Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій

Кафедра технологій управління

**«ВСТУП ДО СПЕЦІАЛЬНОСТІ»**

Самостійна робота

Реферат на тему:

**«Методології управління проектами»**

Роботу виконав

студент групи УПЗ-11

Яковенко Антон

Перевірив

професор Морозов В.В.

Київ 2019

Зміст

[Вступ 3](#_Toc25437591)

[Основні сфери діяльності PMI 7](#_Toc25437592)

[Стандарти, які розроблені PMI 7](#_Toc25437593)

[Зміст ANCI PMI PMBOK Guide 4th Edition, 2008 8](#_Toc25437594)

[Передумови для впровадження управління проектами в ІТ 9](#_Toc25437595)

[Процес розробки ПЗ. Основні складові. Класифікація моделей процесу розробки ПЗ 11](#_Toc25437596)

[Види методологій 11](#_Toc25437597)

[Що задають 11](#_Toc25437598)

[Базові процеси 11](#_Toc25437599)

[Існуючі моделі 11](#_Toc25437600)

[Коротка характеристика моделей розробки ПЗ 13](#_Toc25437601)

[Каскадна модель розробки ПЗ 13](#_Toc25437602)

[Інкрементна модель розробки ПЗ 14](#_Toc25437603)

[Зміни в ПЗ та їх врахування в процесах розробки 15](#_Toc25437604)

[Каскадні моделі 15](#_Toc25437605)

[Гнучкі моделі 16](#_Toc25437606)

[Гнучка методологія розробки та її порівняння з традиційними підходами 18](#_Toc25437607)

[Основні ідеї 19](#_Toc25437608)

[Приклади 19](#_Toc25437609)

[Extreme programming (XP), Scrum 19](#_Toc25437610)

[Інструментарій методології розробки ПЗ «екстремальне програмування» 19](#_Toc25437611)

[Особливості кодування та тестування в методології XP 21](#_Toc25437612)

[Написання коду 22](#_Toc25437613)

[Тестування 22](#_Toc25437614)

[Управління програмними проектами, що розробляються за гнучкою методологією 24](#_Toc25437615)

[Принципи 24](#_Toc25437616)

[Agile маніфест 25](#_Toc25437617)

[Масштабування гнучкої методології розробки 25](#_Toc25437618)

[Висновок 27](#_Toc25437619)

[Джерела 28](#_Toc25437620)

Вступ

Управління компанією - це, перш за все, ефективне управління проектами. Бізнес - це величезна безліч проектів, які можуть йти один за іншим або одночасно. Тому проектний менеджмент заслуговує на особливу увагу, адже грамотна його реалізація - це головний внесок в успішне виконання проекту.

Інше ключове поняття в бізнесі - люди. Це клієнти, для яких виконуються проекти, і співробітники, які цим займаються. Зазвичай проектами відповідає система проектного менеджменту, а за управління клієнтами - CRM. CRM допомагає формалізувати і контролювати всі етапи роботи з клієнтами, а також містить базу даних про них. Знаючи про своїх клієнтів все, ви ефективно задовольняєте їх потреби.

Мета будь-якої компанії, що працює на ринку послуг полягає в тому, щоб клієнти залишалися задоволені і хотіли співпрацювати саме з цією компанією якомога довше.

Стандарти в управлінні проектами

З розвитком дисципліни управління проектами в світі з'являється все більше стандартів, автори яких намагаються систематизувати накопичені знання з управління проектами. Користь стандартів, перш за все, виражається в тому, що вони містять кращі практики в галузі управління проектами. Стандарти акумулюють досвід багатьох експертів, причому цей досвід постійно поповнюється, стандарти регулярно доповнюються і оновлюються. Стандарти управління проектами дозволяють різним компаніям розмовляти однією мовою при реалізації спільних проектів. Поширення стандартів сприяє кращому розумінню принципів управління організацій-партнерів. Наприклад, практика створювати статут проекту є загальноприйнятою в великих проектах. Стандарти є основою для сертифікації керівників проектів та інших фахівців. Міжнародний сертифікат приймається як своєрідна "гарантія якості" при наймі та співробітництво.

