ЗМІСТ

ПЕРЕЛІК ПОЗНАЧЕНЬ ТА СКОРОЧЕНЬ

БД – база даних;

ЖЦ – життєвий цикл;

ІС – інформаційна система;

МЖЦ – модель життєвого циклу;

ПЖЦ – процес життєвого циклу;

ПЗ – програмне забезпечення;

ПРПЗ – процес розробки програмного забезпечення;

ПС – програмна система;

CMM – Capability Maturity Model;

CMMI – Capability Maturity Model Integration;

UML – Unified Modeling Language.

ВСТУП

На сьогоднішній день автоматизація виробничих процесів є одним із найголовніших завдань сучасних підприємств. Найголовнішим інструментом для виконання цього завдання являється розробка програмного забезпечення.

Власне програмне забезпечення та його розробка вже давно не являються привілеями лише IT компаній та великих підприємств. Майже кожна компанія зараз потребує той чи інший програмний комплекс для забезпечення функціонування як і найдрібніших виробничих процесів, так і важливих сфер діяльності компанії. Розробка програмного забезпечення дозволяє оптимізувати документообіг, прискорити та поліпшити обслуговування клієнтів, підвищити якість комунікації співробітників, забезпечити контроль на кожному етапі діяльності компанії, тим самим забезпечуючи достатню рентабельність та підвищуючи конкурентну спроможність компанії. Саме тому до управління процесу розробки програмного забезпечення зараз приділяється стільки уваги майже у кожній компанії. Однак він має свої особливості, які необхідно враховувати при реалізації будь-яких проектів.

В даний момент існує декілька спроб оцінювання якості процесу розробки програмного забезпечення. Однією із таких спроб є технологія моделей зрілості CMMI. Вона надає приблизний опис того, як повинен виглядати ПРПЗ, але не надає чітких інструкцій як саме досягнути певного рівня якості. Ця область знань на сьогоднішній день не являється належно формалізованою. Основними проблемами, які необхідно вирішити на шляху покращення якості ПРПЗ в контексті CMMI являються проблеми синтезу моделі якості ПРПЗ, а також отримання оптимальної траєкторії покращення ПРПЗ. Синтезована модель якості ПРПЗ дозволить нам використати формальні методи оптимізації для знаходження оптимального стану ПРПЗ, який можна досягти маючу певні фінансові та часові ресурси. Тим самим ми отримаємо можливість планування розвитку ПРПЗ для окремо взятої організації.

Об’єктом роботи є процес розробки програмного забезпечення.

Предметом дослідження є моделі управління якістю ПРПЗ, алгоритми для вирішення задачі планування покращення якості ПРПЗ, а також інформаційні технології архітектури програмних систем.

Метою роботи є проектування програмної системи для автоматизації процесу отримання оптимальної траєкторії підвищення якості ПРПЗ, а також дослідження моделей та алгоритмів управління якістю ПРПЗ.

Для досягнення мети, необхідно дослідити модель та алгоритми, а також розглянути варіанти архітектури майбутньої програмної системи.

Оптимальна траєкторія покращення ПРПЗ дозволяє компанії розумно вкладати ресурси, як матеріальні так і часові, що є критичним в умовах сучасної конкуренції і обмеженості ресурсів. Саме автоматизація процесу отримання оптимальної траєкторії підвищення якості процесів ПРПЗ є актуальною на сьогоднішній день.

1 АНАЛІЗ ІСНУЮЧИХ ПІДХОДІВ ОЦІНКИ ЕФЕКТИВНОСТІ ПРОЦЕСУ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА МОДЕЛІ CAPABILITY MATURITY MANAGAMENT

1.1 Огляд предметної області процесу розробки програмного забезпечення

1.1.1 Визначення поняття процесу розробки програмного забезпечення

Основним предметом вивчення цієї роботи є процес розробки програмного забезпечення, тому спочатку необхідно описати його предметну область.

Перше визначення, яке ми розглянемо представлено в ISO12207 [1] , а саме було приведено наступне визначення: процес розробки програмного забезпечення – це сукупність дій, що відносяться до програмної інженерії, необхідна для того, щоб перетворити вимоги користувачів у програмне забезпечення, де під програмною інженерією розуміється організоване прикладення інженерних, природничо-наукових та математичних принципів та методів до економічно обґрунтованого виробництва якісного програмного забезпечення.

Наступне визначення, яке ми розглянемо представлено в документі SWEBOK [2] (що описує базовий набір знань з програмної інженерії; також відомий як стандарт ISO/IEC 19759). За цим документом під ПРПЗ розуміється сукупність діяльностей, методів, практичних процедур та перетворень, які використовуються людьми для розробки та підтримки програмного забезпечення (ПЗ) та зв’язаних з ним продуктів.

І, нарешті, найпростіше визначення має Кембриджський словник [3], який мовить, що ПРПЗ, це діяльність по створенню комп'ютерних програм.

Як можна побачити, дані визначення відрізняються незначно і є узгодженими у головному: у всіх визначеннях під ПРПЗ розуміється впорядкована сукупність дій, що ставлять собі за мету створення програмного продукту. Відмінності фактично пов’язані з інтерпретацією певного набору дій.

1.1.2 Моделі життєвого циклу процесу розробки програмного забезпечення

Життєвий цикл ПЗ - це безперервний процес, який починається з моменту прийняття рішення про необхідність створення ПЗ і закінчується в момент його повного вилучення з експлуатації.

ПРПЗ представляється у вигляді організованої сукупності процесів життєвого циклу ПРПЗ (ПЖЦ), кожний із яких відповідає за певну задачу, що виконується під час розробки. Існує велика кількість варіантів набору ПЖЦ, що використовуються в конкретних реалізаціях процесу розробки. Як правило, такі варіанти містять багато спільних елементів: в роботі [4] наведений спільний набір ПЖЦ, що являє собою покриття більшості існуючих варіантів:

1) системна ініціалізація (планування);

2) аналіз та визначення вимог;

3) функціональна специфікація та завдання прототипів системи;

4) декомпозиція ПЗ на компоненти та вибір стратегії їх реалізації;

5) архітектурне та детальне проектування;

6) реалізація та налагодження компонент;

7) інтеграція та тестування ПЗ;

8) створення документації та поставка ПЗ замовнику;

9) розміщення ПЗ на боці замовника;

10) навчання персоналу роботі за ПЗ;

11) підтримка розгорнутого ПЗ.

Фактично, такий набір будується згідно з прямою послідовністю етапів ЖЦ ПЗ, хоча, як це може стати зрозумілим в подальшому, сучасні МЖЦ дозволяють задавати значно складніші переходи від етапу до етапу.

До найбільш розповсюджених МЖЦ відносяться: модель водоспаду, спіральна модель, інкрементно – ітеративна модель. Розглянемо дані моделі.

Модель водоспаду – традиційна модель водопаду була запропонована в 70 – х роках XX ст. . Вона описує процес, в якому окремі етапи розміщені в суровій послідовності: аналіз вимог, проектування, кодування, тестування та інтеграція.

Кожний етап повинен згідно з повинен завершуватися верифікацією, валідацією та тестуванням, але всі етапи є обов’язковими і неможливо почати наступний етап без завершення попереднього. До недоліків даного підходу відносяться мала гнучкість (неможливо адекватно враховувати зміни до вимог), погана пристосованість до повторного використання рішень, складність в супроводі. Нині модель вважається застарілою.

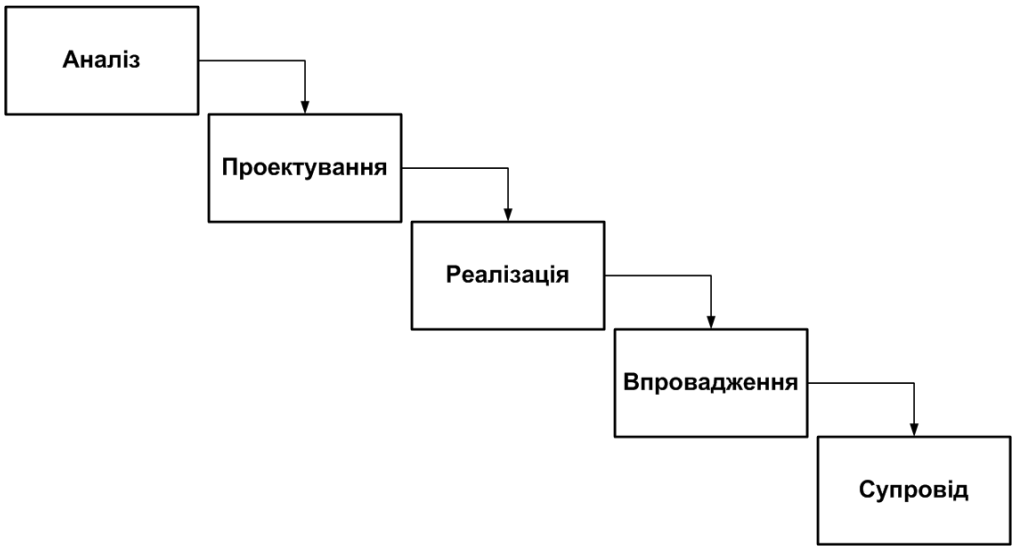

Рисунок 1.1 – Каскадна модель ПРПЗ

Спіральна модель – була розроблена в середині 80 – х років XX ст. Барі Боемом [5]. Вона представляє собою цикл (що постійно повторюється) робіт над всім програмним продуктом. Де кожний крок циклу відповідає повному набору ПЖЦ, визначених так як і для моделі водоспаду. При цьому прийняття рішень про тривалість окремих ітерацій циклу і вибір ПЖЦ на кожній ітерації керується за виявленими ризик – факторами, для чого ітерації циклу включають такі дії як визначення цілей, оцінювання альтернатив, розробка прототипів та фінального коду ПЗ, планування наступного циклу. Традиційна спіральна модель відрізняється від сучасних інкрементно – ітеративних підходів (описаних нижче) тим, що окрема ітерація для цієї моделі відноситься до роботи над всім продуктом в цілому, а не до окремої підмножини його функціональності.

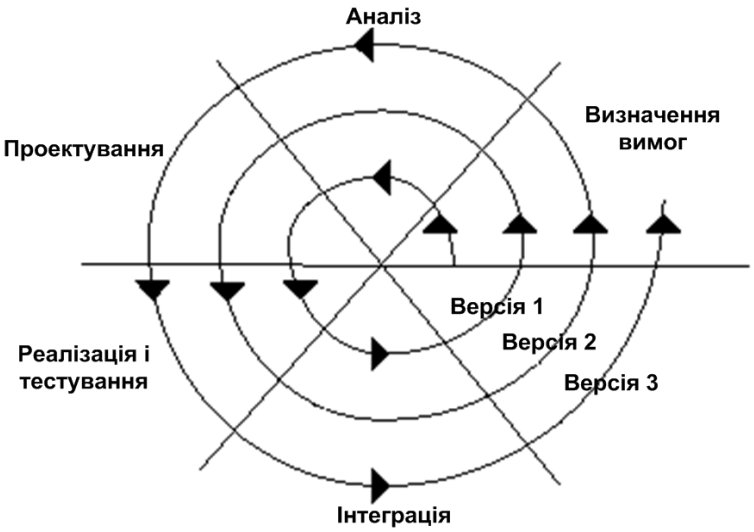

Рисунок 1.2 – Спіральна модель ПРПЗ

Інкрементно – ітеративна модель. Коли говорять про ітеративну та інкрементну розробку [6], мають на увазі два різних атрибути ПРПО: організацію послідовності його кроків і програмний продукт, який необхідно отримати в результаті його застосування.

Ітеративна розробка визначає особливу структуру послідовності кроків процесу розробки. При цьому процес розробки являє собою набір коротких ітерацій, кожна з яких містить набір всіх ПЖЦ моделі водоспаду, що включає, зокрема, етапи аналізу вимог, архітектурного проектування й кодування, при цьому обсяг робіт, що відносяться до різних етапів, відрізняється для різних ітерацій. Перша ітерація зазвичай проводить аналіз вимог, реалізує архітектуру системи і її базову функціональність. По мірі просування по итераціям обсяг робіт, пов'язаний з аналізом вимог і проектуванням архітектури скорочується, тоді як обсяг робіт, пов'язаний з розробкою, збільшується.

Інкрементна розробка визначає особливий підхід до випуску програмного продукту: замість того, щоб випускати остаточну версію за результатами всього ПРПЗ, ПЗ розробляється окремими відносно невеликими наборами функціональності (інкрементами), кожен з яких є проміжним результатом ПРПЗ.

Така розробка передбачає постійне розширення функціональності системи в ході її розробки.

Інкрементна та ітеративна розробка часто використовуються разом, такий підхід називається інкрементно-ітеративною МЖЦ. При цьому результатом ітерації може бути розробка деякого інкремента результуючого програмного продукту.

