UNIVERSITY - INDUSTRY COOPERATION

VOLUME 3

WEB-PORTAL

УНІВЕРСИТЕТСЬКО - ІНДУСТРІАЛЬНА КООПЕРАЦІЯ. ТОМ 3
UNIVERSITY - INDUSTRY COOPERATION. VOLUME 3

2014 2016 CABRIOLET

УНІВЕРСИТЕТСЬКО - ІНДУСТРІАЛЬНА КООПЕРАЦІЯ

ТОМ 3

ВЕБ-ПОРТАЛ

2017

Co-funded by the Tempus Programme of the European Union

Міністерство освіти та науки України
Чернігівський національний технологічний університет
Національний аерокосмічний університет ім. М. Є. Жуковського «ХАІ»

Університетсько-індустріальна кооперація

Том 3

Веб-портал. Настанова з використання

Під редакцією В.С. Харченка

University-industry cooperation

Volume 3

Web-portal. Operational roadmap

TEMPUS CABRIOLET “Model-oriented approach and Intelligent Knowledge–Based System for Evolvable Academia-Industry Cooperation in Electronics and Computer Engineering”
(544497-TEMPUS-1-2013-1-UK-TEMPUS-JPHES)

2017

УДК 378:004
У59

Рекомендовано до видання Вченими Радами Чернігівського національного університету (протокол №1 від 20 вересня 2017 р.) і Національного аерокосмічного університету ім. М. Є. Жуковського «ХАІ» (протокол № 1 від 27.09.2017 р

Рецензенти: Палагін Олександр Васильович, заступник директора Інституту кібернетики Національної академії наук України, академік НАН України, доктор технічних наук, професор;

Сидоренко Микола Федорович, Головний конструктор ДНВП «Об'єднання Комунар» - начальник НТ СКБ «ПОЛІСВІТ» (Харків, Україна), заслужений винахідник України, кандидат технічних наук, доцент.

У59 **Університетсько-індустріальна кооперація.** Веб-портал. Настанова з використання. / Під ред. Кондратенка Ю. П., Харченка В.С. – Міністерство освіти та науки України, Чернігівський національний технологічний університет, Національний аерокосмічний університет ім. М. Є. Жуковського «ХАІ», 2017. Т. 3. – 181 с.

ISBN 978-617-7361-32-8

ISBN 978-617-7361-35-9

Викладено теоретичні основи, принципи побудови та реалізації інструментальних засобів для підтримки кооперації університетів та ІТ-індустрії, створення та розвитку стартап-проектів. Описано етапи створення та особливості функціонування комунікаційного веб-порталу. Надано рекомендації щодо практичного використання при імплементуванні різних моделей кооперації. Матеріали книги підготовлено в рамках проекту TEMPUS CABRIOLET «Model-Oriented Approa**C**h **A**nd Intelligent Knowledge-**B**ased System for Evolvable Academia-Indust**R**y Cooperat**IO**n in E**LE**ctronic and Compu**T**er Engineering» (544497-TEMPUS-1-2013-1-UK-TEMPUS-JPHES).

Для студентів університетів, які навчаються за напрямами інформаційних технологій та починають практичну діяльність, фахівців університетів та ІТ-індустрії, які займаються організацією та підвищенням ефективності університетсько-індустріальної кооперації, а також для викладачів, які проводять заняття з відповідних курсів.

Бібл. – 165 найменувань, рисунків – 22, таблиць – 35.

УДК 378:004

ISBN 978-617-7361-35-9

ПЕРЕЛІК СКОРОЧЕНЬ

ALS	–	Automated learning system
CNUT	–	Chernihiv National University of Technology
GDP	–	Gross domestic product
GTD	–	GettingThingsDone. Methodologyforincreasingpersonaleffectiveness
HMI	–	Human Machine (User) Interface
IT	–	Information technology
PERT	–	Program (Project) Evaluation and Review Technique
PM	–	Project Management
PMBOK	–	Project Management Body of Knowledge
R&D	–	Research and Development
RACI	–	Responsibility Accountable Consulted Informed
SLCM	–	Software Life Circle Model
SMART	–	Self-Monitoring Analysis and Reporting Technology
SW-CMM	–	Software Capability Maturity Model
SWEBOK	–	The Guide to the Software Engineering Body of Knowledge
SWOT-analysis	–	Strengths-Weaknesses-Opportunities-Treats analysis
TTO	–	Technology Transfer Office
UBC	–	University-Business Cooperation
WBS	–	Work Breakdown Structure
ЛМІ	–	Людино-машинний інтерфейс
САН	–	Система автоматизованого навчання

ВСТУП

Багатопланова кооперація університетів та індустрії, розвиток університетських стартап-проектів і стартап-команд потребує відповідної організаційної, методичної та інструментальної підтримки. Така підтримка має охоплювати увесь життєвий цикл кооперації та проектної діяльності. Кооперацію університетських кафедр та ІТ-компаній або окремих їх підрозділів доцільно розглядати як самостійний проект з усіма його атрибутами.

Метою даної книги є представлення теоретичних основ, принципів побудови та реалізації інструментальних засобів університетського бізнес-центру для підтримки кооперації унверситетів та ІТ-індустрії, створення та розвитку стартап-проектів. Крім того, її метою є опис етапів створення та впровадження та особливості функціонування комунікаційного веб-порталу і засобів автоматизованого начання в рамках розвитку кооперації та підприємницької діяльності університетів.

Матеріали книги підготовлено і відповідні розробки виконано за підтримки європейської програми TEMPUS, а саме в рамках проекту CABRIOLET «Model-Oriented ApproaCh And Intelligent Knowledge-Based System for Evolvable Academia-IndustRy CooperatIOn in ELEctronic and CompuTer Engineering» (544497-TEMPUS-1-2013-1-UK-TEMPUS-JPHES).[1]

Консорціум проекту об'єднує університети України – Національний аерокосмічний університет ім. М.Є. Жуковського «Харківський авіаційний інститут», Одеський національний політехнічний університет, Чернівецький національний університет ім. Юрія Федьковича, Чернігівський національний технологічний університет, Чорноморський національний університет ім. Петра Могили, Інститут кібернетики ім. В.М. Глушкова Національної академії наук України, а також університети та ІТ-компанії з Великобританії (Університет Ньюкаслу, координатор та грантхолдер проекту), Іспанії

[1] *Цей проект фінансується за підтримки Європейської комісії. Ця публікація (повідомлення) відображає думки тільки авторів, і Комісія не може нести відповідальність за будь-яке використання інформації, що міститься в ньому.*
This project has been funded with support from the European Commission. This publication (communication) reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

(компанія Inercia Digital), Італії (компанія Critiware), Португалії (Університет Коімбри) та Швеції (Університет КТН, Стокгольм).

За результатами проекту вцілому підготовлено чотиритомне видання:

– **Том 1. Університетсько-індустріальна кооперація. Модельно-орієнтований підхід. Практичне керівництво та приклади (**University-Industry Cooperation. Model-oriented approach. Practical guide and cases);

– **Том 2. Університетсько-індустріальна кооперація. Інтелектуальна знання-орієнтована система прийняття рішень. Вимоги, алгоритми, верифікація і застосування** (University-Industry Cooperation. Intellectual Knowledge-Based Decision Making System: Requirements, Algorithms, Verification and Application);

– **Том 3. Університетсько-індустріальна кооперація. Веб-портал. Настанова з використання** (University-Industry Cooperation. Web-Portal. Operational Roadmap);

– **Том 4. Університетсько-індустріальна кооперація. Нарощування потенціалу. Тренінги** (University-Industry Cooperation. Capacity Building. Trainings).

В рамках третього тому авторами представлено:

- моделі малих підприємств, які утворюються в університетському середовищі, які описують екосистему стартап, його життєвий цикл, методи управління (розділ 1);
- структуру інструментальних засобів університетського бізнес-центру, веб-портал підтримки університетського підприємництва, засобу відбору проектів і команд, моделі оцінки термінів робіт і зрілості компаній, які утворюються (розділ 2);
- моделі та інструментальні засоби автоматизованого навчання в рамках підготовки проектів і команд (розділ 3), а також формалізовані моделі відповідної предметної області (розділ 4).

Книгу підготовлено науковцями кафедр Чернігівського національного технологічний університету доктором технічних наук, професором Литвиновим Віталієм Васильовичем, доктором технічних наук, професором Казимиром Володимиром Вікторовичем, кандидатами технічних наук Посадською Іриною Сергіївною, Савельєвим Максимом

Володимировичем і Гребенник Аллою Григорівною, а також Національного аерокосмічного університету імені М. Є. Жуковського «ХАІ» – доктором технічних наук, професором Вячеславом Сергійовичем і старшим викладачем Ілляшенком Олегом Олександровичем.

О. І. Ілляшенко виконав доопрацювання та верифікацію матеріалів. В. С. Харченком виконано наукове редагування книги.

У розробленні комунікаційного порталу, керівницто з використання якого надано у додатку А, приймали участь студенти Чернігівського національного технологічного університету Іскрижицький Антон Миколайович, Довга Дарина Ігорівна, Гречко Дмитро Сергійович, Заворотний Андрій Олександрович, Зелений Олександр Валерійович, Карпенко Олена Костянтинівна.

Ця книга призначена для:

– студентів університетів, які навчаються за напрямами інформаційних технологій та починають практичну діяльність у галузі ІТ;

– фахівців університетів та ІТ-індустрії, які займаються організацією та підвищенням ефективності університетсько-індустріальної кооперації, виконанням спільних проектів, стартапів, тощо;

– викладачів університетів і тренінг-шкіл, які проводять заняття з відповідних курсів;

– спеціалістів, які займаються створенням і розвитком регіональних інноваційних екосистем.

Автори вдячні рецензентам, колегам по проекту та кафедрам, студентам і аспірантам, фахівцям ІТ-компаній, які приймали безпосередню участь у обговоренні та втіленні результатів проекту.

РОЗДІЛ 1. МОДЕЛІ МАЛИХ ПІДПРИЄМСТВ, ЩО ВИДІЛЯЮТЬСЯ ІЗ АКАДЕМІЧНОГО СЕРЕДОВИЩА

1.1 Академічний стартап

Термін «стартап» був вперше використаний журналами Forbes (1976 г.) і Business Week (1977 р) для позначення компаній з короткою історією операційної діяльності [56]. Даний термін закріпився в мові на порозі XXI ст. в зв'язку з вибуховим зростанням кількості науково-орієнтованих і техно-інноваційних компаній і фактично став синонімом венчурного бізнесу (від англ. venture - ризикований).

Витоки такого бізнесу були закладені в 1950-х роках урядом США з метою наздогнати і перегнати СРСР в світлі передбаченого відставання в науково-технічному розвитку, зокрема, у космічній і ядерній сферах. Для цього був створений спеціальний інститут SBA (Small Business Administration) і прийнятий спеціальний закон про інвестиції в малий бізнес. Держава почала здійснювати пільгове кредитування малого бізнесу, що незабаром привело до проривів в сфері напівпровідників і біотехнологій в 1970-х роках, появи та розвитку персональних комп'ютерів в 1980-х роках, інтернет-революції в 1990-х роках, мобільних і хмарних обчислень в 2000 р . і інфо-соціальних технологій в 2010 р.

У той же час окремо підкреслюється, що стартап - це не просто малий бізнес, а бізнес, націлений на майбутній розвиток. Стів Бланк і Боб Дорф визначають стартапи як «організації, сформовані для пошуку повторюваної і масштабованої бізнес-моделі» [57].

Формальними критеріями для визнання компанії стартапом зазвичай виступають вік компанії, число співробітників, прибуток і її зростання, наукомісткий характер продукту, контроль засновників над компанією і оцінка потенціалу компанії експертним журі [58]. Втім, дані критерії піддаються критиці. Так, наприклад, Пол Грем стверджує, що єдиним критерієм стартапу є швидке зростання [59].