Основні стандарти управління проектами і країни розробки показані на

рис. 1

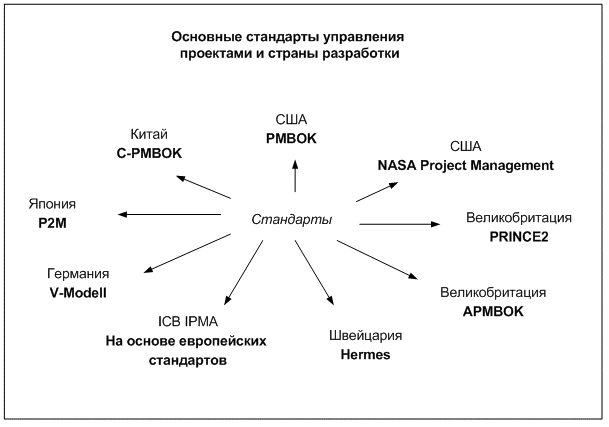


Рисунок 1. Основні стандарти управління проектами

PMBOK. Набір знань з управління проектами від PMI. Застосовується в більшості країн світу. Найбільшого поширення набув у США, Росії, ПАР, Фінляндії, Швеції, Данії, Норвегії, Литві. Стандарт PMBOK заснований на загальновизнаних практиках і знаннях, які можуть застосовуватися по відношенню до більшості проектів.

ICB (IPMA Competence Baseline). ICB поєднує в собі ряд європейських стандартів. Був розроблений на основі британського, швейцарського, німецького і французького стандартів компетенцій. Включає 28 основних галузей знань в управлінні проектами і 14 додаткових. Описує компетенції менеджерів проектів. Найбільшого поширення набув в Євросоюзі, Україні, Індії, Казахстані, Азербайджані.

PRINCE2. Спочатку був розроблений як стандарт для ведення державних ІТ-проектів Великобританії, але незабаром став використовуватися як універсальний метод управління проектами. PRINCE2 є процесно-орієнтований підхід до управління проектами. Виділяє 8 основних процесів управління проектами та 45 підпроцесів. Популярний в Бельгії, Хорватії, Польщі.

ARMBOK. Національний стандарт Великобританії, охоплює 52 області знань, необхідних для успішного ведення проекту. ARMBOK був одним з основних стандартів, які використовуються при розробці ICB.

C-PMBOK. Китайський стандарт розроблений на основі PMBOK. Існує також Cs-PMBOK - китайська версія ICB IPMA.

P2M. Японський P2M фокусується на управлінні програмами. Має на меті реалізацію складних інноваційних ідей та інтеграцію їх з областями діяльності підприємства. Також широко поширений в Південній Кореї.

V-Modell. VEE являє собою набір стандартів в області проектів, що стосуються розробки нових продуктів. Є основним в першу чергу для німецьких федеральних адміністративних і оборонних проектів. Багато в чому схожий з PRINCE2 і описує методи як для проектного управління, так і для системного розвитку. Сучасною версією V-Modell є V-Modell-XT, яка була затверджена в лютому 2005 р.

Hermes. Швейцарський Hermes в основному застосуємо для управління проектами в сфері інформаційних технологій. Використовується в Люксембурзі і федеральних органах влади Швейцарії. При його розробці багато було взято з V-Modell.

NASA Project Management. Стандарт розроблений для управління космічними проектами (для внутрішнього використання NASA).

Розглянемо докладніше найбільш поширений в Україні PMBOK. Даний стандарт буде використовуватися в курсі навчання для вивчення галузей знань і процесів управління проектами. PMBOK розробляє Інститут управління проектами (Project Management Institute).

PMI заснований в 1869 році і є найбільшою в світі некомерційною асоціацією професійних керівників проектів.

Основні сфери діяльності PMI

• розробка стандартів в управлінні проектами;

• проведення конференцій і семінарів;

• організація навчання;

• професійна сертифікація.

Основна професійна сертифікація PMI - ступінь PMP (Project Management Professional). Основні вимоги до кандидатів, це досвід управління проектами не менше 3 років і проходження 35 годин навчальних курсів з управління проектами.