До переваг інкрементно-ітеративного підходу відносять:

– отримання користувачем деякої версії програмного продукту на більш ранньому етапі проекту;

– можливість надати користувачу більш важливі функції раніше, ніж менш важливі;

– психологічна зручність для розробників як наслідок того, що конкретні результати роботи у вигляді функціонуючих проміжних версій ПЗ з'являються на більш ранніх етапах проекту.

1.2 Основні проблеми процесу розробки програмного забезпечення

Насправді ПРПЗ є доволі складним процесом, який також має свої проблеми. Найбільш поширеними проблемами, що виникають в процесі розробки ПЗ, вважають проблеми прозорості, контролю, трасування, моніторингу та надійності [7].

Недолік прозорості полягає у тому, що у будь-який момент часу складно сказати, в якому стані знаходиться проект і який відсоток його завершення. Дана проблема виникає при недостатньому плануванні структури (чи архітектури) майбутнього програмного продукту, що найчастіше є наслідком відсутності достатнього фінансування проекту: програма потрібна, скільки часу займе розробка, якими є етапи, чи можна якісь етапи виключити або заощадити — наслідком цього процесу є те, що етап проектування скорочується.

Недолік контролю. Без точної оцінки процесу розробки зриваються графіки виконання робіт і перевищуються встановлені бюджети. Складно оцінити обсяг виконаної і залишилася роботи.

Наступні проблеми виникають на етапі, коли проект, завершений більш ніж наполовину, продовжує розроблятися після додаткового фінансування без оцінки ступеня завершеності проекту.

Недолік трасування та моніторингу виникає, коли неможливість спостерігати хід розвитку проекту не дозволяє контролювати хід розробки в реальному часі. За допомогою інструментальних засобів менеджери проектів приймають рішення на основі даних, що надходять в реальному часі.

Наступні проблеми виникають в умовах, коли вартість навчання менеджменту володінню інструментальними засобами порівнянна з вартістю розробки самої програми, такі як неконтрольовані зміни. У споживачів постійно виникають нові ідеї щодо розроблюваного програмного забезпечення. Вплив змін може бути суттєвим для успіху проекту, тому важливо оцінювати пропоновані зміни та реалізовувати тільки схвалені, контролюючи цей процес за допомогою програмних засобів. Внаслідок небажання кінцевого споживача використовувати ті чи інші програмні середовища виникають далі описані проблеми. Наприклад, коли при створенні клієнт-серверної системи споживач висуває вимоги не тільки до операційної системи на комп'ютерах-клієнтах, а й на комп'ютері-сервері. Тобто мова йде про недостатню надійність. Найскладніший процес — пошук і виправлення помилок у програмах на ЕОМ. Оскільки число помилок у програмах заздалегідь невідомо, то заздалегідь невідома і тривалість налагодження програм і відсутність гарантій відсутності помилок в програмах. Слід зазначити, що залучення доказового підходу до проектування ПЗ дозволяє виявити помилки в програмі до її виконання. У цьому напрямку багато працювали Кнут, Дейкстра і Вірт. Професор Вірт при розробці Паскаля і Оберона за рахунок строгості їх синтаксису домігся математичної доказовості виконання і правильності програм, написаної на цих мовах.

При неправильному виборі засобів розробки виникає ще одна проблема. Наприклад, при спробі створити програму, що вимагає коштів високого рівня, за допомогою засобів низького рівня. Наприклад, при спробі створити засоби

автоматизації з СУБД на асемблері. У результаті вихідний код програми виходить занадто складним і погано піддається структуруванню.

Ми говоримо про неправильний вибір методології розробки програмного забезпечення. Процес вибору необхідної методології може проблемно відбитися на всіх показниках програмного забезпечення — це його гнучкість, вартість і функціональність. Так звані гнучкі методології розробки допомагають вирішити основні проблеми, однак, варто відзначити, що і каскадна модель так само має свої переваги. У деяких випадках найбільш доцільним буде застосування гібридних методологій МЖЦ.

1.3 Аналіз сучасних підходів оцінки та покращення якості процесу розробки програмного забезпечення

В даний час існує безліч підходів до оцінки якості ПРПЗ. Найбільш успішними у практичному використанні є:

1) стандарт ISO 9001 (Quality management systems requirements);

2) модель CMMI - Capability Maturity Model Integration (належить університету Карнегі-Меллона, США).

Стандарт ISO 9001 [8]. Цей стандарт встановлює вимоги до системи менеджменту якості і спрямований на застосування процесного підходу при розробці, впровадженні та поліпшенні результативності системи менеджменту якості в цілях підвищення задоволеності споживачів шляхом виконання їхніх вимог. SO 9000 - серія міжнародних стандартів, що описують вимоги до системи менеджменту якості організацій і підприємств.

Серія стандартів ISO 9000 розроблено Технічним комітетом 176 (ТК 176) Міжнародної організації зі стандартизації. В основі стандартів лежать ідеї і положення теорії загального менеджменту якості (TQM).

Прийнято вважати, що при розробці першої версії стандартів ISO 9000 ТК 176 керувався британським стандартом BS 5750, розробленим Британським інститутом стандартів (BSI). У свою чергу, вважається, що британський стандарт базувався на галузевих стандартах ВПК.

Стандарти серії ISO 9000, прийняті більш ніж 190 країнами світу в якості національних, застосовні до будь-яких підприємств, незалежно від їх розміру, форм власності та сфери діяльності.

Сертифікація проводиться за єдиним стандартом з цієї серії, який містить вимоги - ISO 9001. Організація ISO не проводить сертифікацію по ISO 9001. Діє дворівнева система підтвердження відповідності. Сертифікацією систем менеджменту якості окремих організацій займаються спеціально сформовані аудиторські організації (органи з сертифікації). Вони, в свою чергу, акредитуються національними акредитаційними товариствами. Також існують і незалежні системи акредитації.

З точки зору ISO 9001 - для успішного функціонування, організація повинна визначити і здійснювати менеджмент численних взаємопов'язаних видів діяльності. Діяльність, що використовує ресурси і керована в цілях перетворення входів на виходи, може розглядатися як процес. Часто вихід одного процесу утворює безпосередньо вхід наступного.

Застосування в організації системи процесів, поряд з їх ідентифікацією і взаємодією, а також менеджмент процесів, спрямований на отримання бажаного результату, в ISO 9001 визначені як "процесний підхід".

Переваги процесного підходу полягає в безперервності управління, яке він забезпечує на стику окремих процесів у рамках їх системи, а також при їх комбінації і взаємодії.

При застосуванні в системі менеджменту якості такий підхід підкреслює важливість:

– розуміння і виконання вимог;

– необхідності розгляду процесів з точки зору цінності, що ними додається;

– досягнення запланованих результатів виконання процесів і забезпечення їх результативності; постійного поліпшення процесів, заснованого на об'єктивному вимірі.

На рисунку 1.3 наведена модель системи менеджменту якості ISO 9001, заснована на процесного підходу.

Рисунок 1.3 – Модель менеджменту якості ISO 9001

Ця модель показує, що споживачі відіграють істотну роль у встановленні вимог, які розглядаються як входи. Моніторинг задоволеності споживачів вимагає оцінки інформації про сприйняття споживачами виконання їх вимог. Наведена модель охоплює всі основні вимоги стандарту ISO 9001, але не показує процеси на детальному рівні.

Нижче наведено перелік областей, які повинні бути впроваджені організацією згідно стандарту ISO 9001:

3) система менеджменту якості;

4) відповідальність керівництва;

5) менеджмент ресурсів;

6) процеси життєвого циклу продукції;

7) вимірювання, аналізування та поліпшення.

У кожній з цих областей міститися вимоги, які повинні бути реалізовані організацією.

Стандарт ISO 9001 допускає адаптацію діючих систем менеджменту для створення системи менеджменту якості, що відповідає вимогам стандарту.

Вимоги стандарту є загальними і призначені для застосування всіма організаціями незалежно від їх виду, розміру і продукції, що поставляється.

В цілому ISO 9001 описує набір областей, вимог організації, які необхідно реалізувати для того, щоб підвищити якість своїх процесів і продуктів.

В ISO 9001 визначені тільки необхідні умови, що сприяють приведення процесів в організований стан.

Стандарт ISO 9001 вирішує наступні проблеми розробки ПЗ: брак прозорості та недолік спостереження (завдяки впровадженню області процесів життєвого циклу продукції можна прозоро відстежувати хід виконання проекту), недолік контролю (такі області як «система менеджменту якості», «менеджмент ресурсів», «процеси життєвого циклу продукції» дозволяють визначити процес адекватного визначення оцінки виконання проекту), недостатня надійність (тому що впроваджується система менеджменту якості). Але стандарт не вирішує такі проблеми як неконтрольовані зміни в вимогах. Стандарт ISO 9001 не вирішує проблеми формалізації в області покращення якості ПЗ.

Очевидним недоліком ISO 9001 є те, що по досягненню організацією вимог стандарту – не надається ніяких рекомендацій щодо подальшого поліпшення процесів.

Стандарт ISO 9001 не дає об'єктивної кількісної оцінки рівня якості ПРПО (процес чи відповідає стандарту (атестований) - чи ні (не атестований)).

Ще одним недоліком стандарту є свобода його інтерпретації, яка є ціною за ту високу абстракцію, яка була введена з метою охопити якомога ширше коло організацій.

Модель CMMI (Capability Maturity Model Integration). Модель CMMI розроблена Інститутом Програмної Інженерії (Software Engineering Institute, SEI), який, у свою чергу, є підрозділом Університету Карнегі-Меллона (Пітсбург, США).

CMMI спрямована в першу чергу на поліпшення процесів розробки програмного забезпечення всередині організації та отримання оцінки їх якості [9].

Ключовим у цій моделі є поняття зрілості організації і ПРПЗ.

Для початку визначимо різницю між зрілими і незрілими організаціями [9].

Характеристики незрілих організацій:

– процеси створюються під час виконання проекту;

– затверджені процеси ігноруються;

– процеси реактивні і не послідовні;

– нереалістичні бюджет і графік виконання;

– якістю нехтують, щоб укластися в розклад;

– відсутні об'єктивні показники якості.

Характеристики зрілих організацій:

– присутній комунікація і координація;

– роботи закінчуються у відповідності з планом;

– практики узгоджені з процесами;

– процеси оновлюються, коли це необхідно;

– ролі та обов’язки чітко визначені;

– управління відбувається офіційно.

Під процесом в CMMI (відповідно до [9]) розуміється послідовність кроків спрямована на досягнення поставленої мети.

Протягом того, як організації стає більш зрілою - ПРПЗ стає більш визначеним і послідовно реалізованим по всій організації .

Далі дамо визначення ключовій категорії CMMI.

Зрілість ПРПЗ - це ступінь, в якій процес повністю і явно визначено, управляємо, вимірюємо, контролюємо, ефективний і передбачуваний. Зрілість передбачає потенціал зростання в можливості і відображає повноту ПРПЗ і послідовність дій, необхідних для того, щоб застосовувати його в усіх проектах організації. ПРПЗ є "прозорим" по всій організації і доноситься до співробітників через спеціальні тренінги, лекції і через документацію. ПРПЗ постійно відстежується і поліпшується людьми, які його використовують.

Для поліпшення і оцінки якості ПРПЗ CMMI пропонує два подання моделі - поетапне (дискретне) і безперервне.

При поетапному (дискретному) поданні є можливість покращувати набір взаємопов'язаних процесів шляхом поетапного поліпшення набору фокусних областей, що послідовно розширюються

Поліпшення і оцінка якості ПРПЗ в CMMI на основі поетапного подання здійснюються відповідно до 5 рівнів зрілості (maturity levels).

Рівень зрілості - точно визначене еволюційне плато на шляху до досягнення повної зрілості бізнес-процесів організації.

Нижче наводиться опис рівнів зрілості.

Рівень 1 - Початковий (Initial) - ПРПЗ характеризується як хаотичний і невизначений. Кілька процесів визначено. Успіх проекту залежить від зусиль окремий особистостей.

Рівень 2 - Повторюваний (Repeatable) - присутні базові процеси по управлінню проектами для відстеження та контролю бюджетів, графіків виконання, і реалізованої функціональності. Введені процеси дозволяють повторити попередні успіхи на майбутніх проектах.

Рівень 3 - Певний (Defined) - ПРПЗ для менеджменту та інженерної діяльності документований, стандартизований і інтегрований у стандартний ПРПЗ організації.

Рівень 4 - Керований (Managed) - збираються детальні, кількісні (статистичні) показники ПРПЗ і якості продукту через які відбувається їх контроль.

Рівень 5 - Оптимізуючий (Optimizing) - засновано безперервне поліпшення ПРПЗ через кількісні показники та пілотні інноваційні ідеї та технології.

Кожен рівень зрілості складається з безлічі процесних областей.

Процесна область (Key Process Area, KPA) - визначає кластер дій, внаслідок яких досягається безліч цілей, важливих з погляду збільшення продуктивності ПРПЗ.

Для того, щоб організація відповідала вимогам процесної області, повинні бути досягнуті всі цілі цієї області.