В XXI столітті синонімом стартапує це компанії, які пов'язані з інформаційними технологіями, тобто програмним забезпеченням, обчислювальною технікою, проектуванням та виробництвом за допомогою комп'ютера, інтернетом, мобільними застосунками (додатками), а також різними телекомунікаційними сервісами та інформаційними послугами.

Останньою тенденцією країн, які здійснюють перехід від індустріального суспільства до інформаційного, стала організація взаємодії бізнесу і університетів через створення в університетському середовищі нових малих підприємств, що займаються комерціалізацією власних науково-технічних розробок і досліджень. Останнім часом, за такими підприємствами стала закріплюватися назва - академічних стартапів, а сам феномен залучення університетів в комерційну діяльність, стали називати академічним підприємництвом.

Слід зауважити, що співробітники таких новоутворених компаній при університетах комплектуються не лише за рахунок учених, але й студентів, що проходять підготовку в цих навчальних закладах. Тим самим відбувається трансформація у підготовці вже не просто майбутнього фахівця, а члена команди. Іншими словами, університет нового типу випускає вже не спеціаліста з набором певних компетенцій, а кластер фахівців, здатних до ефективного вирішення завдань відразу після університетської лави.

У свою чергу, бізнес-центри спеціалізуються на наданні послуг і сервісів, які можуть бути не пов’язані безпосередньо з діяльністю компаній, але більше пов'язані із забезпеченням функціонування їх (компаній) як ефективною бізнес-структурою. На усіх цих стадіях перед опікунською радою бізнес-центру стоїть задача з ухвалення рішень щодо підтримки утворення і формування таких компаній на основі представлених бізнес-ідей, переходу їх із станів стартап у спиноф, а також щодо припинення підтримки як з причини незадовільної діяльності, так і за фактом випуску компанії в самостійний бізнес. Такі рішення досить складні і вимагають пошуку як методологій, так і інструментальних засобів оцінювання якості роботи компаній і виконуваних ними проектів.

1.2 Екосистема стартапу

Розглянемо узагальнену модель оточення, в якій знаходиться бізнес, що зароджується, у вигляді стартап-компанії, іменованої також «екосистемою стартапу», запропоновану міжнародною організацією Startup Commons Global [10]. На думку цієї організації, для успішного формування та виживання стартапу його повинні оточувати ідеї, відкриття і результати сучасних досліджень, які неможливі без участі університетів та інших дослідницьких організацій.

Такий вплив близькості університетів та інших джерел знань до стартапів в області високих технологій підтверджується рядом наукових досліджень. Так, наприклад, в роботі вчених Р. Баптиста та Ж. Мендонка [11] на прикладі Португалії показано, що локальний доступ до знань і людського капіталу з високим рівнем освіти суттєво впливає на можливість появи нових компаній.

Але самі по собі ідеї та нові знання не зможуть вижити без підтримки відповідних організацій, що забезпечують їх підтримку і фінансування. Крім того, в своїй екосистемі стартапи самі є споживачами спеціалізованих послуг, як, наприклад, доступ до інтернету і високотехнологічного обладнання, юридичних і банківських послуг, сервісів різного роду і потребують організацій, здатних їх надати - сервіс-провайдерів. Нарешті, результати, отримані стартапом, повинні знаходити своє застосування в економіці та промисловості, представленої великим бізнесом. Дані положення графічно виражені на рис. 1.1.

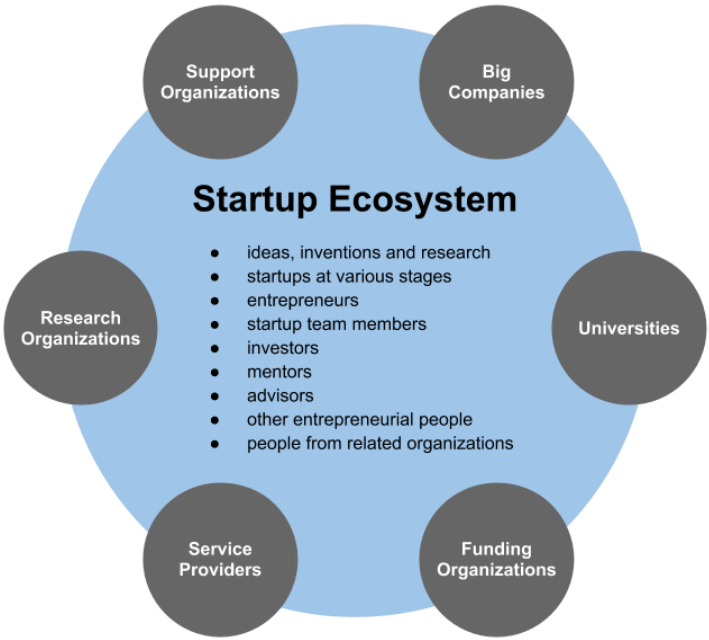


Рисунок 1.1 – Екосистема стартапів по [10].

На жаль, уявлення Startup Commons Global не можуть бути повністю перенесені на умови України, оскільки не включають в себе державу як дійову особу в даній схемі. Для економічно розвинених країн з тривалою історією ринкової економіки роль держави відходить на другий план, оскільки там історично існують умови для самостійності університетів, співпраці бізнесу і науки, розвитку приватного підприємництва. Іншими словами, там вже існують інститути розвитку наукоємного венчурного бізнесу, сформовані державою. Таким чином, у вищеназваних країнах роль держави розчиняється в загальному фоні екосистеми.

Не можна не відзначити, що питання впливу держави на стартапи розглядалися і зарубіжними дослідниками. Зокрема, в роботі Марка де Ревю, Гаррі Бовмана і Яна Макіннеса [12] виділялися три основні аспекти впливу на стартапи в ході їх розвитку: технологічний, ринковий і регуляторний (державний). Причому робилися висновки, що на різних етапах життєвого циклу компаній ці аспекти будуть діяти по-різному. Так, регуляторний фактор матиме вплив на етапі виведення продукту на ринок, оскільки може виявитися, що існують або можуть виникнути нормативні обмеження на використання продукту, як, наприклад, ліцензування радіодіапазону для бездротових пристроїв, наявність «подвійних технологій», створення нових ризиків і т.д. Втім, в тій же статті [12] було сказано, що для всіх досліджених компаній регуляторний фактор виявився незначним. Це лише свідчить про те, що в розвинених країнах держава встигає адаптувати своє законодавство під потреби ринку.

Для України і більшості країн колишнього радянського простору держава продовжує відігравати значну роль. Це типові проблеми, які існують і в інших державах, наприклад, проблеми, пов'язані з існуючими нормативними бар'єрами для розвитку нових технологій, як вищенаведений приклад з ліцензуванням радіочастот (в Україні з'явився 3G мобільний зв'язок тільки в 2015 р, фактично пізніше всіх країн колишнього радянського простору). Але, в першу чергу, це проблеми у відсутності привабливого інвестиційного клімату, умов для розвитку ринкової економіки, академічного підприємництва. Саме держава відповідальна за створення фундаментальних механізмів підтримки науки і освіти, в тому числі через податкові пільги та умови для повернення інвестицій для підприємств, які співпрацюють з університетами, як в НДДКР, так і в перепідготовці кадрів.

Відносно новим еко-середовищем формування стартапів є науково-технологічні кластери [13] - організаційно-територіальні об'єднання підприємств і організацій, що здійснюють діяльність у певній галузі економіки. Такі кластери, як правило, формуються в безпосередній близькості від профільних підприємств, але при цьому включають в себе наукові і освітні установи (наприклад, університети) [14]. Їх характеризує наявність сформованих зв'язків між організаціями, взаємовигідний трансфер технологій. В Україні найбільш успішними IT-кластерами можна вважати Львівський і Харківський кластери, далі йдуть Одеський і Черкаський.

1.3 Модель життєвого циклу IT-стартапу

Вибуховий розвиток інформаційних технологій змінює світ навколо, а разом з ним змінюються технологічні, ринкові та державні умови. На відміну від традиційного товарного виробництва цикл зміни поколінь IT-продуктів і послуг становить менше десятка років, а для деяких товарів може становити 1-2 роки. Це змушує компанії постійно пристосовуватися до умов часу, змінювати свої бізнес-моделі, організаційні структури і навіть ринкові ніші. Хто б міг уявити в XX ст., що компанія IBM відмовиться від бізнесу, пов'язаного з випуском персональних комп'ютерів, а Motorola і Nokia від виробництва мобільних телефонів? Це змушує дослідників вивчати питання, пов'язані з життєвим циклом підприємств і необхідністю зміни бізнес-моделей. Тут слід зазначити роботи [15, 16] з їх основною ідеєю, що компанії повинні переглядати і навіть заново винаходити свої бізнес-моделі. Як зазначалося в статті [12]], компанії в області інформаційних технологій переживають 3 стадії:

- Розробка (Development/R&D) - створення продукту/послуги, починаючи від концептуальної ідеї як самого продукту, так і його майбутнього місця на ринку.
- Впровадження (Implementation/Roll-out) - запуск продукту на ринку і отримання зворотного зв'язку від замовників, зокрема перші доходи.
- Комерціалізація (Commercialization) - перехід від експериментів з продуктом і ринком до випуску сталого товару і отримання прибутку.

Даний підхід можна поширити і на новоорганізовані малі підприємства, створювані в академічному середовищі.

Уявімо узагальнену модель життєвого циклу умовної компанії, яка отримує підтримку через бізнес-центр при університеті.

В роботі [17] було показано, що шлях компанії від колективу однодумців в самостійний бізнес супроводжується зростанням «зрілості компанії» в термінології моделі SW-CMM [18]. Тут під «зрілістю» мається на увазі рівень формалізму процесів і застосовуваного практичного знання (практик) до формально певних кроків і керованих результуючих метрик з метою досягнення оптимізації процесів в компанії.В загальній формі це можна представити у вигляді рис. 1.2.

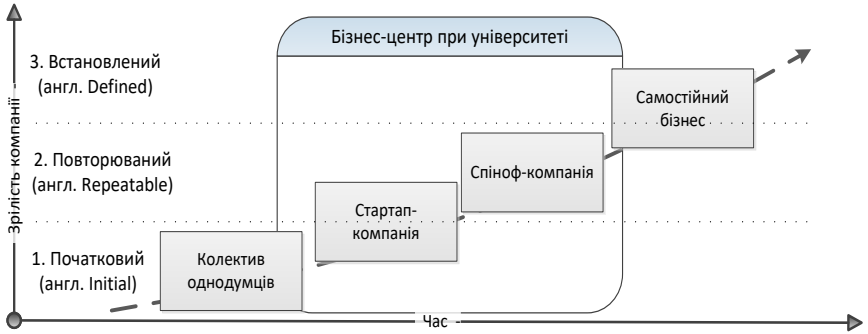


Рисунок 1.2 – Життєвий цикл новостворюваної компанії при університеті

Отже, новостворювана компанія переживає такі стадії розвитку.

1. Колектив однодумців. На даній стадії група людей приходить до усвідомлення комерційної цінності виконаних досліджень або отриманих технологій і об'єднується для ведення бізнесу. Ця фаза завершується розробкою бізнес-планів і подачею заявки на підтримку з боку бізнес-центру при університеті. На даному етапі не можна говорити про зрілість, оскільки фактично ще не відбулося формування організації.

2. Підтримувана стартап-компанія. На даній стадії колектив розвиває наявне «know-how» в пілотну версію продукту (послуги), а також опановує компетенції, пов'язані з веденням бізнесу. Як правило, бізнес-процеси такої компанії будуть відповідати першому «початковому» рівню зрілості по моделі SW-CMM.

3. Розкручувана спіноф-компанія. На цьому етапі компанія здійснює свій вихід на ринок. Однак бізнес-процеси такої компанії лише

тільки починають набувати форми і документуватися. До кінця цієї стадії компанія повинна вийти на другий «повторюваний» рівень зрілості.