Стандарти, які розроблені PMI

• ANCI PMI PMBOK Guide 4th Edition, 2008. Основною стандарт PMI, що описує всі процеси управління проектами;

• The Standard for Program Management. Стандарт управління програмами проектів;

• The Standard for Portfolio Management. Стандарт управління портфелями;

• Practice Standard for Work Breakdown Structure. Практичний стандарт з побудови ієрархічної структури робіт проекту;

• Project Management Competency Development Framework. Керівництво з оцінки та розвитку професійних навичок менеджерів проектів;

• Organization Project Management Maturity Model. Стандарт зрілості управління проектами;

• Practice Standard for Earned Value Management Project. Стандарт управління освоєним обсягом у проекті.

Зміст ANCI PMI PMBOK Guide 4th Edition, 2008

• предмет і основні поняття управління проектами (проект, життєвий цикл проекту, фази і стадії проекту, основні учасники проекту і т.д.);

• 5 груп процесів управління проектами;

• 9 областей знань управління проектами;

• 42 процесу управління проектами (Опис кожного процесу включає в себе вхідні дані, методи і інструменти, вихідні дані).

Стандарти відображають системну картину дисципліни управління проектами. Положення стандартів зазвичай носять рекомендаційний характер і відповідають на питання "Що робити, щоб ефективно управляти проектами", залишаючи компаніям свободу вибору в питанні "Як робити?". Детальний опис, як управляти проектами повинно міститися в корпоративному стандарті управління проектами кожної компанії (регламенти, положення, інструкції, накази, методологія, шаблони), обов'язковому для виконання в організації або проект і відповідному (що не суперечить) вибраному стандарту.

Передумови для впровадження управління проектами в ІТ

Історія розвитку проектного менеджменту як дисципліни відносно молода: її відносять до 30-х років XX століття і пов'язують з розробкою спеціальних методів координації інжинірингу великих проектів в США - авіаційних в «US Air Corporation» і нафтогазових в фірмі «Exon». В СРСР в цей же період почала розвиватися теорія і практика потокової організації робіт по реалізації великих будівельних проектів.

Передумови для впровадження принципів проект-менеджменту в процес розробки ПЗ зародилися в кінці 60х - початку 70-х років минулого століття, коли відбулася подія, яке увійшло в історію як перша криза програмування. Подія полягало в тому, що вартість ПЗ стала наближатися до вартості апаратури («заліза»), а динаміка зростання цих вартостей дозволяла прогнозувати, що до середини 90-х років все людство буде займатися розробкою ПЗ для комп'ютера.

Зараз вибір методології проектування як ніколи схильний до впливу маркетингу. Все більше з'являється консультантів по впровадженню agile, коучерів, які проводять нескінченні тренінги, семінари, вебінари, нескінченні зустрічі, конференції, круглі столи. Всі ці заходи спрямовані на продаж впровадження в ІТ-компаніях за великі гроші запрошеними фахівцями або підвищення рейтингу компаній, які вже впровадили гнучкі методології.

Гнучкі методології зараз - це більшою мірою звід знань по організації роботи людей з психологічної точки зору. Такі методології допомагають команді проявляти творчу складову, вміння працювати в команді, навички комунікації та інше. Технічна сторона організації робіт все більше відходить на другий план.

При неякісному впровадженні Agile ми отримуємо те, що зараз відбувається на ринку IT продуктів. Ринок перенасичений неякісними, нестабільними продуктами, що не відповідають вимогам не до функціоналу, ні до інтерфейсу. При цьому, що швидкість випуску таких продуктів, завдяки пропагованому Agile принципом безперервної інтеграції, стає дедалі більше.

Процес розробки ПЗ. Основні складові. Класифікація моделей процесу розробки ПЗ

Створення ПЗ - це сукупність процесів, що призводять до створення програмного продукту. Ці процеси грунтуються головним чином на технологіях інженерії програмного забезпечення.

Модель виробництва являється деяким абстрактним представленням про процес виробництва програмного забезпечення. Характеризується цей процес з деякої позиції.