Цілі (Goals) резюмують основні дії КРА і можуть використовуватися для визначення ефективності впровадження організацією або проектом цієї КРА. Цілі визначають обсяг, кордони і призначення кожної КРА. Цілі можуть бути загальними (з'являтися в декількох процесних областях) або приватними (належить до певної процесної області і відповідає унікальною характеристикою, яка описує, що повинно бути виконано для реалізації процесної області).

Кожна ціль складається з безлічі практик (Practice).

Практика описує інфраструктуру і дії, які в більшості випадків необхідні для успішної реалізації процесної області. Практики можуть бути загальними (Generic Practice, є важливою в досягненні асоційованої з нею спільної цілі) і приватними (Specific Practice, є важливою в досягненні асоційованої з нею частою цілі).

Процесні області і відповідні рівні зрілості можна побачити на рисунку 1.4.

Level	Focus	Process Areas	Result
5 Optimizing	*Continuous process improvement*	Organizational Innovation & Deployment Causal Analysis and Resolution	Productivity & Quality
4 Quantitatively Managed	*Quantitative management*	Organizational Process Performance Quantitative Project Management	
3 Defined	*Process standardization*	Requirements Development Technical Solution Product Integration Verification Validation Organizational Process Focus Organizational Process Definition Organizational Training Integrated Project Management Risk Management Decision Analysis and Resolution	
2 Managed	*Basic project management*	Requirements Management Project Planning Project Monitoring & Control Supplier Agreement Management Measurement and Analysis Process & Product Quality Assurance Configuration Management	
1 Initial	*Competent people and heroics*		

Рисунок 1.4 - Процесні області CMMI розподілені по рівням зрілості.

При безперервному поданні моделі зрілості є можливість поетапно покращувати процеси, які відносяться до окремо взятої фокусної області (або групі таких областей) а не до всього ПРПО. Для вимірювання якості ПРПО безперервне подання використовує таке поняття як рівні можливості (capability levels).

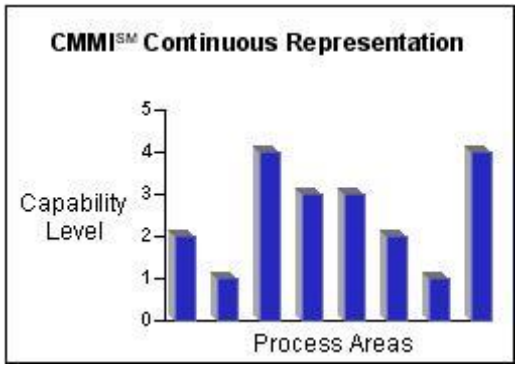

Рисунок 1.5 - Процесні області CMMI (безперервна модель)

Відзначимо, що для досягнення конкретного рівня, організація у будь-якому випадку зобов'язана задовольнити всіх цілям фокусної області або набору фокусних областей, які були намічені для поліпшення.

Основна відмінність між підходами (поданнями) полягає в тому, що поетапне подання використовує рівні зрілості для опису загального рівня процесів організації, взятої у цілому, у той час як безперервне подання використовує рівні можливостей для опису стану процесів організації щодо окремої фокусної області. Рівні можливостей застосовуються для вимірювання досягнень організації щодо поліпшення стану речей в окремих центральних областях (фактично вони оцінюють якість окремих процесів). Рівень зрілості застосовуються для вимірювання досягнень організації щодо поліпшення стану речей в цілому - через кордони фокусних областей. Фактично кожен такий рівень оцінює поліпшення роботи всієї організації в межах заданого для нього набору фокусних областей.

Виділяють 4 рівня можливостей (від 0 до 3) (перераховані нижче).

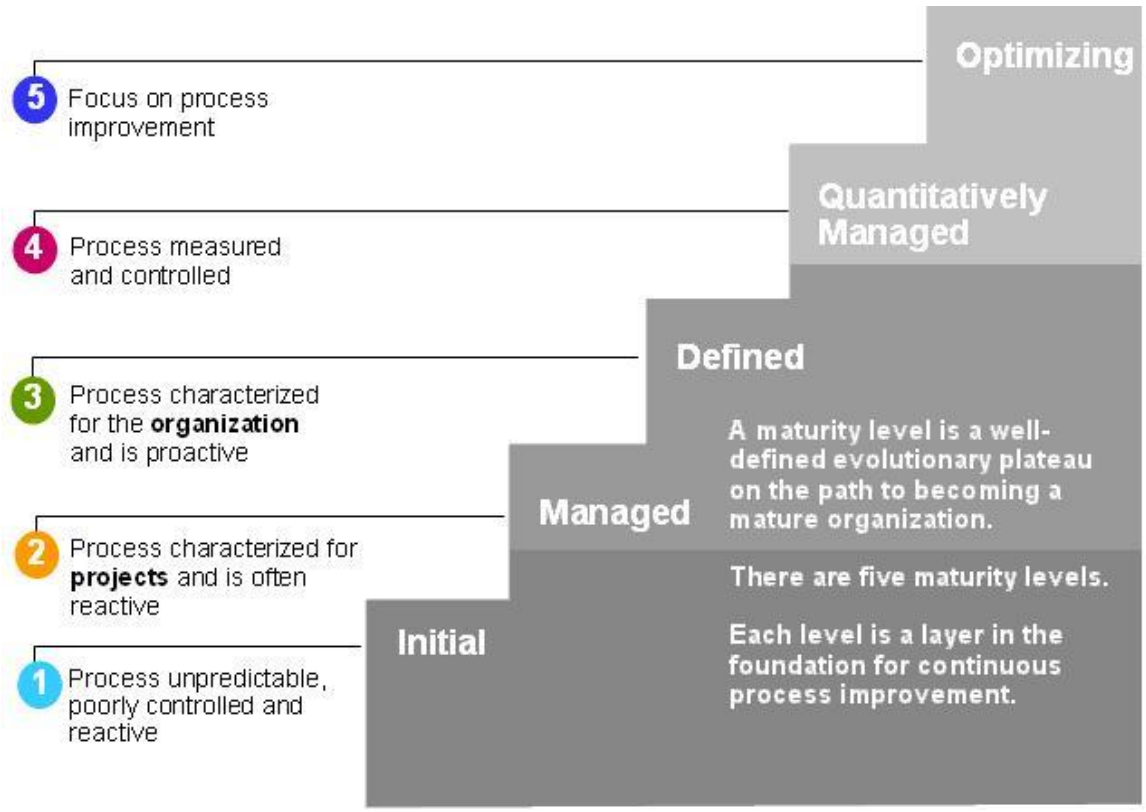

Рисунок 1.6 – Рівні можливостей CMMI

Рівень 0 – «Неповний (Incomplete)» - ПРПЗ характеризується як хаотичний і невизначений. Кілька процесів визначено. Успіх проекту залежить від зусиль окремий особистостей.

Рівень 1 – «Виконуваний (Performed)» - ПРПЗ характеризується як хаотичний і невизначений. Кілька процесів визначено. Успіх проекту залежить від зусиль окремий особистостей.

Рівень 2 – «Керований (Managed)» - ПРПЗ для менеджменту та інженерної діяльності документований, стандартизований і інтегрований у стандартний ПРПЗ організації. Всі проекти використовують схвалену, пристосовану версію стандарту ПРПЗ організації для розробки і підтримки.

Рівень 3 – «Певний (Defined)» - Збираються детальні, кількісні (статистичні) показники ПРПЗ і якості продукту через які відбувається їх контроль.

Якщо всі процеси для фокусної області відповідають певному рівню можливостей, то говорять про досягнення такого рівня для всієї фокусної області,

тобто фактично рівень можливостей фокусної області відповідає мінімальному рівню можливостей її практик.

Безперервне подання дозволяє організації вибрати напрямок зусиль з поліпшення процесу розробки шляхом вибору тих фокусних областей або наборів взаємопов'язаних областей, які вона вважатиме зручним. Після вибору областей потрібно вибрати рівень можливостей, якого ми плануємо досягти (він може бути різним для різних областей). Після досягнення певного рівня можна продовжити покращувати даний набір областей або розширити набір за рахунок нових областей.

Поетапне подання надає шлях до поліпшення від рівня зрілості 1 до рівня 5, який включає в себе досягнення цілей фокусних областей на кожному рівні. Модель зрілості задає рівні зрілості, до яких відноситься кожна фокусна область. Перехід до наступного рівня зрілості проводиться за досягнення всіх цілей всіх областей, що відносяться до поточного рівня.

Для оцінювання організації на основі безперервного подання використовується поняття еквівалентного завдання рівнів (equivalent staging). В даному випадку досягнуті рівні можливостей для набору фокусних областей перетворюються в оцінку рівня зрілості. Для цього використовуються наступні правила:

– для досягнення рівня зрілості 2, всі центральні області, що відносяться до рівня зрілості 2, повинні досягти рівня можливостей 2 або 3;

– для досягнення рівня зрілості 3, всі центральні області, що відносяться до рівнів зрілості 2 і 3, повинні досягти рівня можливостей 3;

– для досягнення рівня зрілості 4, всі центральні області, що відносяться до рівнів зрілості 2, 3 і 4, повинні досягти рівня можливостей 3;

– для досягнення рівня зрілості 5, всі центральні області повинні досягти рівня можливостей 3.

Перевагами моделі CMMI можна назвати: об’єднання системної і програмної інженерії в одну область – інженерія продукту, більш детальне (в порівнянні зі схожими моделями) покриття ЖЦ ПО, збільшення видимості

організаційної діяльності, задоволення вимог користувачів, фокусування на керуванні вимогами, зменшення вартості кінцевого продукту, користувач може вибрати подання моделі (дискретне чи безперервне) в залежності від бізнес – потреб.

Недоліком CMMI є те, що вона не підходить для невеликих організацій, тому що має жорсткі вимоги щодо ведення документації, підтримки досягнутих процесних областей – малим компаніям просто не вистачає ресурсів та знання, щоб забезпечити необхідну інфраструктуру.

Модель CMMI вирішує всі проблеми розробки ПЗ, та поряд з цим пропонує найбільш детальну та легку для розуміння модель розвитку організації, яка у деякому ступені є формалізованою (вербально).

1.4 Постановка задачі

На даний час проблема оцінки та покращення якості ПРПЗ є актуальною у зв'язку зі збільшенням складності програмних систем та процесів, що відбуваються під час їх розробки в IT компаніях. Як наслідок зростають витрати на такі системи і час їх розробки.

Але навіть з достатнім фінансуванням і маючи достатньо часу, не всі IT компанії гарантують успіх розробки ПЗ. Причиною цього слугує те, що в таких компаніях процеси з розробки ПЗ є хаотичними та не стандартизованими.

Одним з рішень є використання технології моделі зрілості CMMI – це модель оцінки якості ПРПЗІТ – компанії, що заснована на її виробничому, технічному та управлінському потенціалі і орієнтована на підвищення ефективності процесу.

Ця модель надає можливість визначити поточний стан ПРПЗ використовуючи рівні зрілості. Рівень зрілості є критерієм якості ПРПЗ. Чим вище рівень зрілості – тим більш якісним є процес розробки ПЗ.

Недоліком є те, що дана модель до теперішнього часу не отримала належної формалізації, що робить неможливим використання математичних методів для визначення якості ПРПЗ (визначення рівня зрілості), та шляхів його покращення

(шляхи збільшення рівня зрілості), а також не має можливості врахувати наявні ресурси для досягнення того чи іншого рівня зрілості та його підтримки.

Зважаючи на проблему, що описана вище, постає задача керування якістю ПРПЗ в умовах обмежених ресурсів на основі моделі зрілості.

Дана задача складається з двох під задач, а саме:

– задача побудови моделі оцінки якості ПРПЗ на основі моделі зрілості CMMI;

– задача покращення якості ПРПЗ на основі моделі зрілості.

Кінцевою метою даної роботи є вирішення задачі синтезу моделі оцінки якості процесу розробки програмного забезпечення, а також проектування програмної системи яка дозволить побудувати оптимальну траєкторію покращення ПРПЗ, шляхом розробки інформаційної технології, яка включає до себе математичне забезпечення та артефакти проектування програмної системи.

2 ОПИС МАТЕМАТИЧНОЇ МОДЕЛІ ОЦІНКИ ЯКОСТІ ПРОЦЕСУ РОЗРОБКИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Модель управління якістю процесу розробки програмного забезпечення при статичній постановці задачі

Для того, щоб повністю розкрити динамічну постановки задачі, спочатку потрібно розглянути статичну постановку задачі.

Виходячи зі структури моделі CMMI, яка представлена у вигляді ієрархії понять, на самому нижньому рівні знаходяться практики, кожна з яких характеризується певним рівнем можливості [10] .