4. Самостійний бізнес без участі бізнес-центру. На початку самостійного ведення справ компанія залишає бізнес-центр. Очікується, що на даній стадії компанія набуває третій «певний» рівень зрілості, при якому бізнес-процеси будуть належним чином документуватися і перестануть бути залежними від окремих особистостей.

При цьому місце бізнес-центру при університеті в рамках представленого життєвого циклу покриває стани «стартап» і «спіноф» майбутньої компанії.

Представлена модель життєвого циклу дозволяє запропонувати рішення задачі якісної оцінки стану вищеназваної компанії шляхом розрахунку її «зрілості» в термінології моделі SW-CMM. Причому такий розрахунок може бути проведений за допомогою інструменту - калькулятора зрілості IT-компанії, описаного в підрозділі 2.5.

1.4 Модель управління IT-стартапом

В роботі Х.Бовмана, Т.Хаакера і Х. Де Воза «Інновації в мобільних послугах та бізнес-моделі» [19] виділяють 4 основних компоненти бізнес-моделі IT-компанії.

- Продукт (в оригінальній роботі сервіс) - опис ціннісної пропозиції і сегмент ринку, на які ця пропозиція направлена.
- Технологія - опис технологічної функціональності, необхідної для реалізації майбутнього продукту або сервісу.
- Організація - опис структури мережі діючих суб'єктів, необхідної для створення і поширення пропонованого продукту/послуги.
- Фінанси - опис механізму покриття ризиків, повернення інвестиції та отримання прибутку від запропонованого продукту.

Всі ці компоненти тісно пов'язані між собою і взаємно впливають один на одного (див. Рис. 1.3).

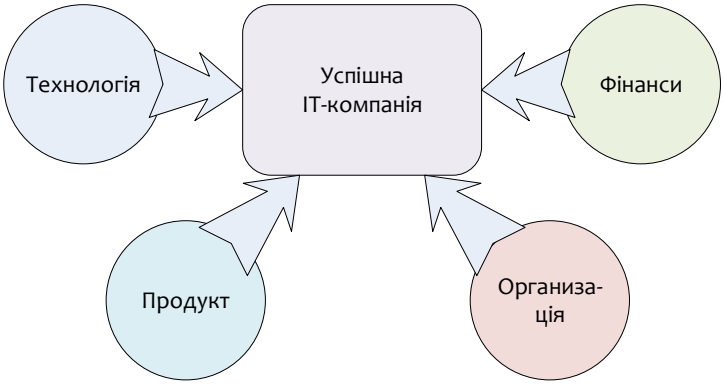


Рисунок 1.3 – Модель IT-компанії по Бовману та ін.

У моделі організаційної зрілості Інституту програмної інженерії університету Карнегі-Меллон CMMI-DEV [20], що прийшла на зміну моделі SW-CMM, виділяють сім процесних областей для організацій, які досягли 2-го рівня зрілості - рівня, досягнення якого можна очікувати від IT-стартапу. Це дві області, пов'язані з плануванням і управлінням проектом, три області, пов'язані з управлінням вимогами, конфігураціями і забезпеченням якості продукту, одна область, пов'язана з вимірюванням та аналізом показників організації, і одна область, пов'язана з управлінням постачальниками.

Техаський інститут якості програмного забезпечення визначає три категорії, якими необхідно управляти з метою успішної реалізації IT-проекту (створення програмного забезпечення). Це продукт, проект і персонал. Причому кожна категорія вимагає свого набору компетенцій [21], список яких наведено нижче.

Компетенції, пов'язані з розробкою продукту.

1.Процеси оцінювання - визначення критеріїв для експертних оцінок продукту, що розробляється.

2. Знання стандартів - розуміння міжнародних і галузевих стандартів.

3. Визначення продукту - ідентифікація клієнтського середовища і вимог до продукту.

4. Оцінка альтернатив - як оцінка різних застосовуваних підходів.

5. Управління вимогами - контроль за змінами вимог до продукту.

6. Управління субпідрядниками.

7. Виконання початкової оцінки - оцінки ризиків, витрат, графіка і складності продукту.

8. Відбір методів та інструментів.

9. Підгонка процесів - зміна стандартних процесів (набору дій по перетворенню вхідних даних в результат) з урахуванням досягнення цілей проекту.

10. Відстеження якості продукту.

11. Розуміння дій по розробці продукту - життєвого циклу розробки.

Компетенції, пов'язані з управлінням проектом.

12. Створення структури пооперaційного переліку робіт (WBS для проекту в термінах Керівництва з управління проектами PMI PMBoK [22]).

13. Документування планів.

14. Оцінка витрат - оцінка витрат, необхідних для завершення проекту.

15. Оцінка трудовитрат - оцінка трудовитрат, необхідних для завершення проекту.

16. Управління ризиками - визначення ступеня впливу і усунення впливу ризиків.

17. Відстеження процесу розробки.

18. Складання графіка робіт.

19. Відбір метричних показників.

20. Відбір інструментів управління проектами.

21. Відстеження процесів діяльності команди проекту.

22. Відстеження ходу виконання проекту - контроль проекту за допомогою метричних показників.

Компетенції, пов'язані з управлінням персоналом.

23. Оцінка продуктивності - оцінка діяльності команди, спрямована на поліпшення її роботи.

24.Питання інтелектуальної власності - розуміння ролі інтелектуальної власності в проекті.

25. Проведення ефективних зустрічей - планування і проведення ефективних нарад.

26. Взаємодія та спілкування - робота з власною командою, вищим керівництвом і іншими командами.

27. Лідерство.

28. Управління змінами.

29. Ведення переговорів.

30. Планування кар'єрного росту.

31. Ефективне представлення - використання письмових і усних навичок.

32. Набір персоналу - успішне залучення і співбесіда з членами команди.

33. Відбір команди - відбір висококомпетентних команд.

34. Створення команди - формування та підтримка ефективної команди.

Виконавши синтез різних моделей і відкинувши чинники, які властиві великим компаніям і не будуть діяти у випадку IT-стартапу, отримаємо модель, представлену на рис. 1.4, на якій пунктиром показаний контур управління IT-стартапом. У даній моделі виділяються п'ять компонентів, дія на які має прямий вплив на цільовий стан стартапу: команда, технології, організаційні процеси, кінцевий продукт і проектна діяльність. Остання є місцем інтеграції для всіх попередніх компонентів.

Тут важливо відзначити, що для університетського бізнес-центру головним цільовим станом стартапу є досягнення здатності до ефективного ведення самостійного бізнесу, що визначається організаційної зрілістю компанії, а не успіхами ідеї в пресі або продукту на ринку.

У функції бізнес-центру не входить розробка продукту або безпосереднє управління командою. Бізнес-центру важливо мати основну проектну інформацію стартапу, таку як поточне планування і статус робіт, розподіл ресурсів за завданнями, а також рівень впровадження процесів забезпечення якості. Необхідно знати про відсутні у стартапа ресурси, технології та компетенції, недостатність яких бізнес-центр може компенсувати. Йому важливо відстежувати динаміку змін в компетенціях стартапа в ході його розвитку. Тоді він може надати керуючий вплив на стартап з метою розвитку необхідних компетенцій.

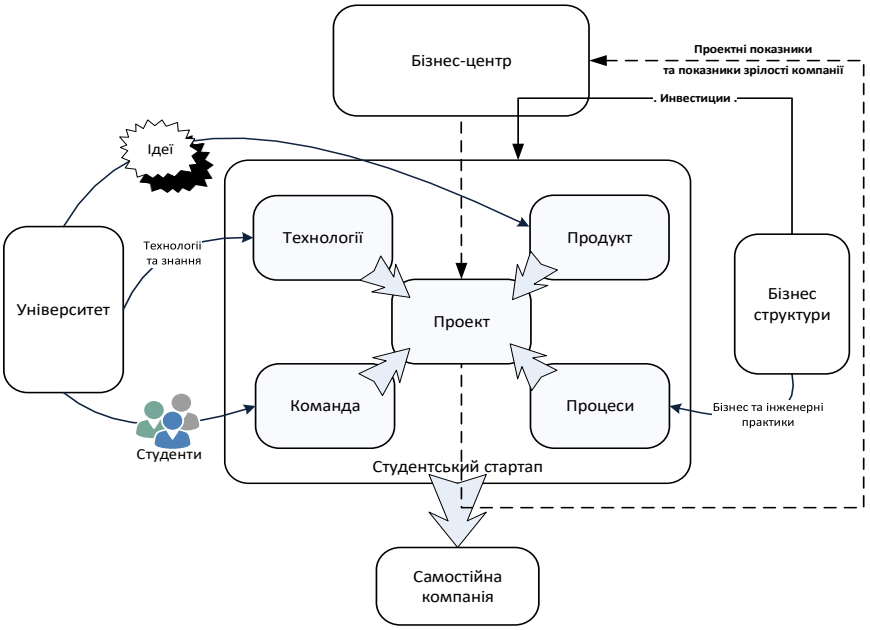


Рисунок 1.4 – Концептуальна модель університетського IT-стартапу

Тобто керуючий вплив на IT-стартап буде реалізовано шляхом видачі рекомендацій щодо включення у графік проекту робіт, пов'язаних з поліпшенням його організаційної зрілості, виробничих процесів, а також необхідних компетенцій як на індивідуальному рівні окремих ролей в команді, так і на рівні компанії в цілому . Такий вплив буде підкріплено умовами доступу до ресурсів бізнес-центру (фінансування і т.д.). Таким чином саме для реалізації керуючого впливу університетського бізнес-центру необхідний ряд інструментів оцінювання стартапу і його проекту.

Головною відмінністю академічного IT-стартапу від самостійної компанії є відсутність у нього ряду ресурсів і функціональних компетенцій, недолік яких на першому етапі повинен компенсувати університетський бізнес-центр і далі забезпечити їх формування всередині IT-стартапу. Як правило для IT-команд, сформованих навколо ідеї або нової технології, такими відсутніми компетенціями є юридична, обліково-фінансова, маркетингова, управлінська діяльність і забезпечення якості. На рис. 1.5

приведена структурна схема функціональних компетенцій, які повинні бути освоєні академічним IT-стартапом.

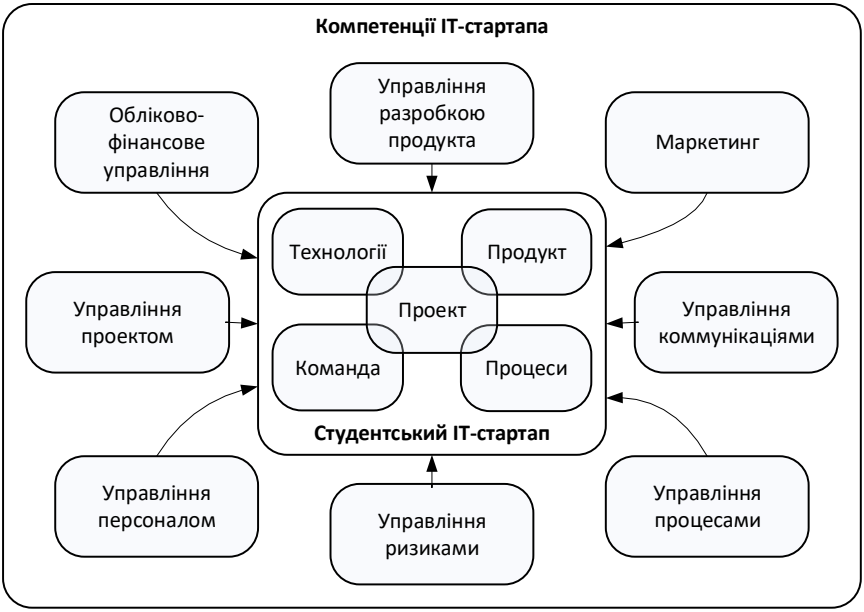


Рисунок 1.5 – Структурна модель компетенцій IT-стартапу

Перерахуємо такі компетенції: управління розробкою продукту, включаючи технологічні компетенції для цього; управління проектом; управління персоналом; управління технологічними процесами в компанії; маркетинг; управління комунікаціями як всередині компанії, так і з зовнішнім середовищем; управління ризиками; управління фінансами, включаючи фінансове планування, бухгалтерський облік тощо.