Види методологій

* Керовані планом
* Гнучкі (agile)

Що задають

* Етапи та їх порядок
* Продукти (результати)
* Ролі розробників
* Перед- та постумови

Базові процеси

* Специфікація
* Проектування
* Атестація
* Розвиток

Існуючі моделі

* Каскадна (waterfall). З чітким розподіленням фаз розробки
* Інкрементна (покрокова) розробка. Одночасне виконання етапів
* На основі повторного використання компонентів
* Підлаштування загальної моделі під деякий підхід. Наприклад, розробка на основі формальних методів

Також є змішаний підхід

Коротка характеристика моделей розробки ПЗ

Каскадна модель розробки ПЗ

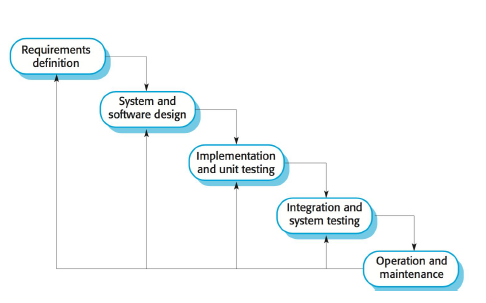


Рисунок 2. Каскадна модель розробки ПЗ

* Найбільш наближена до інженерних процесів у інших сферах
* Основний недолік – «важкість». Внесення змін – проблематичне
* Прив’язана до добре розроблених вимог
* Переважно використовується в розробці великих проектів

Інкрементна модель розробки ПЗ

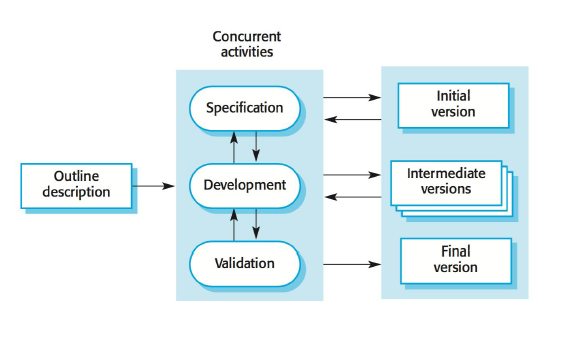


Рисунок 3. Інкрементна модель розробки ПЗ

Паралелльне виконання Специфікацій, Розробки, Атестації.

Переваги

* Зменшення ціни модифікації для потреб змінюваних вимог замовника
* Спрощується взаємодія із замовником. Рання реакція на продукт
* Пришвидшений випуск програмного продукту

Недоліки:

* Відносна інкапсуляція процесу розробки
* Розрихлення програмного продукту по мірі додавання нових інкрементів

Зміни в ПЗ та їх врахування в процесах розробки

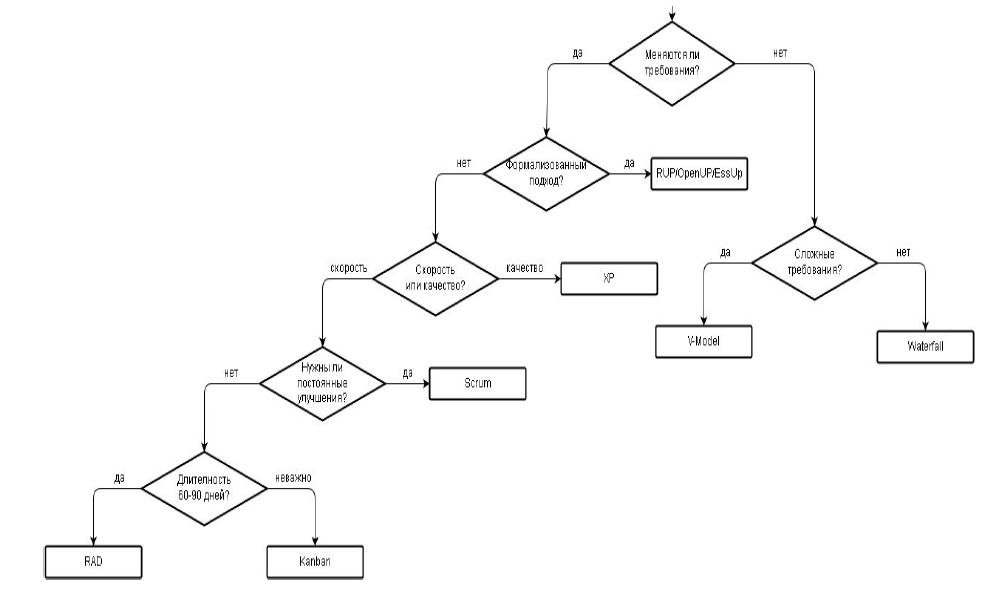


Рисунок 4. Методології розробки ПЗ

Є 2 варіанти моделей розробки ПЗ - гнучка і каскадна, каскадна не стійка до змін ПЗ, гнучка стійка. Розглянемо більш детально на прикладах моделі розробки ПЗ:

Каскадні моделі

«Waterfall Model» (каскадна модель або «водоспад»). Характеризується тим, що етапи розробки, такі як: аналіз, проектування, реалізація, тестування, йдуть один за одним. Дозволяє швидко створювати систему, без додаткових накладних витрат на організацію процесу розробки. Однак вона працює тільки тоді, коли вимоги стабільні і не змінюються в ході розробки, тому що ми відразу описуємо всі вимоги, а потім відразу проектуємо всю систему цілком.

