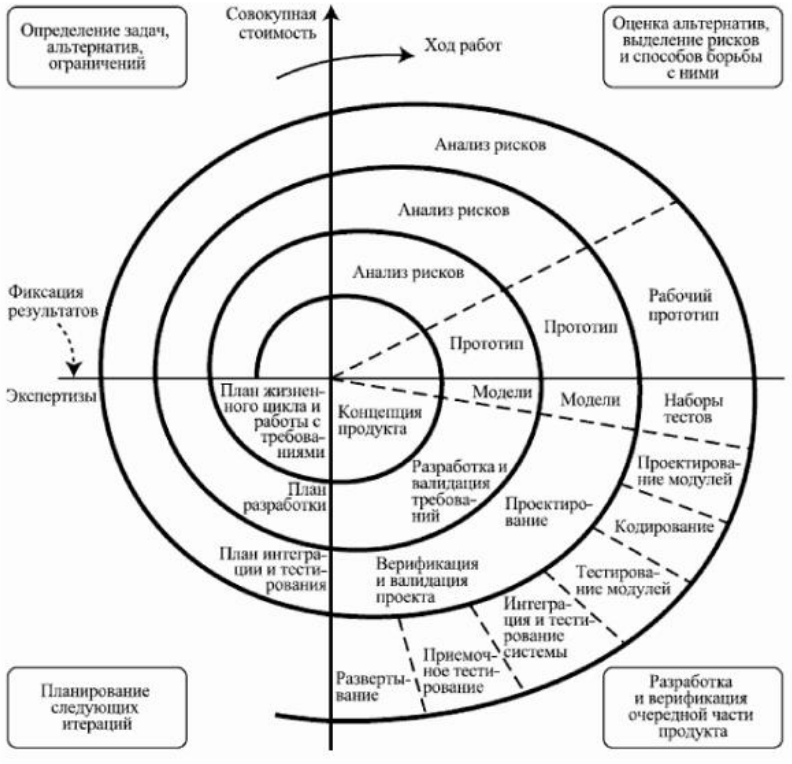
тестування та розробку плану установки програмного продукту.

Для кожної ітерації потрібно виділити цілі, альтернативні варіанти та обмеження,

встановити та вирішити ризики; дати оцінку альтернативним варіантам; розробити дані для

даної ітерації та підтвердити їх правильність; спланувати наступну ітерацію. Потім

необхідно вибрати метод здійснення наступної ітерації у випадку її необхідності.



В квадрантах відсутня задана кількість циклів. Їх кількість треба вибирати по

необхідності, а самі ітерації можна адаптувати під конкретний проект.

Також необхідно відмітити, що кодування відбувається значно пізніше, ніж в інших

моделях. Сенс полягає в тому, щоб мінімізувати ризик за допомогою послідовних

уточнювань вимог користувачем.

в квадрантах відсутня задана кількість циклів. Їх кількість треба вибирати по

необхідності, а самі ітерації можна адаптувати під конкретний проект. : .

Також необхідно відмітити, що кодування відбувається значно пізніше, ніж в інших

моделях. Сенс полягає в тому, щоб мінімізувати ризик за допомогою послідовних

уточнювань вимог користувачем.

1.4.1 Область застосування спіральної моделі

Розглянемо причини за яких ефективне використання спіральної моделі

1) в проекті будуть застосовуватися нові технології и необхідно протестувати базові

концепції

2) вимоги до проекту складні і незрозумілі на момент його початку

3) замовники не впевнені у своїх вимогах до проекту

4) організація не має можливості виділити на початку всі необхідні для виконання

проекту фінансові кошти, и коли в процесі розробки відсутня фінансова підтримка

5) для виконання затягнутих проектів

6) якщо створення прототипу є підходящий тип розробки продукту

7) коли необхідно повідомити, яким чином буде проходити збільшення затрат, і

підрахувати затрати, пов'язані з виконанням дій із квадратна ризику.

8) якщо організація має навики необхідні для адаптації моделі

9) якщо виконання проекту пов'язане з середнім та високим ступенем ризику

10) якщо немає сенсу не братися за довготерміновий проект через потенційні зміни. що

відбутися в економічних пріоритетах, і коли така невизначеність може викликати

обмеження у часі

11) Якщо очікуються значні зміни, наприклад, при вивченні або дослідницькій роботі

12) коли важливо сконцентрувати увагу на незмінних або відомих частинах, при чому

збір інформації про змінні частини ще не закінчений

15) якщо системи що розробляються вимагають великого об'єму обчислень,

наприклад системи, що забезпечують прийняття рішень

16) для виконання бізнес-проектів, а також проектів з області асрокосмічної

омисло оборони та інжинірінга, де використання спіральної моделі вже отримало

популярність.

1.4.2 Аналіз переваг та недоліків спіральної моделі життєвого циклу

Спіральна модель життєвого циклу була розроблена Барі Босмом як альтернатива

каскадній моделі, яка не могла адекватно вирішувати проблеми внесення змін на протягом

розробки програмного продукту. Дана модель має всі переваги каскадної моделі, але в неї

крім того включені аналіз ризиків та управління ними, процеси підтримки та управління.

Розглянемо основні переваги використання даної моделі:

дана модель забезпечує розбиття всього функціоналу системи на невеликі частини, в

спочатку реалізуються найважливіші функції з високим степенем ризику. У випадку

і і стає можливим припинити розрозробку системи і зменшити витрати на проект;

в спіральну модель включені всі переваги каскадної моделі, крім того існує

можливість ітерацій по всім фазам цієї моделі;

в спіральній моделі деякі переваги інкре !

можливість скорочення графіку шляхом переривання інкременттв;

в даній моделі особлива увага приділена ризикам. Забезпечується можливість

визначення найважливіших ризиків без особливих затрат:

активна участь користувачів у житті проекту. Дана модель дозволяє користувачам

приймати участь при плануванні, аналізів ризиків, розробці; все це сприяє кращій адаптації

програмного продукту під вимоги користувача,

в спіральній моделі на більш високому рівні реалізований контроль над процесами

розробки, тестування, дотримання графіку. Все це досягається за допомогою огляду в кінці

кожної і ітерації;

Недоліки спіральної моделі полягають у наступному:

модель має досить складну структуру, можуть виникнути проблеми з її правильним

використанням;

дана модель може виявитися надлишковою для певного виду проектів. Так, якщо

проект має низькі ризики, невеликі розміри, то оцінка ризиків, планування, прототипування

для кожної ітерації може виявитися надто затратним.

Отже, спіральна модель включає в себе всі переваги каскадної та інкрементної моделі,

а також можливість аналізу ризиків, прототипування, тісної взаємодії з замовником. Лле

через її складність, використання спіральної моделі може призвести до надлишку непотрібних витрат.