РОЗДІЛ 2. ІНСТРУМЕНТАЛЬНІ ЗАСОБІ УНІВЕРСИТЕТСЬКОГО БІЗНЕС-ЦЕНТРУ

2.1 Структура інструментальних засобів підтримки академічного підприємництва

Для реалізації керуючого впливу університетському бізнес-центру потрібен набірінструментів оцінювання стартапу і його проекту, структурна схема якогопредставлена на рис. 1.6. Такі інструментальні засоби зручно розділити на групи.

В першу групу можна віднести інструментальні засоби, пов'язані з типовими функціями бізнес-центрів (інкубаторів). Такими типовими функціями є надання технічних і фінансових ресурсів, юридичних, обліково-фінансових і маркетингових послуг. Сюди слід віднести базу даних типової документації малих компаній і базу знань типових процесів, пов'язаних із забезпеченням функціонування самостійного бізнесу. Наприклад, забезпечення якості, організаційні процедури і регламенти, ведення документообігу та управління персоналом. Такі інструменти не є специфічними для компаній, утворених в академічному середовищі, і їх розгляд виходить за рамки даної роботи.

До другої групи слід віднести інструментальні засоби, пов'язані безпосередньо з функціями університету. До них слід віднести банки знань (ноу-хау), ідей і нових технологій, які народжуються в стінах університету. Сюди слід включити автоматизовані навчальні системи [23], а також інструменти формування і оцінки компетенцій, описані в роботах [17, 24, 25].

У третю групу віднесені інструментальні засоби, що мають специфіку, пов'язану з інформаційними технологіями. У цю групу входять інструменти управління IT-проектом; розробки продукту; управління процесами в IT-компанії; оцінки ризиків; оцінки зрілості IT-компанії; оцінки термінів досягнення цілей IT-проекту.

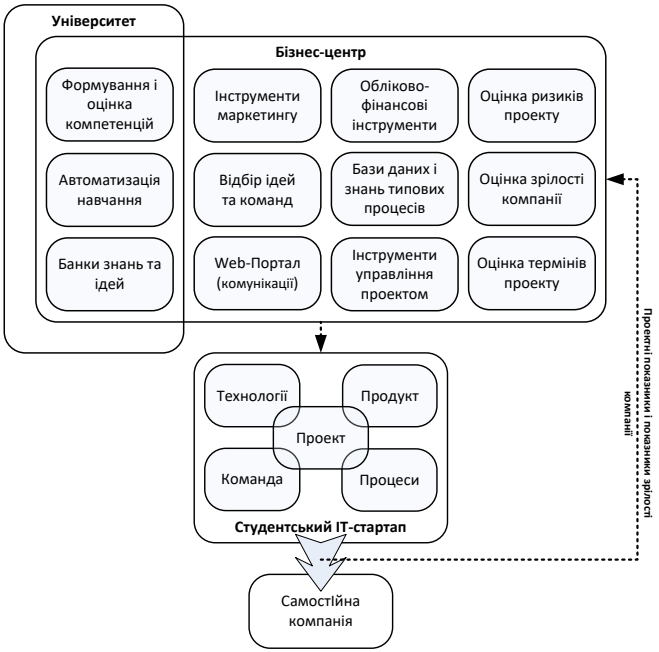


Рисунок 2.1 – Структура інструментальних засобів університетського бізнес-центру

2.2 Веб-портал підтримки академічного IT-підприємництва

На даний час в світі ідеї використання мережі інтернет, для організації взаємодії між університетами та промисловістю мають високу популярність. Практично всі провідні ВНЗ світу мають окремі сайти і веб-сервіси, де відбувається інформування про умови та напрямки взаємодії ВНЗ з бізнесом. Найчастіше там дається інформація про виконувані університетами проекти, їх успішність і поточні показники.

В Україні ідею забезпечення інтеграції університетів з промисловістю через розширений мережевий веб-портал стартап і спіноф-компаній (Advanced Start-up Networking Web portal) запропонував колектив харківських вчених [26]. Далі ця наукова праця була вдосконалена і знайшла відображення в монографії під редакцією проф. В.В. Литвинова [17]. Тут, зокрема, зазначалося, що такий портал, натхненний ідеєю синергетичної співпраці університетів і промисловості, призначений для досягнення двох важливих цілей:

- організації можливості обговорення та обміну ідеями можливих спільних проектів між університетами та промисловістю;
- забезпечення робочого простору для співпраці.

У роботах [-17, 26] виділені основні вимоги до подібного порталу:

- Реєстрація проектних ідей з різним рівнем доступу інших користувачів до їх змісту і опису.
- Перегляд проектних пропозицій і здатність до організації їх обговорень як в публічному, так і закритому просторі.
- Можливість створення проектних команд у відповідному робочому просторі на сайті.
- Робочі простори для роботи над проектами повинні забезпечувати можливість обміну документацією та іншими артефактами проекту.
- Здатність користувача позиціонувати себе і управляти власним профілем на сайті.
- Інтеграція з соціальними мережами і подібними інтернет-ресурсами.

У тій же роботі [26] була запропонована концептуальна модель ASN веб-порталу, представлена на рис. 2.2

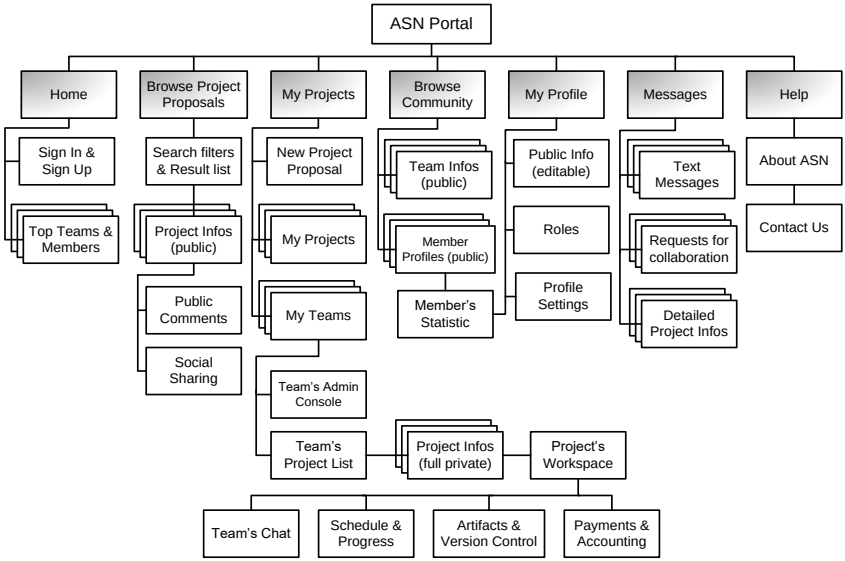


Рисунок 2.2 – Концептуальна модель ASN веб-порталу [26].

Незважаючина опрацювання дана концепція не включала в себе ідею використання веб-порталу в якості носія інструментальних засобів управління ІТ-стартапом з боку університетського бізнес-центру, як це було зазначено вище. Проте, дана концепція може бути базовою моделлю при побудові подібних порталів в університетах. Так наприклад, подібна робота була виконана в рамках міжвузівської кооперації між Славутицькою філією НТУУ «КПІ» та кафедрою програмної інженерії ЧНТУ [27, 28, 29].

2.2.1 Постановка задачі на проектування веб-порталу

Групі студентів ЧНТУ і СФ НТУУ «КПІ» була сформульована і поставлена задача побудови прототипу веб-порталу, як програмно-технічного інструменту підтримки академічного ІТ-підприємництва на базі Славутицької філії НТУУ «КПІ».

Безпосередньо веб-портал повинен бути спроектований на основі відкритих веб-технологій та програмного забезпечення з відкритим кодом. Реалізований портал повинен підтримувати принципи відкритості і розширюваності. Для цієї мети проектований портал повинен мати модульну архітектуру і достатній ступінь документованості, а також реалізовувати програмні інтерфейси для подальшої інтеграції зі сторонніми модулями підтримки академічного ІТ-підприємництва, які створюються і кодуються іншими незалежними групами.

Робота над порталом повинна включати завдання вибору життєвого циклу даного проекту, що виконується студентами самостійно на основі методології, описаної в розділі 2.4 цієї роботи. Робота над проектом з самого початку не передбачає знайомства студентів з існуючою концептуальною моделлю ASN веб-порталу. На етапі перевірки прототипу, дана концептуальна модель буде виступати в якості зразкової моделі для аналізу повноти спроектованих функцій системи.

2.2.2 Вибір життєвого циклу створення

В результаті аналізу життєвих циклів створення ІТ-продукту і на основі існуючих обмежень студентом самостійно був здійснений вибір моделі швидкого прототипування, графічне представлення якої наведено на рис. 2.3.

Рисунок 2.3 – Модель швидкого прототипування

Першим етапом роботи, за даною моделлю життєвого циклу програмного забезпечення, вважається створення плану проекту на основі початкових вимог. Потім виконується швидкий аналіз, проектується база даних, інтерфейс користувача, а також функції системи. Далі переходять до побудови прототипу системи, який представляється замовнику, після чого визначаються зауваження і побажання, уточнюються вимоги і повторюється цикл побудови вже кращого прототипу. Процес закінчується тоді, коли замовник офіційно затверджує робочу модель системи. Після чого допрацьовується проектно-технічна документація, по факту виконаної роботи.

Модель прототипування користується великою популярністю серед студентів, оскільки вона, з їх точки зору, дозволяє швидко приступити до реалізації системи і не обтяжує «паперовою роботою». Хоча саме цю модель заслужено критикують в проектах, де критерій «якості» домінує над «графіком» і «витратами».

2.2.3 Формування вимог

Якісне визначення вимог має велике значення для успішного завершення проекту. Зазвичай, для визначення вимог до проекту аналітики використовують декілька методів, такі як інтерв'ю, створення списків вимог та використання фокус-груп. Також, існують сучасні методи для визначення

вимог, такі як прототипування та створення сценаріїв використання. Класичні методології збору вимог вимагають високої компетенції розробників для перетворення висунутих вимог у специфікацію функціоналу системи.

2.2.3.1 Користувацькі історії

Беручи до уваги, що на початковому етапі у безпосередніх виконавців відсутнє уявлення про кінцевий продукт, для формування вимог до системи було запропоновано використовувати методику, відому в методології «екстремального програмування» (англ. Extreme programming (XP)) [30, 31] як «користувацькі історії»(англ. user story) [32]. Відповідно до даної методики, типовим шаблоном для документування вимог до системи є шаблон роль-дія-цінність (англ. Role-feature-benefit або останнім часом role-feature-reason). У загальному вигляді даний шаблон можна представити у вигляді пропозиції: «Як <роль>, я хочу <дія>, щоб отримати <цінність>».

Шляхом опитувань студентів, були виявлені три основні ролі: студент, підприємець і співробітник університетського бізнес-центру, що в цілому відповідає моделі, запропонованій в роботі [26]. Так само, на основі опитувань студентів, для кожної з ролей були описані ряд користувацьких історій, деякі з яких представлені нижче.