«Incremental Model» (інкрементна модель). У інкрементної моделі повні вимоги до системи діляться на різні збірки. Термінологія часто використовується для опису поетапної зборки ПЗ. Мають місце кілька циклів розробки, і разом вони складають життєвий цикл «мульти-водоспад». Цикл розділений на більш дрібні легко створювані модулі. Кожен модуль проходить через фази визначення вимог, проектування, кодування, впровадження та тестування. Процедура розробки по інкрементної моделі передбачає випуск на першому великому етапі продукту в базовій функціональності, а потім вже послідовне додавання нових функцій, так званих «інкрементів». Процес триває до тих пір, поки не буде створена повна система.

Гнучкі моделі

«Agile Model» (гнучка методологія розробки). У «гнучкій» методології розробки після кожної ітерації замовник може спостерігати результат і розуміти, задовольняє він його чи ні. Це одна з переваг гнучкої моделі. До її недоліків відносять те, що через відсутність конкретних формулювань результатів складно оцінити трудовитрати і вартість, необхідні на розробку. Екстремальне програмування (XP) є одним з найбільш відомих застосувань гнучкої моделі на практиці.

Спіральна модель (Spiral). Орієнтована на проекти, в яких є серйозні ризики. Розробка представляється у вигляді спіралі. Кожен виток спіралі - ітерація. Виток спіралі складається з чотирьох етапів: планування, аналіз ризиків, розробка, оцінювання замовником. В кінці кожної ітерації вирішується, чи варто продовжувати проект. Характерною рисою є те, що на етапі аналізу ризиків створюються прототипи, концепти, моделі які покликані дозволити ризик на ранній стадії. Чим далі рух по спіралі, тим більше розробки продукту і менше прототипів і концептів. Типове застосування такої моделі - дослідні проекти. Є дуже дорогою моделлю, і не виправдана в системах, де ризики незначні.

Ітеративна модель. Орієнтована на проекти, де вимоги можуть змінюватися по ходу розробки. Проект складається з ітерацій (від 1-2 до 6 тижнів). Кожна ітерація може включати в себе етап аналізу, проектування, реалізації, тестування. Має великі накладні витрати на організацію процесу, ніж каскадна модель, проте вартість виправлення помилки в залежності від тривалості проекту не така висока. Наступні методології реалізують ітеративну модель: Scrum, XP, почасти Kanban.

Методологія RAD (Rapid Application Development). Орієнтована на швидку розробку програми, ітеративно, з максимально простою архітектурою, мінімальними витратами на процес, максимально використовуючи готові компоненти і потужні інструменти. Має обмеження на тривалість проекту - 60-90 днів.

Гнучка методологія розробки та її порівняння з традиційними підходами

Гнучка́ розро́бка програ́много забезпе́чення (англ. Agile software development, agile-методи) — клас методологій розробки програмного забезпечення, що базується на ітеративній розробці, в якій вимоги та розв'язки еволюціонують через співпрацю між самоорганізовуваними багатофункціональними командами. Гнучка розробка — найкращий засіб для підвищення продуктивності розробників програмного забезпечення.

Більшість гнучких методологій націлені на мінімізацію ризиків, шляхом зведення розробки до серії коротких циклів, що мають назву ітерацій, які зазвичай тривають один-два тижні. Кожна ітерація сама по собі виглядає як програмний проект в мініатюрі, і включає всі завдання, необхідні для видачі мінімального приросту за функціональністю: планування, аналіз вимог, проектування, кодування, тестування і документування. Хоча окрема ітерація, як правило, недостатня для випуску нової версії продукту, мається на увазі те, що гнучкий програмний проект готовий до випуску наприкінці кожної ітерації. Після закінчення кожної ітерації, команда виконує переоцінку пріоритетів розробки.