Відповідно до цього введемо дискретні змінні $x^{j}{}_{is}$ приймають значення цілих чисел, кожне з яких відповідає рівню можливості j-ой практики, яка бере участь в забезпеченні досягнення s-ой мети і-ой фокусної області. У свою чергу змінна:

$$y_{is} = f_{is}\left(\left\{x^{j}{}_{is}\right\}\right) \tag{2.1}$$

визначає рівень можливості s-ой мети і-ой фокусної області, де is f - функція перетворення значень рівнів можливостей приватних практик, що забезпечують досягнення s-ой мети і-ой фокусної області. Сукупність цілей і-ой фокусної області визначає її рівень можливості:

$$Z_i = F_i\left(\left\{y_{is}\right\}\right) \tag{2.2}$$

Згідно з ієрархією понять моделі CMMI введемо такі позначення: J^{s}_{i} - множина практик s-ой мети і-ой фокусної області; S_i – множина цілей і-ой фокусної області; l^{i}_{k} - множина фокусних областей належать l-ой категорії k-го рівня зрілості; L_k - множина категорій k-го рівня зрілості; K - множина рівнів зрілості.

Кожна фокусна область вносить свій внесок у досягнення деякого рівня зрілості ПР ПО організації. Тому приватним критерієм оцінки рівня зрілості ПР ПО вважатимемо ступінь досягнення заданого «якості» реалізації фокусної області. Одним з підходів оцінки альтернатив безлічі Ω є побудова окремих приватних функцій корисності і на їх основі узагальнених функцій корисності (відповідних узагальненими критеріями, розглянутим вище) в рамках теорії корисності. Якщо в цьому є необхідність, будується скалярна функція корисності.

У нашому випадку критерії z_i мають різну розмірність, так як описують рівень рішення різних завдань (рівень досягнення різних цілей). Тому вони не зрівнювальні між собою і їх необхідно нормалізувати. Надалі такі нормалізовані приватні критерії будемо називати функціями корисності часткових критеріїв або приватними функціями корисності, кожна з яких має відповідати таким вимогам: мати однаково інтервал вимірювання (в нашому випадку [0,1]), бути безрозмірною і інваріантної до виду екстремуму приватного критерію. Це означає, що оптимальне значення має відповідати максимальному, рівному одиниці, і найгірше - мінімального, рівного нулю значенням приватної функції корисності [11]. Крім цього приватна функція корисності повинна мати можливість реалізовувати як лінійні, так і нелінійні опуклі вгору і вниз неспадні залежності корисності від приватного критерію. З огляду на це, будемо використовувати фундаментальну властивість систем [12], де говориться, що ефективність (корисність) будь-якої системи від вкладених ресурсів на всьому інтервалі життєвого циклу якісно може бути описано логістичної кривої, яка має S-подібний характер. Так як на аналізованому періоді зазвичай ЛПР оперує з певним інтервалом зміни значень приватного критерію, то можна говорити і про різний вигляд функції корисності.

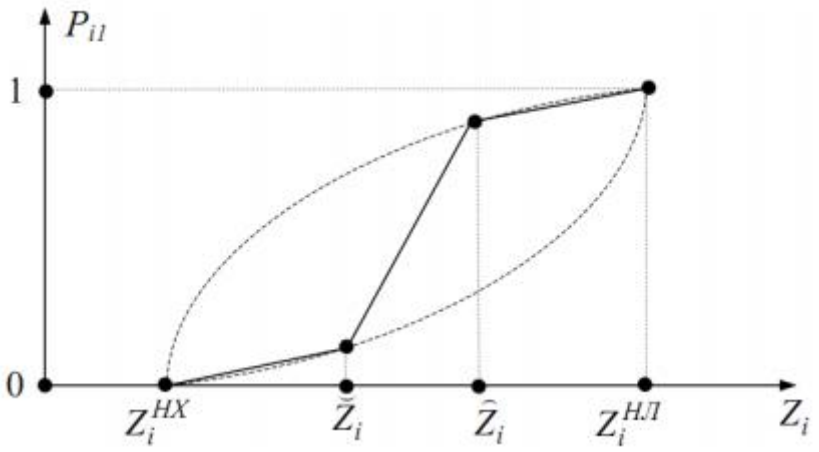

Рисунок 2.1 – Опис функції корисності логістичної кривої

Таким чином, в залежності від області зміни змінних моделі на основі логістичної кривої може бути синтезована аналітична функція корисності часткових критеріїв ступеня досягнення цільового профайла.

Представимо функцію корисності в наступному виді:

$$P_{il}\left(Z_i\left(\{x_{is}^j(t)\}\right)\right) = \left(\frac{z_i\left(\{x_{is}^j(t)\}\right) - z_i^{НГ}}{z_i^{НК} - z_i^{НГ}}\right)^{\alpha_i}, \quad (2.3)$$

де $Z_i^{НК}$, $Z_i^{НГ}$ – найкраще та найгірше, відповідно, і-го критерію. $Z_i^{НГ}$ визначається початковим станом ПР ПЗ $\{x_{is}^j(0)\}$. $Z_i^{НК}$ визначається на основі цільового профайлу. Параметр α_i визначає вид залежності функції корисності. Ця залежність опукла вгору при умові, що $0 < \alpha_i < 1$, лінійна при $\alpha_i = 1$; опукла вниз при $\alpha_i > 1$, як зображено на рисунку 2.2.

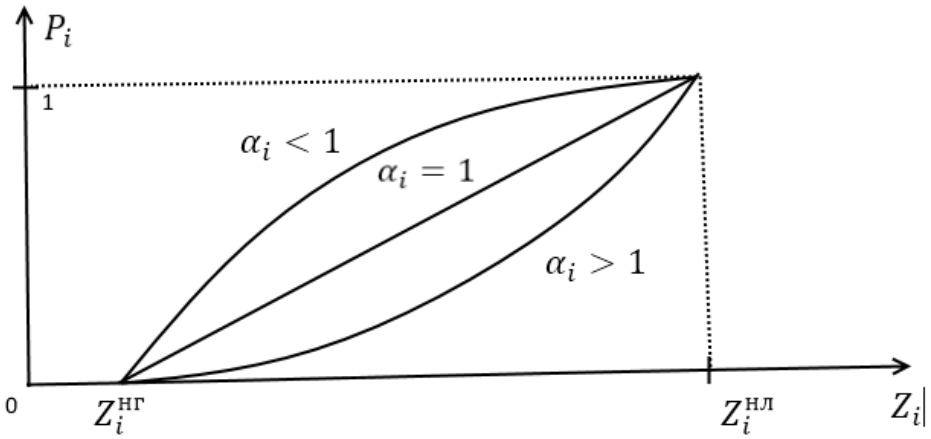

Рисунок 2.3 – Функція корисності частного критерія першої групи

Розглядається два типа ресусів: фінансові затрати та час. Перший критерій базується на двух складових:

– необхідні витрати на додаткове технічне. Інформаційне програмне та методичне забезпечення;

– фінансові витрати на оплату праці співробітників компанії.

Другий критерій фактично зв’язан з другою складовою першого критерію та визначається часом, який необхідно витратити працівнику компанії, за яке працівник не буде займатися основним видом діяльності, зв’язаним з розробкою ПЗ.

Перші дві групи критеріїв противоречиві, так як чим більше вкладено ресурсів, тим в більшій степені можуть бути досягнені цілі та навпаки, тому виникає задача визначення раціонального (компромісного) рішення.

Для визначення об’єму необхідних ресурсів необхідно знати два стани і-ї фокусної області $\{x_{is}^t(t-1), j \in J^s, s \in S_i\}$ та $\{x_{is}^t(t), j \in J^s, s \in S_i\}$.

При цьому для кожної j-ї практики і-ї фокусної області введемо множину булевих змінних $\left\{\lambda_{is}^j(v), v \in \left[x_{is}^t(t-1), \overline{x_{is}^J}\right]\right\}$ та множину значень фінансових ресурсів $(\tau = 2)$ та ресурсів часу $(\tau = 3)$ $\left\{r_{is}^{j\tau}(v), v \in \left[x_{is}^t(t-1), \overline{x_{is}^J}\right]\right\}, \tau = 2,3$, необхідних для переходу в новий стан j-ї практики і-ї фокусної області. При цьому

$\lambda_{is}^{j}(v)$ дорівнює одиниці, якщо рівень практики приймає значення, рівне v та ноль в іншому випадку.

На основі введених позначень необхідне ресурсне забезпечення для кожної практики при переході на новий рівень можливості має вигляд:

$$\bar{R}_{is}^{j\tau}\left(\left\{\lambda_{is}^{j}(v)\right\}\right)=\sum_{v=x_{is}^{j}(t-1)}^{\bar{x}_{is}^{j}} r_{is}^{j\tau}(v)\cdot\lambda_{is}^{j}(v),\tau=2,3, \tag{2.4}$$

$$\sum_{v=x_{is}^{j}(t-1)}^{\bar{x}_{is}^{j}}\lambda_{is}^{j}(v)=1 \tag{2.5}$$

За аналогією з першою групою критеріїв функції корисності для другої групи запишемо в наступному вигляді:

$$P_{i\tau}\left(\left\{\lambda_{is}^{j}(v)\right\}\right)=\left(\frac{R_{i\tau}\left(\left\{\lambda_{is}^{j}(v)\right\}\right)-R_{i\tau}^{\text{НГ}}}{R_{i\tau}^{\text{НК}}-R_{i\tau}^{\text{НГ}}}\right)^{\alpha_{i\tau}},\tau=2,3, \tag{2.6}$$

Де P_{i2}, P_{i3} - функції корисності, відповідно для першого та другого видів ресурсі. Ці функції опуклі вгору при умові, що $0<\alpha_i<1$, лінійні при $\alpha_i=1$; опуклі вниз при $\alpha_i>1$.

Беручи до уваги статичну постановку задачу, далі змінні $x_{is}^{t}(t)$ будемо записувати як x_{is}^{t}. Функція корисності степені досягнення цілі окремої категорії зрілості записується наступним чином:

$$Z_i=\sum_{s\in S_i}\beta_{is}y_{is},\sum_{s\in S_i}\beta_{is}=1,i\in I_k^l,l\in L_k,k\in K. \tag{2.7}$$

Перейдемо до формування двох підгруп корисності ресурсного забезпечення. Функція корисності фінансових витрат та витрат часу має наступний вигляд:

$$W_\tau^l\left(\left\{\lambda_{is}^j(v)\right\}\right)=\sum_{k\in K}\delta_{k\tau}^l Q_{k\tau}^l\left(\left\{\lambda_{is}^j(v)\right\}\right),\sum_{k\in K}\delta_{k\tau}^l=1, l\in L_k, L\bigcap_{k\in K}L_k, \tau=2,3.$$

Таким чином векторна цільова функція моделі задачі планування покращення ПР ПЗ при статичній постановці задачі має наступний вигляд:

$$\Phi\left(\left\{x_{is}^j,\lambda_{is}^j(v)\right\}\right)=\left\{\widehat{W}_1^l\left(\left\{x_{is}^j\right\}\right),\widehat{W}_2^l\left(\left\{\lambda_{is}^j(v)\right\}\right),\widehat{W}_3^l\left(\left\{\lambda_{is}^j(v)\right\}\right), l\in L\right\}. \quad (2.8)$$

Обмеження можна розділити на 3 групи.

1 Обмеження, які накладаються на змінні цільової функції моделі

$$x_{is}^j\in N,\lambda_{is}^j(v)\epsilon\{0,1\}, \quad (2.9)$$

$$\bar{\bar{x}}_{is}^j\le x_{is}^j\le\bar{x}_{is}^j, \quad (2.10)$$

$$\sum_{v=\bar{\bar{x}}_{is}^j}^{\bar{x}_{is}^j}\lambda_{is}^j(v)=1. \quad (2.11)$$

2 Обмеження, які зв'язують дискретні та булеві змінні

$$x_{is}^j=\sum_{v=\bar{\bar{x}}_{is}^j}^{\bar{x}_{is}^j}\left(v\cdot\lambda_{is}^j(v)\right). \quad (2.12)$$

3 Обмеження на фінансові ресурси

$$\sum_{k\in K}\sum_{l\in L}\sum_{i\in I_k^l}R_2\left(\left\{\lambda_{is}^j(v)\right\}\right)\le R, \quad (2.13)$$

де R – фінансові ресурси, виділені на першому підперіоді планування.

Задача многокритеріальної оптимізації з врахуванням умов, які накладаються на вагові коефіцієнти важливості, в залежності від цілі може маті різні постановки. В роботі пропонується розглянути наступну постановку: необхідно знайти компромісне рішення відносно різних категорій моделі CMMI та двох груп функцій корисності.

Оптимальне рішення для кожної цільової функції корисності відповідає одиниці.

Перейдемо до розгляду методів та алгоритмів, які доцільно використовувати при вирушенні поставлених задач. Для цього проведемо аналіз розмірності задачі: допустим що розглядається задача виходу ПР ПЗ на третій рівень зрілості та всі практики фокусних областей відповідні другому рівню та третьому рівню мають першій рівень зрілості. В цьому випадку усі практики цільового профайлу повинні маті третій рівень зрілості. Виходячи с цього кількість дискретних цілочисельнних змінних дорівнює 140, а кількість булевих змінніх дорівнює 540. Для практичної постановки задачі можна сказати, що це максимальна розмірність задачі. Виходячи з проведеного аналізу методів рішення такого класу задача математичного програмування може бути використан метод гілок та границь.