- Як студент, я хочу зареєструвати свій проект, щоб отримати підтримку бізнес-центру;
- Як студент, я хочу отримати підтримку бізнес-центру, щоб реалізувати свій проект, підвищити свою професійну компетенцію, створити власну IT-компанію;
- Як студент, я хочу реалізувати свій проект, щоб привернути увагу інвесторів, отримати практичний досвід, отримати готовий проект для резюме;
- Як студент, я хочу створити команду, щоб реалізувати свій проект;
- Як студент, я хочу отримати відгуки від людей про мій проект, щоб покращити його;
- Як студент, я хочу мати можливість проінформувати людей про свої здібності та навички, щоб привернути увагу роботодавців і інвесторів;

- Як студент, я хочу спілкуватися з іншими студентами і командами, щоб перейняти досвід, створити контакти та вступити в іншу команду;
- Як студент, я хочу отримати інвестиції, щоб розвивати проект, створити власну IT- компанію;
- Як студент, я хочу придбати практичний досвід створення програмного забезпечення, щоб відповідати вимогам сучасного ринку;
- Як працівник бізнес-центру, я хочу допомогти студентам зорганізуватися в команди, щоб студенти мали змогу реалізувати свої проекти і вміли працювати в команді;
- Як працівник бізнес-центру, я хочу допомагати студентам реалізувати свої проекти, для того щоб студенти мали шанс створити власну IT-компанію, набути досвід в сфері створення програмних продуктів;
- Як працівник бізнес-центру, я хочу щоб студенти створили власну IT-компанію, щоб отримувати фінансування від новостворених IT-компаній;
- Як підприємець, я хочу переглянути всі проекти, щоб знайти найперспективніший і оцінити доцільність фінансування проекту;
- Як підприємець, я хочу вкласти гроші в проект, щоб отримати долю прибутку від проекту або підтримати цікавий проект;
- Як підприємець, я хочу знайти перспективного кваліфікованого працівника, щоб закрити вакансію на своєму підприємстві;
- Як підприємець, я хочу викупити проект, щоб самостійно його розвивати та отримати прибуток.

Дані відповіді, при всій своїй наївності, дозволяють зробити висновок не тільки про безпосередню функціональність майбутнього веб-порталу, але і про очікування сучасних студентів від бізнес-центрів при університеті.

2.2.3.2 Моделювання функцій системи

Метою створення веб-порталу університетського бізнес-центру є автоматизація процесів бізнес-центру. Одним із основних завдань веб-порталу бізнес-центру вважається можливість організовувати студентів в команди, з чітким поділом ролей. Це надає змогу розвивати здатність студентів працювати в команді, нести відповідальність за певний об’єм робіт. Поділ ролей надає змогу розвиватись студентам в напрямках цікавих

самим студентам. Завдяки організації студентів в команди, підприємець може оцінити ймовірність реалізовуваності проекту або знайти студента, який компетентний в певній області для запрошення його на роботу. Тому даний веб-портал повинен мати змогу представляти склад команд, з ролями кожного учасника команди. Кожна команда повинна мати лідера, який буде створювати команду, запрошувати в команду учасників, та редагувати склад команди за потреби.

Не менш важливим завданням веб-порталу бізнес-центру вважається можливість зареєструвати проект, або ідею для проекту. Веб-портал повинен представляти вичерпну інформацію про проект. Дана інформація надає підприємцям змогу оцінити компетенцію команди, а також звичайним користувачам оцінити та обговорити проект з іншими людьми. Кожний проект повинен мати дати початку і прогнозну дату завершення, тип проекту, стадію на якому знаходиться даний проект, додаткову інформацію та команду, яка реалізує даний проект.

Також, всі користувачі повинні мати можливість спілкуватися між собою. Це надає змогу працівникам бізнес-центру координувати діяльність студентських команд, інвесторам встановити зв’язок з бізнес-центром для взаємодії, або встановити зв’язок зі студентом та самим студентам спілкуватися між собою, отримувати вказівки від бізнес-центру та повідомлення від потенційних роботодавців.

Для побудови моделі системи використовується методологія IDEF0 [33] (Integrated DEFinition), яка представляє собою сукупність правил, процедур, методів, які призначені для побудови моделі системи. Функціональна модель IDEF0 відображає функціональну структуру системи, тобто дії які виконує система і зв’язки між цими діями.

Рисунок 2.4 – Діаграма верхнього рівня IDEF0

На вхід надходять персональна інформація, проекти. Нові користувачі, які реєструються в системі, вводять персональні дані, прикріпляють фотографію до облікового запису та описують власні компетенції. Дана інформація слугує для оцінки компетенції користувача та для інформування інших користувачів. Також, на вхід надходять проекти, тобто ідеї для проектів, або вже існуючи проекти, але які не реалізовані до кінця. Це надає змогу поділитися своєю ідею з іншими, або заручитися підтримкою бізнес-центру та інших користувачів. Групи однодумців можуть, завдяки бізнес-центру, реалізувати власний проект, або розвиватись, щоб утворити власну IT-компанію.

До ресурсів системи слід віднести фінансування, техніку, та ресурси бізнес-центру. Фінансування відбувається від підприємців, які зацікавленні в реалізації проекту, або в підтримці конкретної команди. Також, деякі проекти потребують технічної підтримки, тобто команди потребують певної техніки для реалізації проекту. До ресурсів бізнес-центру слід віднести як матеріальну підтримку, так і підтримку шляхом усилення команд компетентними людськими ресурсами та викладачами, які будуть підвищувати компетенцію команди в певній області.

Серед обмежень, слід вважати доцільність підтримки команд і проектів. Кожний проект оцінює експертна група на доцільність підтримки та оцінює на рівень компетентності команди.

На виході із системи отримуємо реалізовані проекти, новостворені IT-компанії, оцінені проекти, кваліфікованих спеціалістів. Завдяки

підтримці веб-порталу, проекти мають можливість бути реалізовані. Завдяки інструментам веб-порталу проекти мають можливість отримати експертну оцінку.

У результаті аналізу вимог виділяються такі модулі системи:

1. «Команда»;
2. «Проект»;
3. «Користувач».

Діаграма нижнього рівня набуде вигляду(Рис. 2.5):

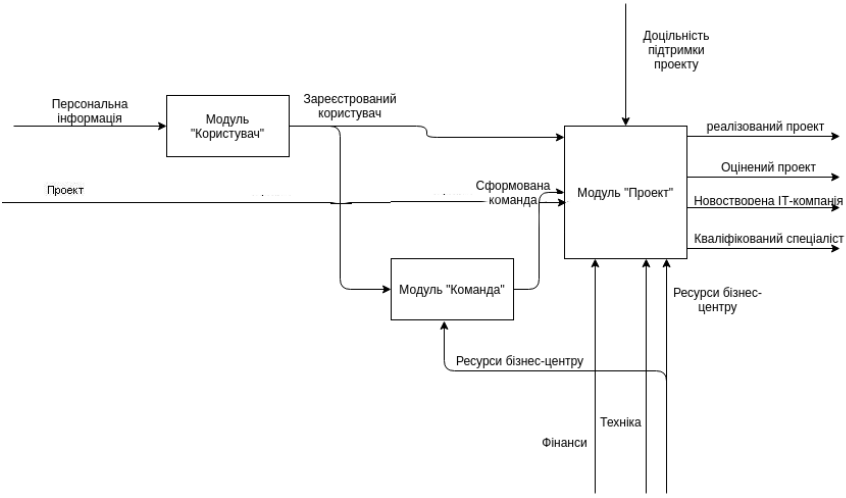


Рисунок 2.5 – Діаграма нижнього рівня IDEF0

Діаграма нижнього рівня складається з трьох модулів: «Користувач», «Команда» і «Проект». На вхід до модулю «Користувач» надходить персональна інформація, з метою реєстрації нових користувачів. На виході з модулю «Користувач» отримуємо зареєстрованого користувача, який може вступити або створити команду та зареєструвати власний проект або ідею для проекту. На вхід до модулю «Команда» надходять зареєстровані користувачі. На виході з модулю «Команда» отримуємо сформовану команду, яка може складатися з зареєстрованих користувачів і працівників бізнес-центру. До модулю «Проект» надходять сформовані команди і існуючі проекти. Модуль «Проект» має такі ресурси як: ресурси бізнес-центру, фінанси та техніка. Серед обмежень модулю «Проект» варто зазначити доцільність підтримки, оскільки якщо проект буде не доцільно підтримувати, тоді проект не буде реалізовуватись. Всі особи, які

займались реалізацією проекту підвищують свої компетенції, вони можуть утворити новостворену IT-компанію, реалізувати проект, або отримати експертну оцінку існуючого проекту.

Побудуємо діаграми для кожного з модулі системи. Діаграма для модулю «Користуач» набуде вигляду (Рис. 2.6):

Рисунок 2.6 – Діаграма модулю «Користувач»

Діаграма модулю «Команда» набуде вигляду (Рис. 2.7):

Рисунок 2.7 – Діаграма модулю «Команда»

Діаграмма модулю «Проект» набуде вигляду(Рис. 2.8):

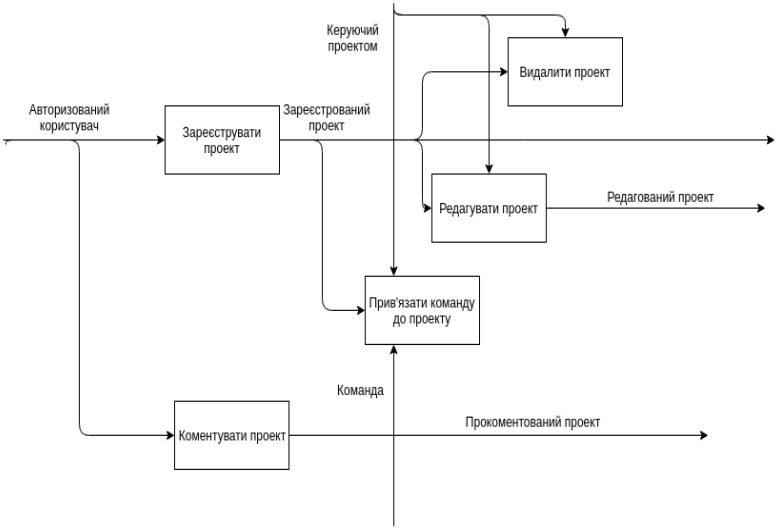


Рисунок 2.8 – Діаграма модулю «Проект»

2.2.3.3 Діаграми прецедентів UML

Найбільш важливий ефект від методології опису вимог виражається в тому, що подібна робота полегшує перехід до моделювання майбутньої системи за допомогою графічних нотацій UML [34], а саме створення діаграм прецедентів (англ. Use case diagram).

Діаграми прецедентів UML забезпечують високорівневий опис того, що система повинна робити і з ким, або чим, вона буде взаємодіяти. Діаграми прецедентів вважаються одним із методів визначення функціональних вимог, або візуалізацію цих вимог. Існує декілька правил для побудови діаграм прецедентів:

- Актором може виступати окрема особа, компанія, пристрій, програмне забезпечення, або будь-який інший зовнішній об’єкт, який взаємодіє з системою.

• Варіант використання представляє собою дію, яка виконується в рамках системи.

Враховуючи завдання до веб-порталу університетського бізнес-центру та користувацькі історії, можна поділити веб-портал на такі модулі, як «Користувач», «Проект» і «Команда».