Agile акцентує увагу на безпосередньому спілкуванні «віч-на-віч». Більшість agile команд розташовані в одному офісі, його іноді називають bullpen. Як мінімум вона включає і «замовників» (замовники, які визначають продукт, також це можуть бути менеджери продукту, бізнес аналітики або клієнти). Офіс може також включати тестувальників, дизайнерів інтерфейсу, технічних авторів і менеджерів. Основною метрикою agile методів є робочий продукт. Віддаючи перевагу безпосередньому спілкуванню, agile-методи зменшують обсяг письмової документації в порівнянні з іншими методами. Це привело до критики цих методів як недисциплінованих.

Основні ідеї

* Особистості та їхні взаємодії важливіші, ніж процеси та інструменти;
* Робоче програмне забезпечення важливіше, ніж повна документація;
* Співпраця із замовником важливіша, ніж контрактні зобов'язання;
* Реакція на зміни важливіша, ніж дотримання плану.

Приклади

Extreme programming (XP), Scrum

В основі такого типу - нетривалі щоденні зустрічі - «Scrum» і регулярно повторювані зборів (раз в тиждень, раз на два тижні або раз на місяць), що називаються «Sprint». На щоденних нарадах учасники команди обговорюють:

* звіт про виконану роботу з моменту останнього Scrum'a;
* список завдань, які співробітник повинен виконати до наступних зборів;
* труднощі, що виникли в ході роботи.
* Коли використовувати Agile?
* Коли потреби користувачів постійно змінюються в динамічному бізнесі.
* Зміни на Agile реалізуються за меншу ціну через часті інкрементів.
* На відміну від моделі водоспаду, в гнучкою моделі для старту проекту достатньо лише невеликого планування.

Інструментарій методології розробки ПЗ «екстремальне програмування»

Дванадцять основних прийомів екстремального програмування можуть бути об'єднані в чотири групи:

1. Короткий цикл зворотного зв'язку (Fine scale feedback)

a. Розробка через тестування (Test driven development)

Автоматизована система модульного тестування використовуються для написання тестів для нової частини функціональності до її реалізації.

b. Інкрементальне планування (Incremental planning)

Вимоги записуються на картах і ті картки, які будуть включені в прес-релізі, визначаються часом їхнього виконання та їхнім відносним пріоритетом. З цих карток складається перелік завдань для розробників.

c. Замовник завжди поруч (Onsite customer)

Представник кінцевого користувача системи (замовник) повинен бути доступний повний робочий день, щоб використовувати команду “екстремального програмування”. Під час екстремального процесу програмування, клієнт є членом команди розробників і відповідає за пояснення вимог системи команді для подальшої реалізації

d. Парне програмування

Розробники працюють в парах, перевіряючи роботу один одного і забезпечують підтримку для того, щоб завжди добре виконувати роботу.

2. Безперервний, а не пакетний процес

a. Безперервна інтеграція (Continuous Integration)

Як тільки робота над завданням буде завершена, завдання інтегрується в систему. Після будь-якої такої інтеграції, всі юніт тести в системі повинні завершуватися успішно.

b. Рефакторинг (Refactor)

Розробники повинні виконувати рефакторінг коду відразу як будуть знайдені можливості його поліпшення. Це забезпечує простоту коду та можливість його подальшої підтримки.

c. Часті невеликі релізи (Small Releases)

Мінімальний корисний набір функціональних можливостей, які мають певне “бізнес” значення (business value (хуй зна як його перекласти)), розробляються в першу чергу. Релізи системи є частими та інкрементально додають нової функціональності до першого релізу.

3. Розуміння, що поділяється всіма учасниками

а. Простота (Simple design)

Дизайн здійснюється для задоволення поточних вимог і не більше.

b. Метафора системи (System metaphor)

c. Колективне володіння кодом (Collective ownership)

Пари розробників працюють на всіх ділянках системи так, що не існує острівців в системі, що вимагають виключних знань для розробки, і всі розробники беруть на себе відповідальність за всю систему. Будь-хто може змінювати будь-що.

d. Стандарт кодування (Coding standard or Coding conventions)

4. Соціальна захищеність програміста (Programmer welfare):

a. 40-годинний робочий тиждень (Sustainable pace)

Велика кількість понаднормових вважається неприйнятною та призводить до погіршення продуктивності та зниженню загальної якості коду.