Розглянемо біль детальніше поняття компромісного рішення. На першому етапі будемо вважати, що обмеження на фінансові ресурси відсутні. В цьому випадку $V_2 = 0$ це відповідає розміру використовуємих ресурсів, які забезпечують досягнення поставленої цілі. При цьому $V_1 = 1$. Припустимо, що ми робимо поступку ΔV_2 по функції корисності V_2 та розглядаємо задачу

$$V_1 = \left(\{x_{is}^j\}\right) \to max. \qquad (2.14)$$

Необхідно відзначити, що чим більше V_2, тим менше ресурсів ми можемо задіяти. Збільшуючи величину поступки та вирішуючи подібні задачі ми отримаємо ломану ABCDE, як зображено на рисунку 2.4.

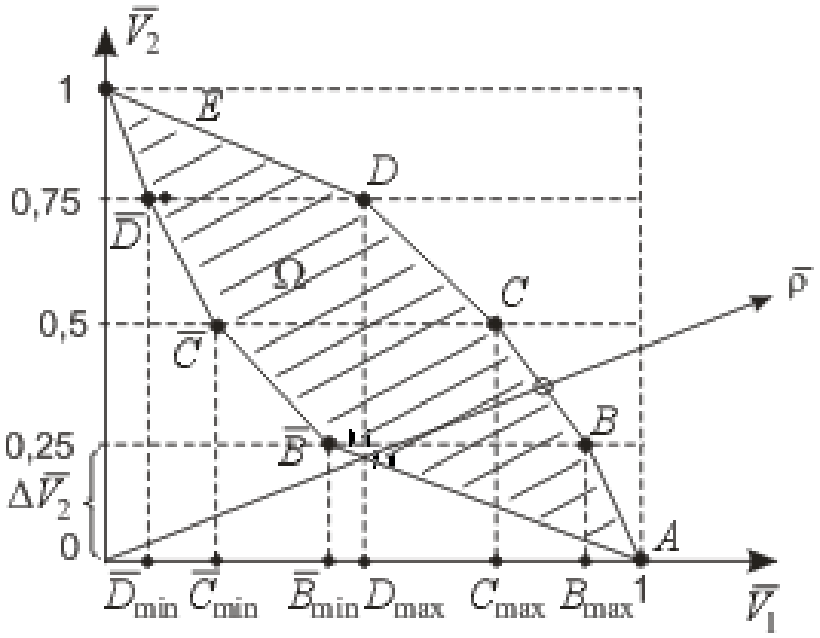

Рисунок 2.4 – Область допустимих значень рішень в просторі функції корисності

Збільшуючи величину поступки по V_2 ми отримуємо іншу ломану ABCDE та область допустимих значень функцій V_1 та V_2.

Необхідно знайти компромісне рішення відносно різних категорій моделі CMMI та двух груп фінкцій корисності. Перша група це $\left\{\widehat{W_1^l}\left(\left\{x_{is}^j\right\}\right)\right\}$ та друга група

$$\widehat{W_2^l}\left(\left\{\lambda_{is}^j(v)\right\}\right) = \mu_2^l W_2^l\left(\left\{\lambda_{is}^j(v)\right\}\right) + \mu_3^l W_3^l\left(\left\{\lambda_{is}^j(v)\right\}\right), \mu_2^l + \mu_3^l = 1, l \in L, \quad (2.15)$$

де $\{\mu_2^l\},\{\mu_3^l\}$ – вагові коефіцієнти важливості окремих цільових функцій корисності другої групи.

Оптимальне рішення для кожної цільової функції корисності відповідає одиниця. Пі компромісів, розуміється однакове відхилення значення всіх цільових функцій корисності з урахуванням їх коефіцієнтів важливості. Тому, в цьому випадку вирішується задача.

$$K_0 \to \max_{\{x_{is}^j\},\{\lambda_{is}^j(v)\}} \tag{2.16}$$

при умовах

$$\hat{\rho}_1^l \widehat{W_2^l}(\{x_{is}^j\}) = \hat{\rho}_2^l W_2^l(\{\lambda_{is}^j(v)\}) = K_0, l \in L, \sum_{i=1}^2 \sum_{l \in L} \hat{\rho}_i^l = 1. \tag{2.17}$$

2.3 Модель управління якістю процесу розробки програмного забезпечення при динамічній постановці задачі

Розглянемо питання побудови динамічної моделі управління якістю процесу розробки ПЗ на основі технології «модель зрілості» з урахуванням основних принципів і припущень, які були наведені в п. 2.1.

У даній роботі в якості основного критерію будемо використовувати інтегральний показник, пов'язаний зі збільшенням рівня зрілості ПР ПО протягом аналізованого планового періоду [13]. При цьому будемо вважати, що ступінь важливості збільшення всіх p -х рівнів зрілості, де $p = \overline{k,5}$, визначається вектором параметрів $\{\lambda_p\}$, які задовольняють наступним умовам:

$$\lambda_p \geq 0,\ \ p = \overline{k,5};\ \sum_{p=k}^{5} \lambda_p = 1 \tag{2.18}$$

Тоді інтегральний показник ступеня досягнення рівня зрілості ПР ПО на t-ому підперіоді планування визначається наступним чином

$$\mu^t\left(\left\{x_{ij}\right\}^t\right)=\sum_{p=k}^{5}\lambda_p\prod_{s=k}^{p}\omega_s^t\left(\left\{x_{ij}\right\}^t\right). \tag{2.19}$$

Надалі сукупність значень приватних практик в t-ому підперіоді планування позначимо матрицею змінних $\left\{x_{ij}\right\}^t$.Тоді приріст рівня зрілості при переході з$t-1$ -го наt - ий підпериод планування визначається наступним чином

$$\overline{\Phi}_t(\chi_{t-1},\chi_t)=\mu^t(\chi_t)-\mu^{t-1}(\chi_{t-1}),\ t=\overline{1,T}\ . \tag{2.20}$$

Крім цього, введемо поняття ступеня важливості приросту рівня зрілості на t-ом підперіоді планування шляхом введення вагових коефіцієнтів важливості

$$\xi_t\geq 0,\ t=\overline{1,T},\ \sum_{t=1}^{T}\xi_t=1\ . \tag{2.21}$$

Тоді приріст рівня зрілості напишемо в наступному вигляді

$$\Phi_t(\chi_{t-1},\chi_t)=\xi_t\overline{\Phi}_t(\chi_{t-1},\chi_t)\ ,\ t=\overline{1,T}\ . \tag{2.22}$$

У підсумку цільова функція, що визначає інтегральний показник ступеня збільшення рівня зрілості на всьому плановому періоді з урахуванням ступеня важливості кожного підперіоди і ступеня важливості досягнення окремим p-м рівнем певною мірою рівня зрілості, де, записується таким чином

$$F(\chi)=\sum_{t=1}^{T}\Phi_t(\chi_{t-1},\chi_t) \tag{2.23}$$

Конкретний вид моделі (цільова функція і обмеження) завдання управління якістю ПР ПО залежить від стратегії особи, що приймає рішення (ОПР) стосовно послідовності досягнення відповідного рівня зрілості організації.

θ - Змінна, яка визначає на якісному рівні (більше, менше) значення рівнів можливості практик, що належать фокусною областям відповідних рівнів зрілості ПР ПЗ.

Припустимо, досягнутий перший рівень зрілості (вважається, що він завжди досягнутий). Це означає, що всі практики фокусних областей мають рівень можливості не менше $\bar{\theta}_1$. Для досягнення другого рівня зрілості необхідно, щоб всі практики фокусних областей, що відносяться до другого рівня зрілості, мали рівень можливості не менше $\bar{\theta}_2$, а рівні можливості практик фокусних областей, що відносяться до першого рівня зрілості, підтяглися мінімум до $\bar{\theta}_2$. Далі, якщо мета стоїть досягти третій рівень зрілості, то практики

відповідних фокусних областей повинні мати рівень можливості не менше $\bar{\theta}_3$, а рівень можливості практик фокусних областей, що відносяться до першого і другого рівнями зрілості необхідно підтягнути мінімум до рівня можливості $\bar{\theta}_3$ і т.д. Припустимо, деякий $k-1$-й рівень зрілості досягнутий. Тоді можливі два підходи.

1. ОПР на деякому плановому періоді планує досягти k-й рівень зрілості і далі в порядку черговості наступні рівні.

2. ОПР виставляє пріоритети по відношенню до фокусною областям різних рівнів зрілості, і завдання вирішується паралельно для деякої підмножини рівнів зрілості або відразу для всіх.

Перший підхід фактично є інтерпретацією лексикографічної задачі, коли абсолютний пріоритет виставляється для k-го рівня зрілості по відношенню до решти, більш старшим. Коли k-й рівень досягнутий, то абсолютний пріоритет виставляється для $k+1$- го рівня і т.д.

Цільова функція відповідає другому підходу, коли ресурси можуть використовуватися для збільшення ступеня зрілості на кожному рівні з урахуванням їх пріоритетів. При використанні першого підходу на основі досягнутого $k-1$-го рівня зрілості вирішується завдання досягнення тільки k-го рівня і залежності, на основі яких будується цільова функція, значно спрощуються. Так в (2.14) - (2.19) використовуються тільки фіксовані значення k, а умови (2.20), (2.21), (2.22) наводяться до наступного вигляду

$$\mu^t\left(\{x_{ij}\}^t\right)=\omega_k^t\left(\{x_{ij}\}^t\right) \tag{2.24}$$

і в цьому випадку вектор вагових коефіцієнтів $\{\lambda_p\}$ взагалі не використовується.

Надалі модель задачі управління якістю ПР ПО буде розглядатися з урахуванням обмеження на фінансові ресурси і другий критерій перейде в обмеження. В результаті виникає завдання синтезу функції, що визначає витрати на збільшення рівня зрілості ПР ПЗ.

Перейдемо до розгляду питання формування функції витрат на $t-1$-м підперіоді управління, які забезпечують приріст рівня зрілості ПР ПЗ з t-го підперіоду на величину, яка визначається згідно (3.5). З цією метою введемо поняття генеральних трикутних матриць варіантів розвитку приватних практик. Для наочності представимо їх на таблиці 3.1.

Елементи генеральних трикутних матриць визначають необхідні фінансові ресурси при переході практик l-го на s-ий рівень можливості, де l-номер рядка матриці, а s- номер стовпчика. При цьому за визначенням всі елементи головної діагоналі матриці дорівнюють нулю.

Ми можем сформулювати генеральну трикутну матрицю для j-ої приватної практики і-ої фокусної області, яка буде відображати загальну картину щодо практик для певної фокусної області.

Таблиця 3.1 - Генеральна трикутна матриця для j-ої приватної практики i-ої фокусної області

s	m_{ij}	$m_{ij}+1$	$m_{ij}+2$	...	n_{ij}
m_{ij}	$r_{ij}(m_{ij},m_{ij})$	$r_{ij}(m_{ij},m_{ij}+1)$	$r_{ij}(m_{ij},m_{ij}+2)$	...	$r_{ij}(m_{ij},n_{ij})$
$m_{ij}+1$		$r_{ij}(m_{ij}+1,m_{ij}+1)$	$r_{ij}(m_{ij}+1,m_{ij}+2)$	...	$r_{ij}(m_{ij}+1,n_{ij})$
$m_{ij}+2$			$r_{ij}(m_{ij}+2,m_{ij}+2)$	...	$r_{ij}(m_{ij}+2,n_{ij})$
...	...	...	...	...	...
n_{ij}				...	$r_{ij}(n_{ij},n_{ij})$

Якщо вважати, що на періоді $[0,T-1]$ управління будь-яка практика для всіх фокусних областей є потенційним об'єктом вкладу ресурсів, то фінансові витрати, які використовуються на $\tau-1$-м підперіоді управління визначаються наступним чином

$$\overline{R}_{\tau}(\chi_{\tau-1},\chi_{\tau})=\sum_{i\in\bigcup_{s=1}^{5}I_s}\sum_{j\in J_i}r_{ij}\left(x_{ij}^{\tau-1},x_{ij}^{\tau}\right),\ \tau=\overline{1,T}\ . \qquad (2.25)$$

При цьому накладаються умови

$$x_{ij}^{\tau-1}\le x_{ij}^{\tau},\ j\in J_i,\ i\in\bigcup_{s=1}^{5}I_s,\ \tau=\overline{1,T}\ , \qquad (2.26)$$

де

$$x_{ij}^{0}=\tilde{m}_{ij}^{0},\ j\in J_i,\ i\in\bigcup_{s=1}^{5}I_s\ . \qquad (2.27)$$

Якщо ОПР використовує перший підхід до стратегії управління якістю ПР ПЗ, то в залежностях (2.8) - (2.30) замість $\bigcup_{s=1}^{5} I_s$ необхідно використовувати $\bigcup_{s=1}^{k} I_s$

Будемо вважати, що в кожному підперіоді τ на управління розвитком ПР ПО виділяються ресурси в обсязі R_τ і невикористані ресурси на τ-у підперіоді можуть бути витрачені на наступних періодах управління. В результаті ресурсне обмеження записується таким чином

$$\sum_{\tau=1}^{t} \overline{R}_\tau(\chi_{\tau-1}, \chi_\tau) \le \sum_{\tau=0}^{t-1} R_\tau = \hat{R}^{t-1}, \ t = \overline{1,T} \ . \qquad (2.28)$$

В результаті модель задачі управління якістю ПР ПО згідно з технологією «модель зрілості», відповідна другого підходу, записується таким чином. Знайти оптимальне значення матриці, що забезпечує максимальне значення критерію (3.6) при умовах (2.14) - (2.20), (2.21) - (2.25), (2.28) - (2.31).