Діаграма прецедентів модуля «Команда» має вигляд (Рис. 2.9):

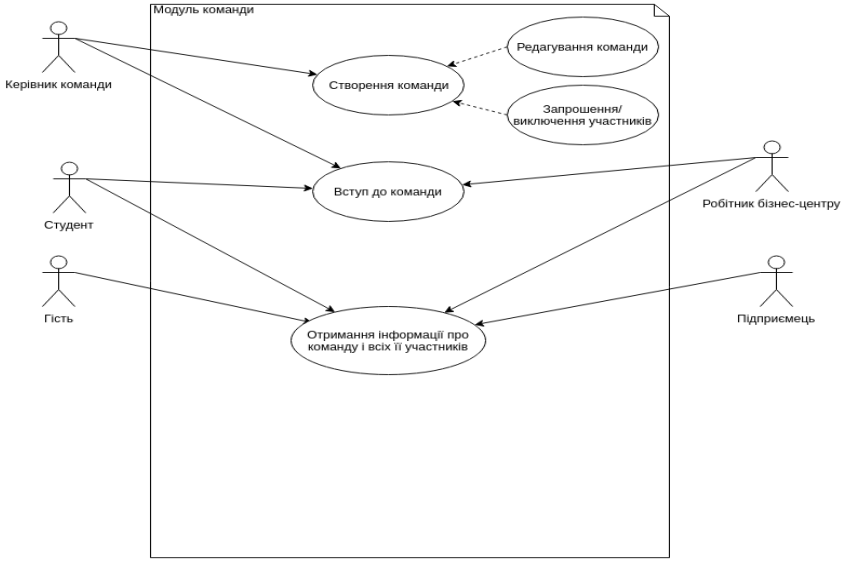


Рисунок 2.9 – Діаграма прецедентів для модуля «Команда»

Актор	**Прецедент (дія)**
Керівник команди	створює команду; редагує команду; запрошує та виключає учасників команди
Студент	вступає в команду; отримує вичерпну інформацію про команду та її учасників.
Робітник бізнес-центру	вступає в команду; отримує інформацію про команду та її учасників.
Підприємець	отримує інформацію про команду та її учасників.

Діаграма прецедентів модуля «Проект»має вигляд(Рис. 2.10):

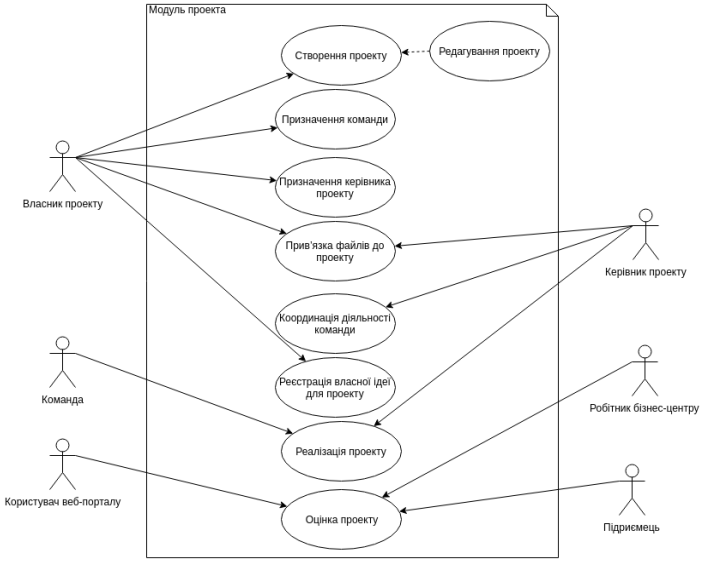


Рисунок 2.10 – Діаграма прецедентів модуля «Проект»

Актор	**Прецедент (дія)**
Власник проекту	створює проект; реєструє ідею для проекту; редагує проект; призначає команду, яка буде реалізувати проект; призначає керівника проекту; прив'язує файли до проекту.
Команда	реалізує проект
Користувач веб-порталу	дає оцінку проекту
Керівник проекту	прив'язує файли до проекту; координує діяльність команди; реалізує проект
Підприємець	оцінює проект
Робітник бізнес-центру	оцінює проект

Діаграма прецедентів модуля «Користувач» має вигляд (Рис. 2.11):

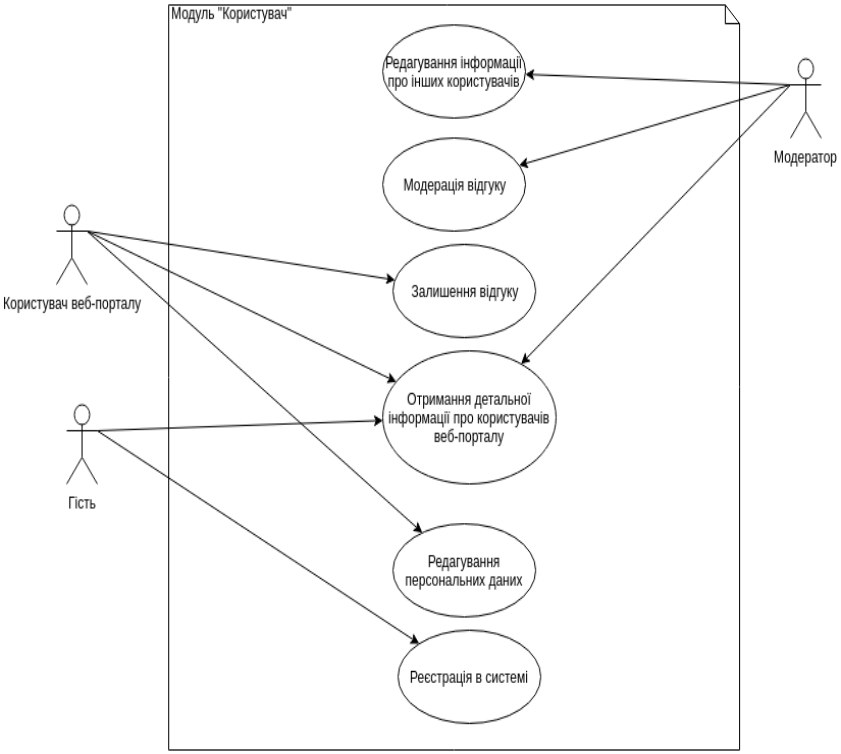


Рисунок 2.11 – Діаграма прецедентів модуля «Користувач»

Актор	**Прецедент (дія)**
Користувач веб-порталу	залишає відгук; отримує детальну інформацію про користувачів веб-порталу; редагує персональні дані
Гість	отримує детальну інформацію про користувачів веб-порталу; реєструється в системі
Модератор	редагує інформацію про інших користувачів; модерує відгуки; отримує детальну інформацію про користувачів веб-порталу

2.2.3.4 Функції модулів веб-порталу

Виконане моделювання використання системи у вигляді нотації UML дозволяє визначити наступний набір функцій для модулів системі. Для модулю «Користувач» таблиця функцій набуде вигляду:

Таблиця 2.1 Функції модулю «Користувач» для веб-порталу

Модуль	**Функція**	**Адміністратор**	**Власник облікового запису**	**Звичайний користувач**
Користувач	Ведення бази даних користувачів			
	Реєстрація користувачів	-	-	+
	Редагування облікового запису	+	+	-
	Видалення облікового запису користувача	+	+	-
	Поділ користувачів на групи	+	+	+
	Обмеження повноважень різних груп користувачів	+	+	+
	Пошук користувача по імені або прізвищу	+	+	+
	Шифрування персональних даних			
	Модерація персональних даних	+	+	-
	Додавання відгуку про бізнес-центр	-	+	
	Модерація відгуку про бізнес-центр	+	-	-

Для модулю «Команда» таблиця функцій матиме вигляд:

Таблиця 2.2 Функції модулю «Команда» для веб-порталу

Модуль	**Функція**	**Адміністратор**	**Власник команди**	**Звичайний користувач**
Команда	Ведення бази даних команд			
	Додавання команд	-	-	+
	Редагування інформації про команду	+	+	-
	Видалення команди	+	+	-
	Додавання нових учасників до команди	+	+	-
	Призначення ролей новим учасникам команди	+	+	-
	Виключення з команди учасників	+	+	-
	Пошук команди, серед всіх команд, за назвою	+	+	+
	Оцінка компетенції команди	+	-	+
	Модерація інформації про команди	+	-	-

Для модулю «Проект» таблиця функцій матиме вигляд:

Таблиця 2.3 Функції модулю «Проект» для веб-порталу

Модуль	**Функції**	**Адміністратор**	**Власник проекту**	**Звичайний користувач**
Проект	Ведення бази даних проектів	+	+	-
	Додавання проекту	-	-	+
	Редагування інформації про проект	+	+	-
	Видалення проекту	+	+	-
	Призначення відповідальної особи за реалізацію проекту	+	+	-
	Призначення проектної команди	+	+	-
	Прикріплення файлів до проекти	+	+	-
	Здатність залишати коментарії проекту	+	+	+
	Можливість оцінити проект	+	-	+
	Оцінка проекту експертною командою	+	-	-

	Пошук проектів за типом, статусом проекту	+	+	+
	Прив’язка технологій і платформ до проекту	+	+	-
	Сортування проектів за різними параметрами	+	+	+
	Пошук проекту за назвою	+	+	+

2.2.4 Проектування бази даних

Особливістю моделі прототипування є раннє проектування бази даних, що передує проектуванню призначеного для користувача інтерфейсу і переліку функцій системи. Для проектів з нечіткими вимогами і короткими термінами виконання такий підхід виправданий з позиції того, що проектування сутностей майбутньої бази даних, по суті тотожне об'єктному аналізу майбутньої системи і дозволяє на ранніх стадіях виділити такі об'єкти і описати їх атрибути.

На рис. 2.12 представлена ER-діаграма сутність-зв'язок для базової функціональності веб-порталу, пов'язаної з управлінням інформацією про проекти, які підтримуються бізнес-центром.

Як видно з діаграми, центральними сутностями системи є проект, користувач порталу та команда, що здійснює реалізацію проекту. Сутність члена команди вбирає в себе інформацію про роль, яку відіграє в команді її член і має пряму асоціацію з користувачем веб-порталу. Інші представлені на діаграмі сутності носять другорядний характер і призначені для організації повноти опису зареєстрованого проекту, а так само для створення користувацьких оцінок і коментарів. В цілому, дана модель покриває функціональність ASN веб-порталу [26]. Крім цього, в моделі представлені рішення, що підтримують суб'єктивні оцінки проектів з боку користувачів веб-порталу. Детальна інформация стосовно сутностей системи наведена в табл. 2.4.

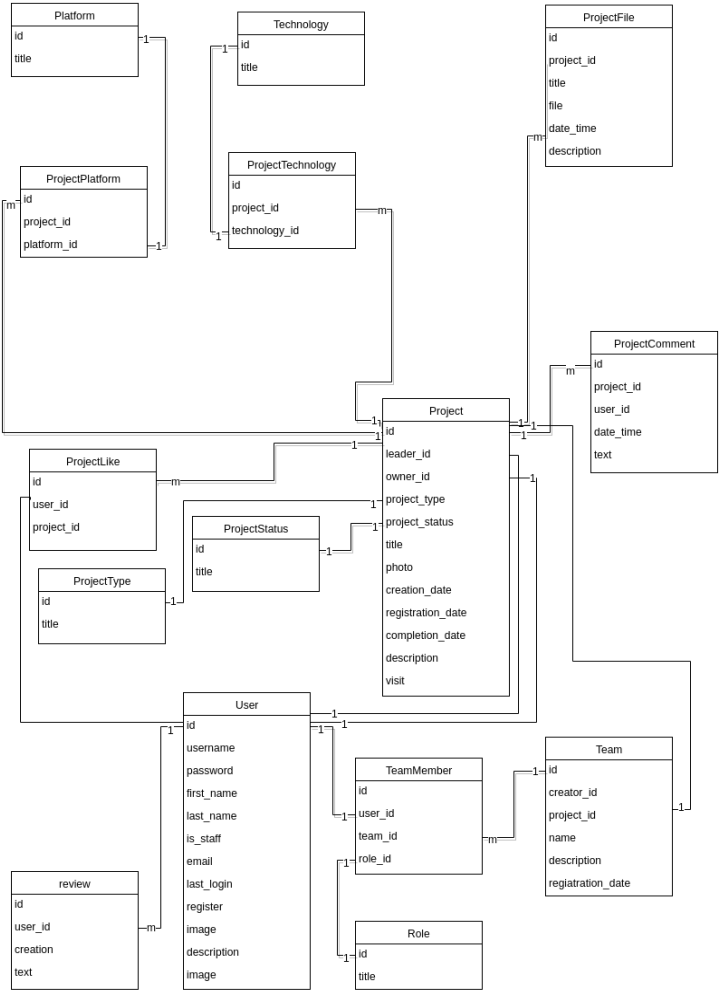


Рисунок 2.12 – ER-діаграма сутність-зв'язок для базової функціональності веб-порталу [91].