Особливості кодування та тестування в методології XP

Екстремальне програмування (XPвід англ. *Extreme Programming*) — методологія розробки програмного забезпечення, найпопулярніша серед так званих гнучких методологій. Має на меті поліпшення якості програмного забезпечення та чутливість до змін у вимогах замовників. Як вид гнучких методологій, XP радить часті "випуски" програми у коротких циклах розробки, що має на меті поліпшити продуктивність праці та покращити можливості виконання вимог замовника що змінюються. Авторами даної методології є Кент Бек, Ворд Каннінгем, Мартін Фаулер та інші.

Інші елементи екстремального програмування включають в собі: парне програмування, проведення обширної перевірки сирцевого коду, модульне тестування всього коду, уникання створення функціональності до того як вона дійсно необхідна, простота та ясність коду, очікування на зміну вимог замовників з плином часу та коли вимоги до продукту стають ясніші, досить часте спілкування із замовником та між самими програмістами. Назва методології походить від ідеї застосувати корисні методи і практики розробки програмного забезпечення, піднявши їх до "екстремальних" рівнів. Критики XP зауважують на потенційні недоліки цієї методології – нестабільні вимоги, незадокументовані компроміси конфліктів користувачів, відсутність загального документу дизайну програми.

Написання коду

Прихильники ХР заявляють що єдиним дійсно важливим результатом розробки ПЗ є код: без готового коду нема продукту.

Тестування

Методологія екстремального програмування заявляє, що якщо дрібне тестування може перевірити незначну частину функціональності, то багато дрібних тестів можуть перевірити набагато більше частинок і продукт в цілому.

1. Планування процесу.

2. Тісна взаємодія з замовником.

3. Загальносистемні правила іменування.

4. Проста архітектура.

5. Рефакторинг.

Це оптимізація існуючого коду з метою його спрощення, Така робота повинна вестися постійно.

6. Парне програмування.

Всі програмісти повинні працювати в парах: один пише код, інший дивиться. Таким чином, необхідно розміщувати групу програмістів в одному місці.

7. 40-годинний робочий тиждень.

Програміст не повинен працювати більше 8 годин на день. Необхідність понаднормової роботи – це чіткий індикатор проблеми на даному конкретному напрямку розробки

8. Колективне володіння кодом.

Кожен програміст в колективі повинен мати доступ до коду будь-якої частини системи і право вносити зміни в будь-код. Обов'язкове правило: якщо програміст вніс зміни і система після цього працює некоректно, то саме цей програміст повинен виправити помилки.

9. Єдині стандарти кодування.

10. Невеликі релізи.

11. Безперервна інтеграція.

Інтеграція нових частин системи повинна відбуватися якомога частіше, як мінімум раз на кілька годин. Основне правило інтеграції наступне: інтеграцію можна проводити, якщо всі тести проходять успішно.

12. Тестування.

На відміну від більшості інших методологій тестування в XP – одна з найважливіших складових. Екстремальний підхід передбачає, що тести пишуться до написання коду. Кожен модуль повинен мати unit test – тест даного модуля. Таким чином, в XP здійснюється регрессионное тестування, «непогіршення якості» при додаванні функціональності. Більшість помилок виправляються на стадії кодування. Тести пишуть самі програмісти, будь-який з них має право написати тест для будь-якого модуля. Ще один важливий принцип: тест визначає код, а не навпаки (test-driven development), тобто шматок коду кладеться в сховище тоді і тільки тоді, коли всі тести пройшли успішно, у противному випадку дана зміна коду відкидається.

Управління програмними проектами, що розробляються за гнучкою методологією

ГМ — клас методологій розробки програмного забезпечення, що базується на ітеративній розробці, в якій вимоги та розв'язки еволюціонують через співпрацю між самоорганізовуваними багатофункціональними командами.