У тому випадку, якщо використовується перший підхід, то модель спрощується і виглядає наступним чином. Знайти оптимальне значення матриці, що забезпечує максимальне значення критерію (2.26) при умовах (2.14) - (2.19) з фіксованими значенням і умов при заміні $\bigcup_{s=1}^{5} I_s$ на $\bigcup_{s=1}^{k} I_s$.

3 ПРОЕКТУВАННЯ АРХІТЕКТУРИ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Розробка специфікації вимог до програмної системи

3.1.1 Розробка функціональних вимог

У розробці програмного забезпечення та інженерних систем, функціональна вимога визначає функцію системи або її компоненти. Функція описується як набір входів, поведінки і виходів [14].

Функціональні вимоги можуть бути розрахунки, технічні характеристики, маніпуляція і обробка даних і інші конкретні функціональні можливості, які визначають те, що система повинна виконати. Як визначено в інженерії вимог, функціональні вимоги визначають конкретні результати системи. Це необхідно протиставити з нефункціональними вимогами, які визначають загальні характеристики, такі як вартість і надійність.

Для програмного компонента, що розробляється були сформовані наступні функціональні вимоги:

1. Компонент повинен надавати можливість введення даних про поточний стан ПРПЗ та про розподілення фінансових ресурсів на підперіоди.
2. Компонент повинен компонувати оптимальні траєкторії покращення стану ПРПЗ для статичної та динамічної задачі.
3. Компонент повинен відображати результат розрахунку оптимальної траєкторії покращення стану ПРПЗ.
4. Компонент повинен надавати можливість редагування введених даних.
5. Компонент повинен надавати можливість видалення даних.

На основі вищезазначених функціональних вимог була побудована діаграма варіантів використання, що показана на рисунку 3.1.

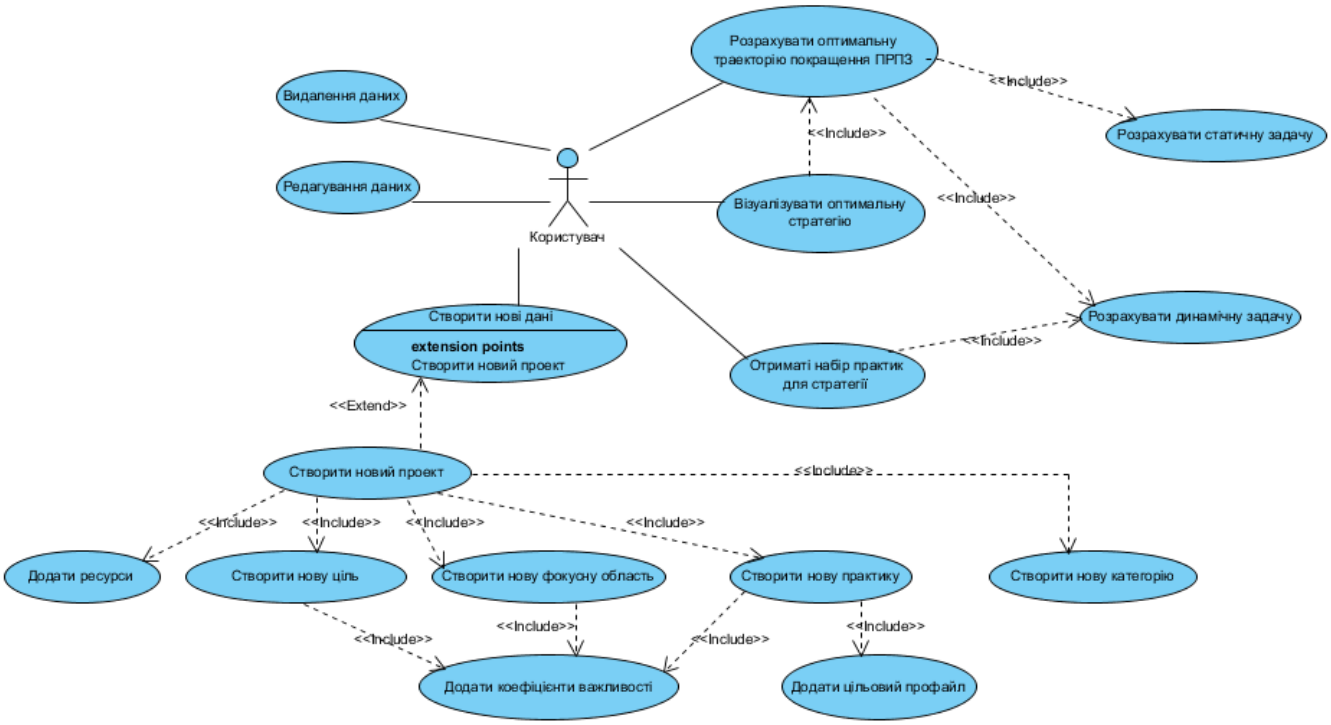

Рисунок 3.1 - Діаграма варіантів використання

Діаграма варіантів використання — діаграма, на якій зображено відношення між суб’єктами та варіантами використання в системі.

Діаграми варіантів використання відіграють важливу роль не тільки у комунікації між збирачами вимог до проекту і потенційними користувачами. Діаграми варіантів використання дописані бізнес логікою і детальними специфікаціями варіантів використання, як джерельна інформація, успішно використовують учасники розробки проекту на всіх його фазах (зародження, дизайн, програмування, тестування, документування.). Добре продумані і завершені специфікації варіантів використання легко перевтілюються у тестові випадки.

Також має сенс сформувати для основного процессу діаграму нотації IDEF0.

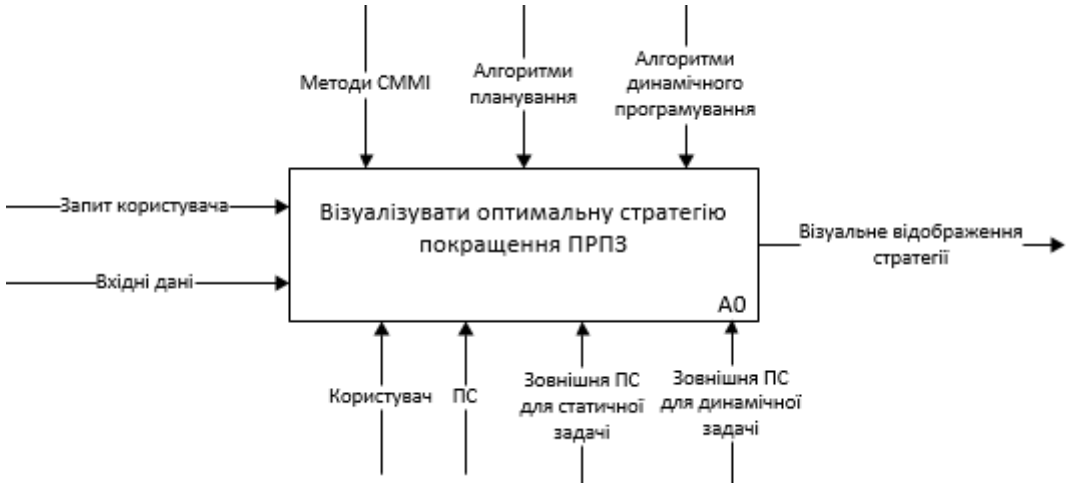

Рисунок 3.2 – Контекстна діаграма IDEF0(для основного процессу)

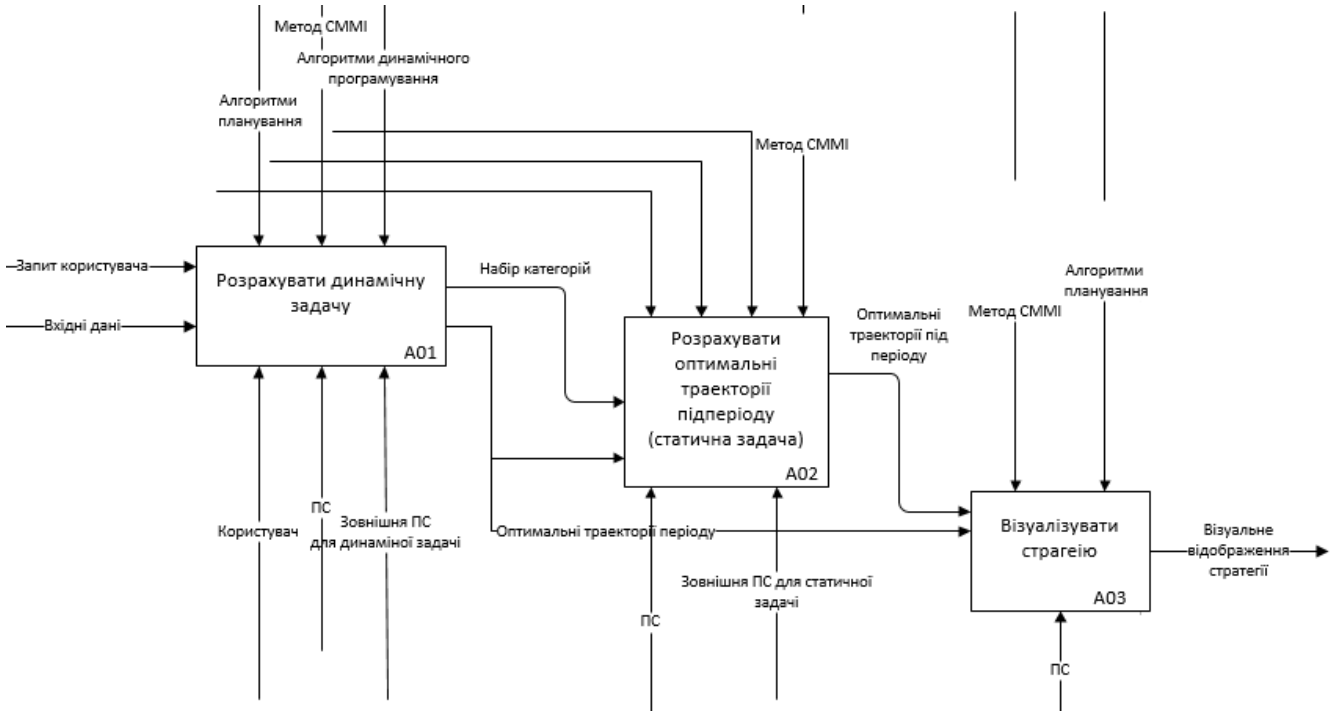

Рисунок 3.3 - Діаграма IDEF0(декомпозиція)

3.1.2 Розробка нефункціональних вимог

Нефункціональні вимоги – вимоги, що представляють атрибути якості ПЗ (продуктивність, надійність, безпека, зручність користування, переносимість, супроводжуваність), котрі повинні бути досягнуті в системі, що розроблюється за умовою, що раніше сформульовані функціональні вимоги будуть виконані [14].

Для програмного компонента, що розробляється були сформовані наступні нефункціональні вимоги:

1 Зручність використання інтерфейсу користувача. Найголовнішим завданням інтерфейсу є показати стан ПРПЗ та криву його поліпшення, тому відображення всіх даних повинно бути на високому рівнім.

2 Розширюваність. Компонент повинен бути спроектованим таким чином, щоб була можливість розширювати його можливості без будь – якого помітного впливу на системи, що будуть його використовувати.

3 Компонент повинен бути спроектованим таким чином, щоб була можливість розширювати його можливості без будь – якого помітного впливу на системи, що будуть його використовувати.

4 Повинна бути документація щодо побудови програмного коду.

3.2 Моделювання даних

3.2.1 Побудова онтології проекту

На основі прецедентів та опису процесу програмної системи було розроблено онтологію предметної області, яка зображена на рисунку 3.4 у вигляді діаграми класів UML.