Таблиця 2.4 Сутності системи

Сутність	**Атрибут**	**Тип**	**Опис**
Користувач (User)	id	Ціле число	Ідентифікатор користувачів
	username	Символьний	Логін
	password	Символьний	Пароль облікового запису користувача
	first_name	Символьний	Ім’я користувача

	last_name	Символьний	Прізвище користувача
	is_staff	Булевий	Флаг приналежності до адміністрації веб-порталу
	email	Символьний	Електронна адерса
	last_login	Дата/час	Дата/час останнього візиту на веб-портал
	register	Дата/час	Дата/час реєстрації
	image	Файл	Фото користувача
	desctiption	Текстовий	Додаткова інформація про користувача
Відгук (Review)	id	Ціле число	Ідентифікатор відгуку
	user_id	Ціле число	Ідентифікатор користувача який залишив відгук
	creation	Дата/час	Дата/час реєстрації відгуку
	text	Текст	Зміст відгуку
	is_checked	Булевий	Поле премодерації відгуку
Роль (Role)	id	Ціле число	Ідентифікатор ролі
	title	Символьний	Найменування ролі
Учасник команди (TeamMember)	id	Ціле число	Ідентифікатор учасника команди
	user_id	Ціле число	Ідентифікатор користувача
	team_id	Ціле число	Ідентифікатор команди
	role_id	Ціле число	Ідентифікатор ролі
Команда (Team)	id	Ціле число	Ідентифікатор команди
	creator_id	Ціле число	Ідентифікатор користувача
	name	Символьний	Назва команди
	description	Текст	Опис команди
	registration_date	Дата/час	Дата реєстрації команди
Проектна команда (ProjectTeam)	id	Ціле число	Ідентифікатор проектної команди
	team_id	Ціле число	Ідентифікатор команди
	project_id	Ціле число	Ідентифікатор проекту
Платформа (Platform)	id	Ціле число	Ідентифікатор платформи
	title	Символьний	Назва платформи
Технологія (Technology)	id	Ціле число	Ідентифікатор технології
	title	Символьний	Назва технології
Платформа проекту (ProjectPlatform)	id	Ціле число	Ідентифікатор платформи проекту
	project_id	Ціле число	Ідентифікатор проекту
	platform_id	Ціле число	Ідентифікатор платформи
Технологія проекту(ProjectTechnology)	id	Ціле число	Ідентифікатор технології проекту
	project_id	Ціле число	Ідентифікатор проекту
	technology_id	Ціле число	Ідентифікатор технології
Оцінка проекту	id	Ціле число	Ідентифікатор оцінки

(ProjectLike)	user_id	Ціле число	Ідентифікатор користувача
	project_id	Ціле число	Ідентифікатор проекту
Тип проекту (ProjectType)	id	Ціле число	Ідентифікатор типу проекту
	title	Символьний	Найменування типу проекту
Статус проекту (ProjectStatus)	id	Ціле число	Ідентифікатор статусу проекту
	title	Символьний	Найменування статусу проекту
Коментар до проекту (ProjectComment)	id	Ціле число	Ідентифікатор коментару проекту
	project_id	Ціле число	Ідентифікатор проекту
	user_id	Ціле число	Ідентифікатор користувача
	date_time	Дата/час	Дата/час додавання коментару
	text	Текст	Зміст коментару
Файл проекту (ProjectFile)	id	Ціле число	Ідентифікатор файлу проекта
	project_id	Ціле число	Ідентифікатор проекту
	title	Символьний	Найменування файлу
	file	Файл	Посилання на файл
	date_time	Дата/час	Дата/час додавання
	description	Текст	Опис файлу
Проект(Project)	id	Ціле число	Ідентифікатор проекту
	leader_id	Ціле число	Ідентифікатор користувача
	owner_id	Ціле число	Ідентифікатор користувача
	projectType_id	Ціле число	Ідентифікатор типу проекту
	projectStatus_id	Ціле число	Ідентифікатор статусу проекту
	title	Символьний	Найменування проекту
	photo	Файл	Зображення проекту
	creation_date	Дата/час	Дата/час початку створення проекту
	registration_date	Дата/час	Дата/час реєстрації проекту
	completion_date	Дата/час	Дата/час закінчення реалізації проекту
	description	Текст	Опис проекту
	visit	Ціле число	Кількість переглядів проекту

У той же час слід зазначити, що дана модель є неповною для завдання підтримки академічного підприємництва, оскільки не містить рішень по об'єктивному оцінюванню підтримуваних IT-проектів і компаній. Таке оцінювання неможливе без обробки інформації, пов'язаної з конкретними роботами, які виконуються в ході реалізації проекту. Саме перелік таких робіт і їх зв'язок з відповідальними виконавцями в команді, плановані терміни і прогрес виконання можуть дати необхідний фундамент для об'єктивних оцінок. До таких оцінок можна віднести можливі терміни завершення проекту або його окремих етапів, а також зрілість проектної команди. Таким чином, ми приходимо до необхідності розширення розглянутої вище моделі новою сутністю - «робота» (англ. Activity -

відповідно до прийнятої в проектному управлінні термінології). Очевидно, що такі роботи можуть групуватися в завдання, які в свою чергу будуть зводитись до дій і фаз в рамках узагальненої моделі життєвого циклу створення IT-продукту (див. 2.3.1.).

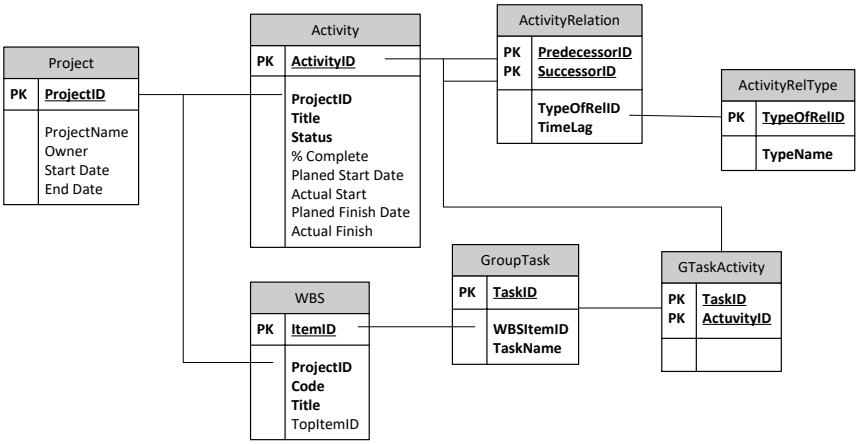


Рисунок 2.13 – ER-діаграма проектних робіт веб-порталу.

На рис. 2.13 представлена розширена ER-діаграма для виконання завдання обліку інформації, пов'язаної з окремо виконуваними завданнями на проекті. У запропонованій схемі сутність «проект» показана умовно, з єдиною метою показати зв'язок окремого проекту з безліччю робіт в цьому проекті. Кожна робота має набір атрибутів, який її характеризує, розгляд яких не є принциповим в даному випадку. Набагато важливішим є відношення слідування між окремими роботами, яких, як відомо [35] всього чотири: «старт-фініш», «старт-старт», «фініш-фініш» і «фініш-старт» (англ. Start-to- Finish, Start-to-Start, Finish-to-Finish, Finish-to-Start). Сутність таких відносин наступна.

Якщо між двома роботами є відношення слідування, що одна робота не може бути розпочата раніше, ніж закінчиться попередня, то тоді говорять, що між ними існує відношення «старт-фініш». Наприклад, не завершивши компіляцію програми неможливо її запустити на виконання, або не отримавши потрібні компоненти неможливо розпочати збірку виробу. Таке відношення характерне для робіт, виконання яких буде строго послідовним.

Якщо між двома роботами є відношення слідування, що одна робота не може бути розпочата раніше, ніж почнеться попередня, то тоді говорять, що між ними існує відношення «старт-старт». Таке відношення характерне для робіт, які повинні виконуватися паралельно і починатися синхронно.

Якщо між двома роботами є відношення слідування, що одна робота не може завершитися раніше, ніж почнеться інша, то тоді говорять, що між ними існує відношення «фініш-старт». Наприклад, батьківський процес не може завершити своє виконання до тих пір, поки не буде згенеровано синівський процес або поки не буде отриманий зворотний зв'язок від синівської процесу, який розпочав виконання. Таке відношення характерне для робіт, виконання яких буде послідовним.

Якщо між двома роботами є відношення слідування, що одна робота не може бути завершена раніше, ніж закінчиться попередня, то тоді говорять, що між ними існує відношення «фініш-фініш». Наприклад, батьківський процес повинен очікувати до тих пір, поки не завершаться всі синівські процеси, і тільки потім зможе завершитися сам. Таке відношення характерне для робіт, які повинні виконуватися паралельно і закінчуватися синхронно.

Роботи, між якими не задані відношення слідування, можуть виконуватися як послідовно, так і паралельно при наявності необхідних ресурсів.

Відношення проходження між окремими роботами моделюються на ER діаграмі (див. рис. 2.13) за допомогою сутності ActivityRelation, що має такі атрибути як: посилання на пару робіт (PredecessorID і SuccessorID), представлених парою примірників сутності Activity; тип відношення слідування, а так само цілочисельний атрибут Lag - моделює ситуацію з реального світу, коли роботи можуть мати відоме запізнювання / випередження в своїх відношеннях. Наприклад, нехай визначені 2 роботи А і Б, між якими визначено відношення слідування таке, що обидві роботи повинні виконуватися паралельно, але робота Б повинна початися з випередженням роботи А на 2 умовні одиниці часу - тоді значення атрибута Lag дорівнюватиме -2.

Представлені на діаграмі рис. 2.13 сутності WBS, GroupTask і GTaskActivity носять не принциповий допоміжний характер, і призначені для групування окремих робіт в групи і фази з метою адаптації їх подання у вигляді проектного графіка або інших способів подання виконуваних робіт. Зокрема, в ході реалізації даного проекту було запропоновано розробити

уявлення ходу виконання проекту за методологією Scrum [36], а саме через створення «панелі управління Скрам» (англ. Scrum dashboard), що представляє собою віртуальну дошку, простір якої поділений на області планованих, виконуваних і виконаних завдань і по якій переміщаються виконувані завдання.

2.2.5 Рішення в області користувацького інтерфейсу

Як попередник гнучких (Agile) методів створення програмного забезпечення, модель прототипування передбачає створення ескізів інтерфейсу на самих ранніх етапах проектування ПЗ. Крім того, що подібний підхід дає можливість отримати миттєвий зворотний зв'язок з боку майбутнього користувача, він так само стимулює творчі здібності розробника. Ефектом раннього створення користувацьких інтерфейсів є можливість виявлення раніше неочевидних функцій системи, необхідних користувачеві, як наприклад вибірка і відображення конкретних даних, різні сортування та подання. Зворотній зв'язок з користувачем, у процесі проектування призначеного для користувача інтерфейсу, залучає його до процесу створення ПЗ, що добре впливає на кінцевий результат.

Створення призначеного для користувача інтерфейсу проходило в ітеративній формі, коли спочатку створювалися ескізи, які обговорювалися і узгоджувалися членами команди, і лише потім концепція переносилася в програмний код. На рис. 2.14 наведені приклади рішень в частині призначеного для користувача інтерфейсу створюваного веб-порталу на етапі проектування призначеного для користувача інтерфейсу.