Принципи

* задоволення клієнта за рахунок ранньої та безперебійної поставки
* коштовного програмного забезпечення;
* вітання змін вимог навіть наприкінці розробки (це може підвищити
* конкурентоспроможність отриманого продукту);
* часта поставка робочого програмного забезпечення (кожен місяць або тиждень або ще частіше);
* тісне, щоденне спілкування замовника з розробниками впродовж
* всього проекту;
* проектом займаються мотивовані особистості, які забезпечені
* потрібними умовами роботи, підтримкою і довірою;
* рекомендований метод передачі інформації — особиста розмова
* (віч-на-віч);
* робоче програмне забезпечення — найкращий вимірювач прогресу;
* спонсори, розробники та користувачі повинні мати можливість
* підтримувати постійний темп на невизначений термін;
* постійну увагу поліпшенню технічної майстерності та зручному
* дизайну;
* простота — мистецтво не робити зайвої роботи;
* найкращі технічні вимоги, дизайн та архітектура виходять у
* самоорганізованої команди;
* постійна адаптація до мінливих обставин.

Agile маніфест

* Особистості та їхні взаємодії важливіші, ніж процеси та інструменти;
* Робоче програмне забезпечення важливіше, ніж повна документація;
* Співпраця із замовником важливіша, ніж контрактні зобов'язання;
* Реакція на зміни важливіша, ніж дотримання плану.

Масштабування гнучкої методології розробки

Невеликі команди показують частіше хороші результати, ніж великі, тому необхідно по можливості вести розробку компактними командами. На жаль, часто буває так, що розмір проекту і терміни його реалізації просто не дозволяють вести розробку 5-9 людьми і доводитися задіяти кілька команд.

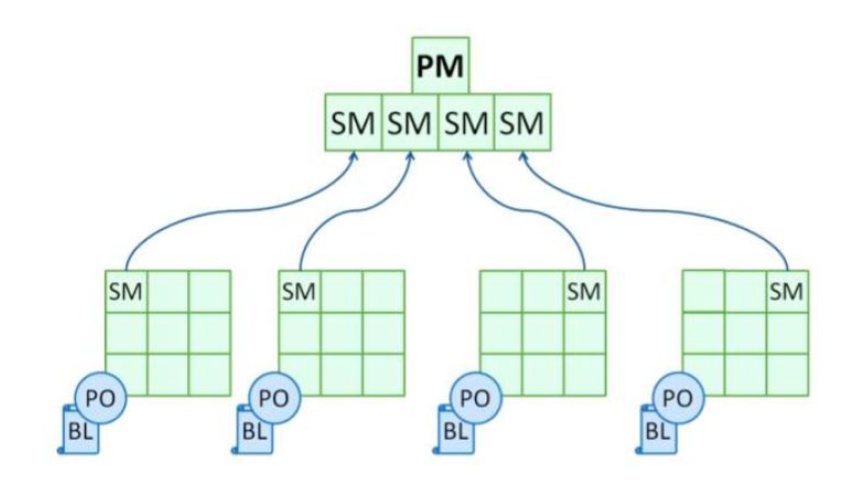


Рисунок 4. Масштабування гнучкої методології розробки

Для організації розробки великих проектів або портфеля проектів необхідно масштабувати Scrum на наступний рівень. З боку команд розробки – це виливається в проведенні Scrum of Scrum.

На цей мітинг збираються скрам-майстра (SM), в якості представників конкретних команд. Організує збори керівник програми (Program Manager - PM). У разі використання дивізійної організаційної структури на даному рівні, він також може бути керівником відповідного дивізіону (підрозділу).

Висновок

Під час написання реферату детально розглянуто тему «Методології управління проектами». У першій частині сформовано й описано основні стандарти, як вимоги до розглянутих далі методологій управління проектами. У другій частині перелічено самі методологій розробки ІТ-проектів, їх деталі, переваги недоліки та особливості впровадження. Увагу приділено гнучким методологіям, тому що саме такий їх різновид використовується у 80-90% управління проектами розробки ПЗ.

Джерела

1. https://habrahabr.ru/post/297612/
2. https://habrahabr.ru/company/edison/blog/269789/
3. <https://www.pmservices.ru/project-management-news/top-7-metodov-upravleniya-proektami-agile-scrum-kanban-prince2-i-drugie/>
4. <https://qaevolution.ru/metodologiya-menedzhment/waterfall/>
5. <https://ru.wikipedia.org/wiki/%D0%93%D0%B8%D0%B1%D0%BA%D0%B0%D1%8F_%D0%BC%D0%B5%D1%82%D0%BE%D0%B4%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D1%8F_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8>
6. <https://habr.com/ru/company/edison/blog/313410/>