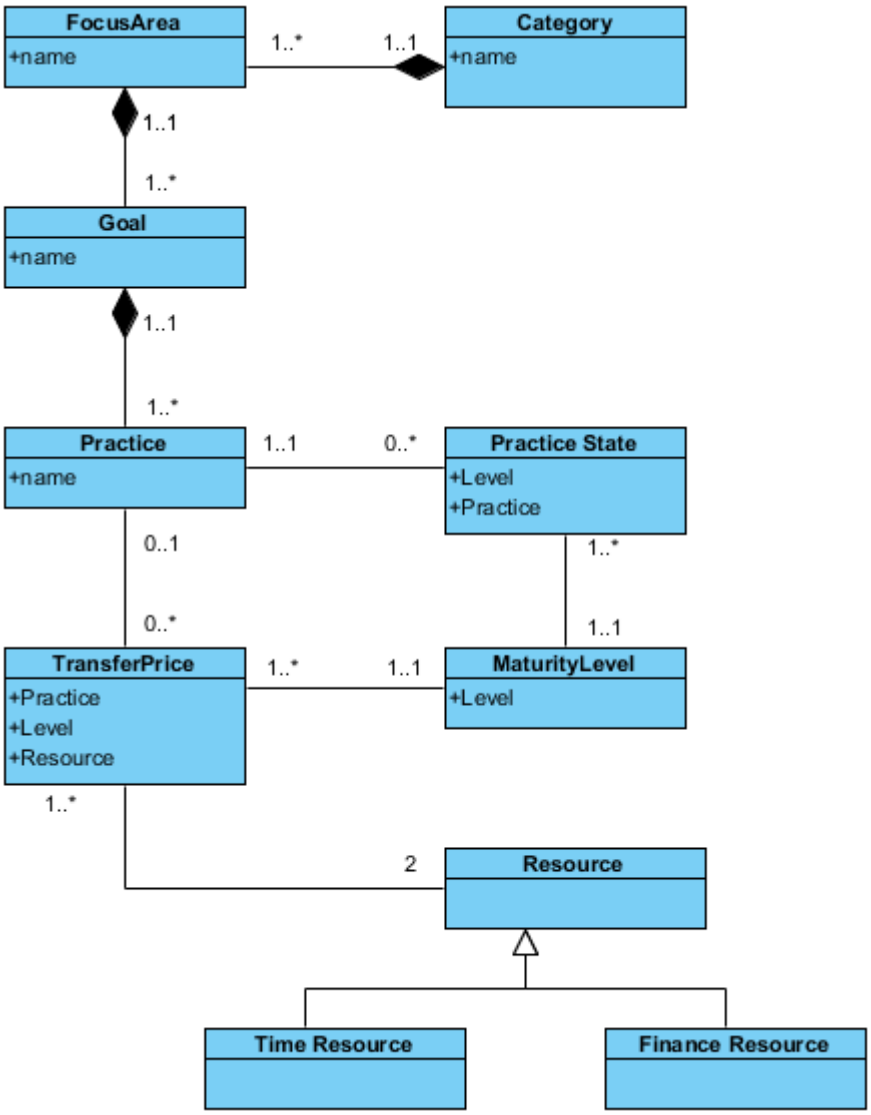

Рисунок 3.4 – Онтологія проекту

3.2.2 Розробка концептуальної моделі бази даних

Логічна модель описує поняття предметної області, їх взаємозв'язок, а також обмеження на дані, що накладаються предметною областю. Основними компонентами такої моделі є сутності, їх атрибути і зв'язки між ними.

Логічна модель даних є початковим прототипом майбутньої бази даних. Логічна модель будується в термінах інформаційних одиниць, але без прив'язки до конкретної СКБД.

ERD-діаграма дозволяє розглянути систему цілком і з'ясувати вимоги, необхідні для її розробки, що стосуються для зберігання інформації. ERD-діаграми

можна підрозділити на окремі частини, відповідні окремих завдань, розв'язуваних проектованої системою.

Це дозволяє розглядати систему з точки зору функціональних можливостей, роблячи процес проектування керованим .

На рисунку 3.5 зображена концептуальна модель бази даних.

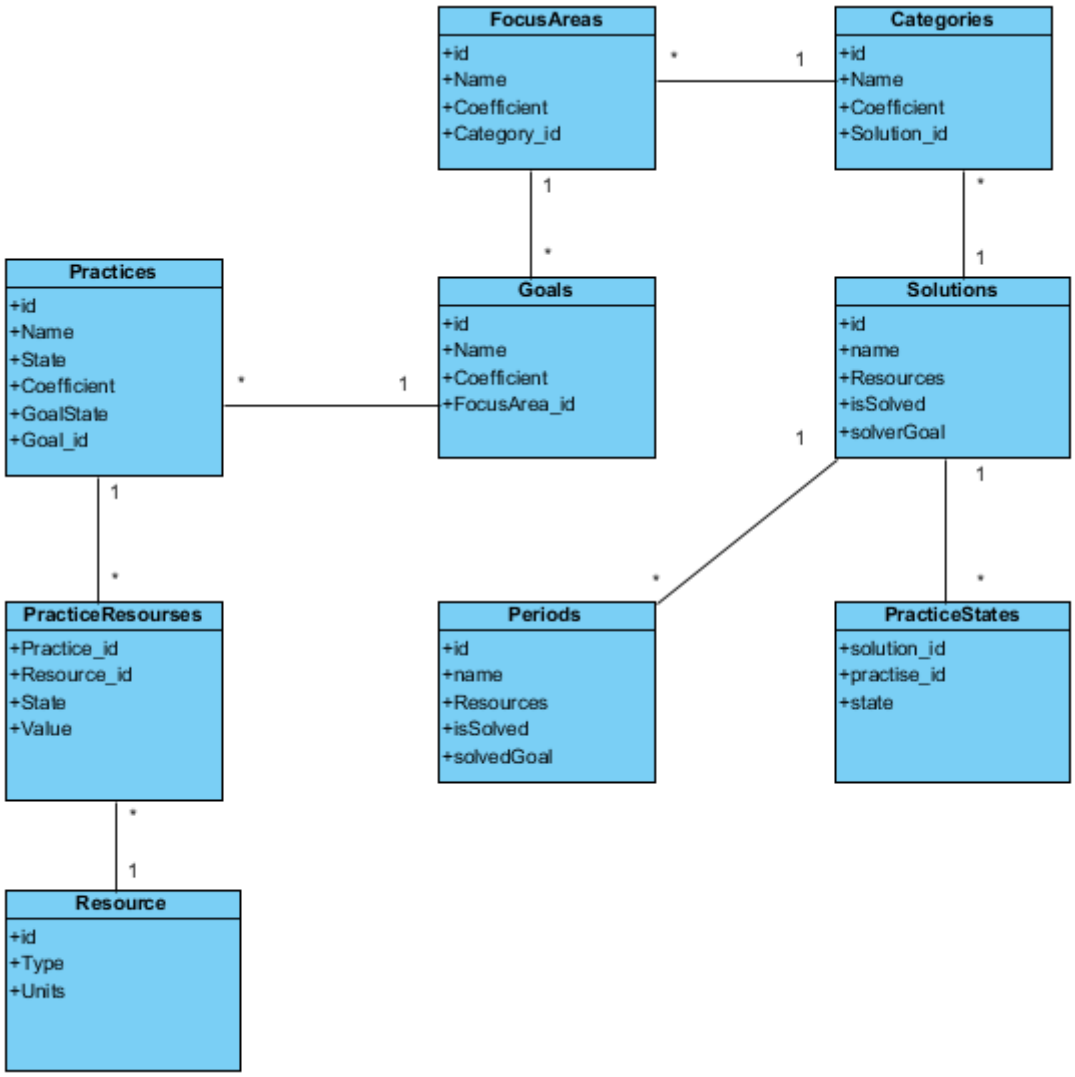

Рисунок 3.5 – Концептуальна модель бази даних

Опис сутностей бази даних відображено в таблицях 3.1-3.12.

Таблиця 3.1 – Сутність Category

Category	Таблиця містить список категорій	
Id	Guid	Унікальний ідентифікатор категорій
Name	Nvarchar(50)	Назва категорій

Таблиця 3.2 – Сутність FocusArea

FocusArea	Таблиця містить список фокусних областей	
Id	Guid	Унікальний ідентифікатор фокусних областей
Coefficient	double	Коефіцієнт важливості фокусної області.
CategoryId	Guid	Зовнішній ключ таблиці Category
Name	Nvarchar(50)	Назва фокусної області

Таблиця 3.3 – Сутність Goal

Goal	Таблиця містить список цілей	
Id	Guid	Унікальний ідентифікатор цілі
Coefficient	double	Коефіцієнт важливості цілі.
FocusAreaId	Guid	Зовнішній ключ таблиці FocusArea
Name	Nvarchar(50)	Назва цілі

Таблиця 3.4 – Сутність Category

Category	Таблиця містить список категорій	
Id	Guid	Унікальний ідентифікатор категорій
Name	Nvarchar(50)	Назва категорій

Таблиця 3.5 – Сутність FocusArea

FocusArea	Таблиця містить список фокусних областей	
Id	Guid	Унікальний ідентифікатор фокусних областей
Coefficient	double	Коефіцієнт важливості фокусної області.
CategoryId	Guid	Зовнішній ключ таблиці Category
Name	Nvarchar(50)	Назва фокусної області

Таблиця 3.6 – Сутність Goal

Goal	Таблиця містить список цілей	
Id	Guid	Унікальний ідентифікатор цілі
Coefficient	double	Коефіцієнт важливості цілі.
FocusAreaId	Guid	Зовнішній ключ таблиці FocusArea
Name	Nvarchar(50)	Назва цілі

Таблиця 3.7 – Сутність Resource

Resource	Таблиця містить список типів ресурсів	
Id	Guid	Унікальний ідентифікатор типу ресурсу.
Type	nvarchar(50)	Назва типу ресурсу.
Units	nvarchar(50)	Одиниці вимірювання ресурсу.

Таблиця 3.8 – Сутність Practice

Practice	Таблиця містить список практик	
Id	Guid	Унікальний ідентифікатор практики
Coefficient	double	Коефіцієнт важливості практики.
GoalId	Guid	Зовнішній ключ таблиці Goal
Name	Nvarchar(50)	Назва практики
StartState	int	Початковий стан практики

Таблиця 3.9 – Сутність Period

PracticeResource	Таблиця містить список ресурсів практик	
PeriodId	Guid	Ключ таблиці Period
SolutionId	Guid	Зовнішній ключ таблиці Solution
FinancialResources	int	Виділенні фінансові ресурси
TimeResource	int	Виділенні часові ресурси

Таблиця 3.10 – Сутність Practice

Practice	Таблиця містить список практик	
Id	Guid	Унікальний ідентифікатор практики
Coefficient	double	Коефіцієнт важливості практики.
Name	Nvarchar(50)	Назва практики
StartState	int	Початковий стан практики
GoalState	int	Цыльовий стан практики

Таблиця 3.11 – Сутність PracticeResource

PracticeResource	Таблиця містить список ресурсів практик	
PracticeId	Guid	Зовнішній ключ таблиці Practice
ResourceId	Guid	Зовнішній ключ таблиці Resource
PracticeState	int	Стан практики
Value	double	Кількість ресурсів, необхідних на переход до вказаного стану практики

3.2.3 Побудова системи бізнес-правил та глосарію проекту

Хоча програмний компонент на має своєї бази даних, він повинен відсилати та отримувати об’єкти, які зберігаються у зовнішніх базах даних. Тому є доцільним побудувати систему бізнес-правил та глосарій предметної області, для того щоб реалізувати взаємодію з цією базою даних.

Система бізнес – правил (СБП) представляє собою текстовий опис на природній мові, який складається розробником моделі даних в процесі спілкування з носіями інформації про дану предметну область. СБП повинна бути логічно не суперечливою, ні надлишковою та достатньо повною, щоб описувати усі основні інформаційні об’єкти та їх взаємозв’язок і дозволяти потім застосовувати до них процедури узагальнення та агрегації.

Для предметної області «модель оцінки якості ПРПЗ на основі технології моделі зрілості» була розроблена наступна СБП:

1) кожна фокусна область належить тільки до одного рівня зрілості;

2) кожний стан практики належить тільки одному стану ПРПЗ;

3) кожний стан практики може мати один рівень можливості;

4) кожний рівень можливості належить багатьом станам практик;

5) кожний коефіцієнт важливості практики належить одному підперіоду покращення;

6) кожний стан практики належить тільки до однієї практики;

7) кожна практика має багато станів практики;

8) кожна практика належить до однієї цілі;

9) кожна ціль має безліч практик;

10) кожна ціль належить до однієї фокусної області;

11) кожна фокусна область має багато цілей;

12) кожна фокусна область належить одній категорії;

13) кожна категорія має багато фокусних областей;

14) кожний підперіод покращення має багато коефіцієнтів важливості фокусної області;

15) кожний період покращення має багато станів ПРПЗ;

16) кожний стан ПРПЗ належить одному періоду покращення;

17) кожний підперіод покращення має множину коефіцієнтів важливості практики.

Далі, щоб формалізувати предметну область пропонується навести глосарій основних змістовних понять.

Глосарій проекту – словник основних термінів, які є найбільш значимі для СБП даної ПРО. Глосарій складається з повного імені, скороченого означення, смислової інтерпретації та синонімів, до терміну, що розглядається. В таблиці 3.1 наведений глосарій проекту.

Таблиця 3.12 – Глосарій проекту.

Повне ім’я	Скорочене означення	Смислова інтерпретація	Синоніми
Процес розробки програмного забезпечення	ПРПЗ	Процес створення ПЗ який складається з областей діяльності (фокусних областей).	
Фокусна область	ФО	Визначає кластер дій, внаслідок яких досягається безліч цілей, важливих з погляду збільшення продуктивності ПРПЗ	
Ціль фокусної області	Ціль (в контексті, що розглядається)	Цілі резюмують основні дії ФО і можуть використовуватися для визначення ефективності впровадження організацією.	

Продовження таблиці 3.1

Повне ім’я	Скорочене означення	Смислова інтерпретація	Синоніми
Практика	П	Практика описує інфраструктуру і дії, які в більшості випадків необхідні для успішної реалізації процесної області.	
Рівень зрілості	РЗ	Рівень зрілості – це точно визначене еволюційне плато на шляху до досягнення повної зрілості бізнес-процесів організації	Зрілість (в контексті, що розглядається)
Рівень можливості	РМ	Рівень можливості описує досягнення певних характеристик окремою фокусною області чи практикою.	Можливість (в контексті, що розглядається)
Коефіцієнт важливості	КВ	КВ визначає ступінь важливості ФО чи П для досягнення певного рівня зрілості	Важливість (в контексті, що розглядається)
Підперіод покращення	ПП	Визначає підперіод, до якого відноситься певний стан ПРПЗ	

3.3 Вибір цільової системної архітектури

Розглянемо три найбільш використовуваних системних архітектури: «клієнт – сервер», 3-рівневий «клієнт-сервер» з виділеним сервером додатків, «веб – сервіс – орієнтована».

Але для початку визначимо термін системна архітектура.

Існує декілька визначень поняття системна архітектура. Один з найбільш загальних описів системної архітектури є опис наведений нижче.

Архітектура – це кортеж 3-х множин (C, F, I), де C – множина програмних компонент, які реалізують функціональність даної системи; F – множина допустимих форм взаємодії (конфігурацій), в якій можуть бути визначені елементи з множини компонент; I – множини системних інтерфейсів, за допомогою яких множини C та F взаємодіють один з одним, чи звертаються до зовнішніх по відношенню до системи компонентам або іншим системам.

Усі компоненти системних архітектур можна поділити на 3 типа: компоненти сервісу представлення даних (PRS – Presentation Service); компоненти сервісів бізнес-логіки (BLS - Business – Logic Service); компоненти сервісу доступу даних (DAS - Data Access Service).

PRS – сервіси презентації даних під якими розуміємо будь-які програмні рішення, які забезпечують візуалізацію вхідних даних, їх проміжний стан, чи кінцевий результат обробки.

BLS – будь-які функції, об’єкти, які слугують для реалізації алгоритмів обробки даних.

DAS – будь-які програмні рішення та їх структури даних, які забезпечують створення, модифікацію, доступ до даних в базі даних.

В залежності від того як ці три групи сервісів будуть розподілені по вузлах деякої мережевої структури і буде визначатись той чи інший тип системної архітектури.

Діаграма розгортання – відображає фізичні взаємозв’язки між програмними та апаратними компонентами системи, що розробляється. Ця діаграма є добрим

засобом для представлення маршрутів переміщення об’єктів та компонентів в розподіленій системі.

Кожний вузол на діаграмі розгортання являє собою деякий тип обчислювального пристрою – в більшості випадків самостійну частину апаратури. Ця апаратура може бути як прости пристроєм чи датчиком, так і мейнфреймом.

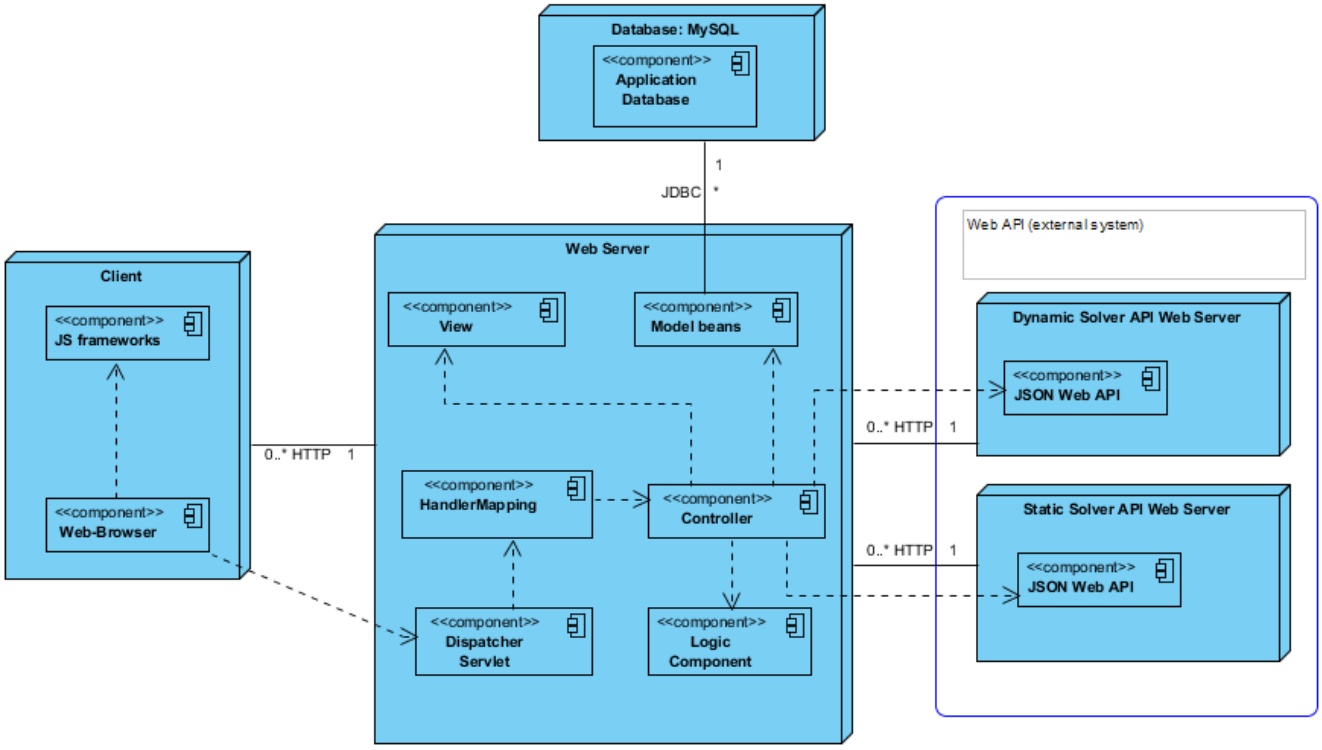

Рисунок 3.5 – Діаграма розміщення компонентів

На стороні клієнта розміщено 2 компоненти:

1) Веб-браузер - програма, яка дає користувачеві доступ до системи;

2) JavaScript Frameworks – JS фреймворки, такі як JQuerry, які необхідні в HTML-сторінкі для реалізації деякої бізнес-логіки.

Вузол "Веб-сервер" - це веб-сервер додатку, клієнт взаємодіє з ним за допомогою протоколу HTTP.

Наступні компоненти розміщені на цьому вузлі:

1) Web API - це компонент, який обробляє запити; відправляє їх на контролер або відправляє їх на зовнішні сервери (Dispatcher Servlet, фронт-контролер);

2) Контролер - приймає вхідні дані і перетворює їх в команди для моделі; посилає до подання необхідних даних;

3) Model Beans - містить бізнес-логіку, формує особливе рішення на основі даних, отриманих з 2 зовнішніх API;

4) View - відображає дані (JSP; Vaadin; SPA);

Основним архітектурним шаблоном був вибраний MVC(Модель-вид-контролер) — архітектурний шаблон, який використовується під час проектування та розробки програмного забезпечення [15].

Цей шаблон поділяє систему на три частини: модель даних, вигляд даних та керування. Застосовується для відокремлення даних (модель) від інтерфейсу користувача (вигляду) так, щоб зміни інтерфейсу користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без змін інтерфейсу користувача.

Мета шаблону — гнучкий дизайн програмного забезпечення, який повинен полегшувати подальші зміни чи розширення програм, а також надавати можливість повторного використання окремих компонентів програми. Крім того використання цього шаблону у великих системах призводить до певної впорядкованості їх структури і робить їх зрозумілішими завдяки зменшенню складності.

Для реалізації програмних компонентів розроблюваної ПС було вирішено використовувати наступні технології і фреймворки.

1 Spring MVC - фреймворк для каркаса веб-додатку.

2 Spring Containers – IoC контейнер.

3 AngularJS - основний JavaScript фреймворк, для реалізації клієнтської логіки.

ВИСНОВКИ

У ході виконання даної роботи був проведений аналіз оцінки якості процесу розробки програмного забезпечення з точку зору моделі зрілості. Була описана модель та алгоритм розрахунку оцінки якості ПРПЗ.

У роботі було розглянуто підходи до оцінки ефективності процесу розробки програмного забезпечення. Також був проведений аналіз оцінки якості процесу розробки програмного забезпечення з точки зору моделі зрілості. Був проведений аналіз архітектури майбутньої програмної системи. На базі розглянутого математичного забезпечення було спроектовано програмне забезпечення.

У результаті було вивчено модель управління якістю ПРПЗ CMMI, алгоритми для вирішення задачі планування покращення якості процесу розробки програмного забезпечення на основі моделі CMMI. В результаті проектування ПЗ було розроблено відповідні UML діаграми.

СПИСОК ДЖЕРЕЛ ІНФОРМАЦІЇ

1 ISO/IEC 12207 [Електронний ресурс]. – 2008. – Режим доступу до ресурсу: https://www.iso.org/obp/ui/#iso:std:iso-iec:12207:ed-2:v1:en, 12.11.2016.

2 Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK), ISO/IEC TR 19759:2005 / Software Engineering — Guide to the Software Engineering Body of Knowledge (SWEBOK), ISO/IEC TR 19759:2005. ISO. – 2005. – 173 p.

3 Cambridge Dictionary [Електронний ресурс] – Режим доступу до ресурсу: http://dictionary.cambridge.org/dictionary/english/software-development, 14.11.2016.

4 Scacchi W. Process models in software engineering / W. Scacchi // J.Marciniak (ed.) Encyclopedia of Software Engineering, 2nd edition. – New York: John Wiley & Sons. – 2001.

5 Boehm B.W. A Spiral model of software development and enhancement / B.W. Boehm // IEEE Software. – 1988. – Vol. 21. – N 5. – P. 61-72.

6 Royce W. Software Project Management—A Unified Approach / W. Royce. – Reading, MA: Addison-Wesley. – 1998.

7 Sommerville I. Інженерія програмного забезпечення / Ian Sommerville. – Williams, 2002. – (6).

8 ISO 9001 [Електронний ресурс]. – 2008. – Режим доступу до ресурсу: http://www.iso.org/iso/ru/iso_9000, 17.11.2016.

9 Chrissis M.B. CMMI: Guidelines for Process Integration and Product Improvement / M.B. Chrissis, M. Konrad, S. Shrum. – Addison-Wesley. – 2003. – 688 p.

10 Годлевский М. Д. Синтез статических моделей планирования улучшения качества процесса разработки программного обеспечения / М. Д. Годлевский, А. А. Голоскокова. // Восточно-Европейский журнал передовых технологий. – 2015. – С. 23–29.

11 Овезгельдыев А. О. Синтез и идентификация моделей многофакторного оценивания и оптимизации [Текст] / А. О. Овез-гельдыев, Э. Г. Петров, К. Э. Петров. – К. : Наукова думка, 2002. – 163 с.

12 Саркисян С. А. Большие технические системы. Анализ и прогноз развития [Текст] / С. А. Саркисян, В. М. Ахундов, Э. С. Минаев. – М. : Наука, 1977. – 350 с.

13 Годлевский М. Д. Динамическая модель планирования улучшения качества процесса разработки программного обеспечения / М. Д. Годлевский, Э. Е. Рубин, А. А. Голоскокова // Вестник Нац. техн. ун-та "ХПИ" : сб. науч. тр. Темат. вып. : Системный анализ, управление и информационные технологии. – Харьков : НТУ "ХПИ". – 2015. – № 58 (1167). – С. 3-6.

14 Wiegers K. Software Requirements / Karl Wiegers. – California: Microsoft Press, 2013. – 673 с.

15 Fowler M. The evolution of MVC and other UI architectures [Електронний ресурс] / Martin Fowler. – 2006. – Режим доступу до ресурсу: http://martinfowler.com/eaaDev/uiArchs.html, 02.12.16.

16 Web Ontology Language [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3.org/OWL, 04.12.16.