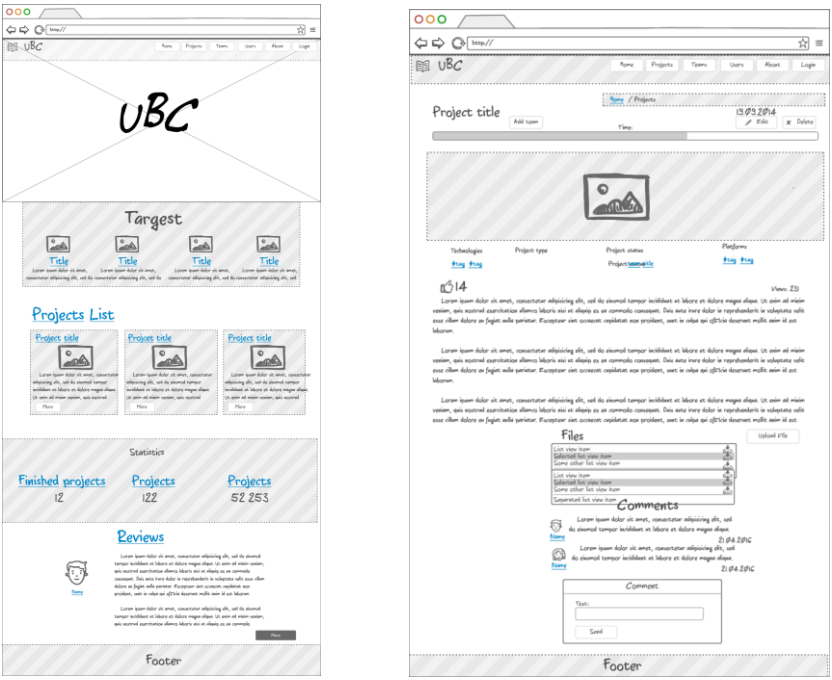


Рисунок 2.14 – Ескізи користувацького інтерфейсу веб-порталу [29].

Слід зазначити, що на етапі проектування призначених для користувача інтерфейсів було прийнято рішення про використання технологій адаптивної верстки, з метою можливості відображення контенту веб-сайту на різних мобільних платформах в доповнення до стандартного інтерфейсу персонального комп'ютера. Таке рішення вплинуло на вибір технологічних платформ для реалізації порталу.

2.2.6 Програмно-архітектурні рішення

Вимоги до створення веб-порталу, як складного веб-додатку, що має розвинений користувацький інтерфейс, взаємодія з базою даних і зовнішніми модулями однозначно диктують архітектуру системи у вигляді окремих компонентів, розділених таким чином, щоб модифікація одного з них надавала мінімальний вплив на інші.

В якості подібного рішення підходить архітектурний каркас або шаблон проектування «модель-уявлення-контролер» (англ. Model-View-Controller, MVC) [37, 38]. Такий шаблон проектування є невід'ємною частиною програмного фреймворка Django [39, 40], що забезпечує його вибір в якості базового фреймворка для розробки веб-порталу.

Концепція MVC розподіляє дані, представлення та обробку дій користувача на три компоненти:

- Представлення - відповідає за візуалізацію інформації;
- Контролер - забезпечує зв'язок користувача з системою;
- Модель - дані і методи роботи з цими даними.

Відповідно до прийнятої концепції було розроблено поділ програмного продукту на три основних функціональних блоки: блок призначеного для користувача інтерфейсу, блок бізнес-логіки і блок роботи з даними. Така концепція дозволяє в свою чергу інтегрувати новорозроблені модулі веб-порталу, забезпечуючи реалізацію інтерфейсу з модулями представлення і даних в контексті фреймворка Django.

Схема функціонування системи матиме вигляд(Рис. 2.15):

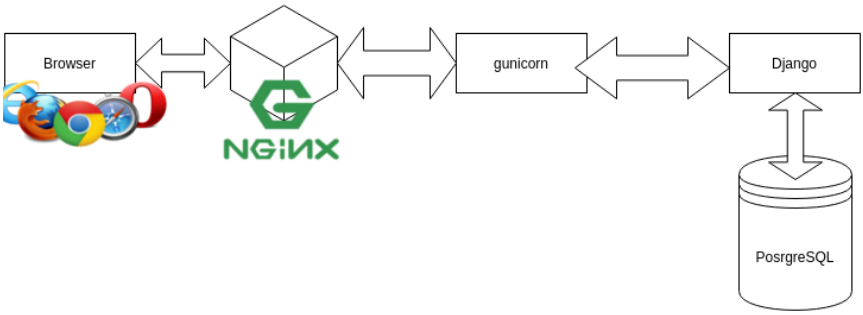


Рисунок 2.15 – Схема функціонування веб-порталу

Дана система функціонує на сервері ngixn [41]. Спочатку, користувач надсилає запит до серверу nginx з браузеру. Сервер перехоплює запит, та надсилає його до програми gunicorn[42], яка призначена для обробки запитів від сервера до Django [40]. Django, якщо необхідно звертається до бази даних PostgreSQL [43], обробляє інформацію, та відправляє відповідь на запит в зворотному порядку до браузеру користувача.

2.3 Інструменти відбору проектів і команд

Відбір проектів і команд був і, ймовірно, залишиться в майбутньому суб'єктивної діяльністю групи осіб - експертів, які, на підставі свого знання і досвіду дають оцінку майбутнього проекту. Як зазначалося в роботі [17], одним з перших, хто зрозумів перспективність технологій експертного оцінювання, був видатний радянський математик, академік, директор Інституту кібернетики АН УРСР В.М. Глушков. Значний внесок у розвиток методів експертного оцінювання внесли такі вчені, як С. Д. Бешелев, О.Ф. Волошин, Ф. Г. Гурвич, А.В. Добров, О.І. Ларічев, Б. Г. Литвак, Б. Г. Миркин, А. І. Орлов, В. Парето, В.В. Подиновский, Д.А. Поспєлов, Т. Сааті, Г. Саймон, Ю.І. Саєнко, Г. Фішер, К.Дж. Ерроу [44, 45]. Ними розроблено ряд моделей і методів експертного оцінювання та основи математичної теорії прийняття рішень.

В даний час на практиці добре зарекомендували себе такі методи експертної оцінки, як Делфі [46], парних порівнянь [47], аналізу ієрархій. Останній, за твердженням автора методу Т. Сааті, «дозволяє групі людей взаємодіяти з певної проблеми, модифікувати свої судження і в результаті об'єднати групові судження відповідно до основного критерію» [48]. В якості перспективного методу відбору можна згадати метод на основі нечіткого логічного висновку, описаного в монографії під редакцією проф. Ю.П. Кондратенко [49].

Слід зазначити, що багато «західних» університетів на практиці використовують більш спрощені методи відбору проектів [50, 51], деякі з яких наведені в підрозділі 2.2.1. Однак такі спрощені підходи можуть бути виправдані лише для невеликих проектів, що реалізуються силами університетських команд і студентських стартапів, в той час як сама проблема вибору проектів значно глибше, про що свідчать дослідження сучасних вчених [52 - 54].

2.3.1 Спрощені методи відбору перспективних проектів

2.3.1.1 Методика відбору проектів Технологічного університету Сіднея

Технологічний університет Сіднея (Австралія) пропонує диференціювати IT-проекти, що виконуються університетом, всього за трьома критеріями [51]: 1) складність, 2) ризик для університету, 3) бюджет проекту.

З метою оцінки потенційної складності даним університетом запропонована «матриця цілей та методів» (рис. 2.16), яка поділяє проекти на 4 типи. Та звідки слідує додаткова класифікація проектів на «чіткі» (англ. well defined) і «розмиті» (англ. fuzzy projects) [51].

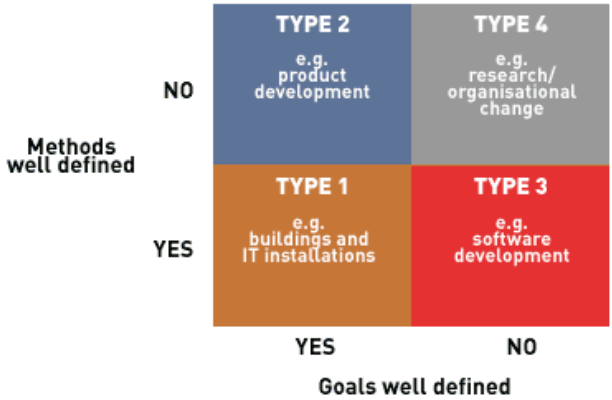


Рисунок 2.16 – Матриця цілей і методів відбору проектів Технологічного університету Сіднея

Отримана оцінка складності безпосередньо впливає на оцінку ризику проекту для університету, оскільки так звані «розмиті» проекти несуть в собі великі ризики. Сама по собі процедура оцінки ризиків не відрізняється від загальноприйнятих в бізнесі підходів і включає в себе наступні дії:

- Ідентифікація, коли члени експертної групи виносять на обговорення факти наявності ризиків.
- Аналіз і ранжування, коли члени експертної групи проводять оцінку ризиків і виділяють найбільш важливі з них.

І, нарешті, розглядаються питання бюджету проекту, де приймається рішення про здатність або нездатність університету (або спонсора проекту) його підтримувати в ході реалізації.

Сам механізм відбору передбачає тільки якісні судження і не має математично обґрунтованого апарату прийняття рішення. Слід зазначити, що такий підхід може бути виправданий для невеликих проектів, що реалізуються силами університетських команд і студентських стартапів, в той час коли проблема вибору проектів значно глибше, про що свідчать

дослідження сучасних вчених [52, 53], в тому числі за участю вчених з того ж Технологічного університету Сіднея [54].

2.3.1.2 Методика відбору проектів університету Вісконсін-Медісон

Іншим прикладом відбору проектів є методика, запропонована в університеті Вісконсін-Медісон (США) [50]. В основу її покладено ідею так званої матриці пріоритетів, де вибір альтернативи здійснюється шляхом ранжування критеріїв відбору та розрахунку числового значення ваги по кожному з критеріїв порівняння для кожної альтернативи. Основними перевагами такого підходу є простота і наочність для експертної групи, що здійснює вибір.

Процедура відбору проектів включає в себе 5 основних кроків:

1. Визначення набору критеріїв і шкали вимірювання. На цьому кроці визначають набір критеріїв, за якими будуть відбиратися проекти. Як правило, це 6-12 основних критеріїв. Для кожного критерію визначається шкала оцінювання, за якою будуть порівнюватися проекти, а саме, наскільки повно відповідає розглянутий проект даному критерію.
2. Визначення відносної ваги кожного критерію. На даному етапі критерії сортуються в порядку їх важливості, і для кожного з них визначається його вага.
3. Формування матриці пріоритетів у вигляді табл. 2.5.

Таблиця 2.5 – Матриця приорітетів

Критерій		Вага	Оцінка		Результат	
			Проект А	Проект Б	Проект А	Проект Б
1	Критерій 1	w_1	a_1	b_1	$w_1 * a_1$	$w_1 * b_1$
• • •						
i	Критерій i	w_i	a_i	b_i	$w_i * a_i$	$w_i * b_i$
• • •						
n	Критерій n	w_n	a_n	b_n	$w_n * a_n$	$w_n * b_n$
Разом:					$\sum(w_i * a_i)$	$\sum(w_i * b_i)$

4. Групова робота з оцінки проектів. На даному етапі відбувається оцінювання кожного проекту на відповідність критеріям відповідно до шкали, розробленої на етапі №1. Результат заноситься в графу «Оцінка». Наприклад, в даній табл. 2.5 величина a_i буде дорівнювати оцінці відповідності проекту А критеріюі, а величина b_j – оцінці відповідності проекту Б критерію j. Після цього в графі «Результат» підраховуються добутки отриманих оцінок на вагу критерію і далі - сума оцінок по кожному з проектів.

5. Обговорення результатів і випуск підсумкового документа. Після виконання математичних розрахунків відбувається обговорення результатів і узгодження спільної позиції на основі консенсусу.

2.3.2 Відбір проектів методом аналізу ієрархії Сааті

2.3.2.1Метод аналізуі єрархій Сааті

Метод аналізу ієрархій (МАІ) [56] - математичний інструмент системного підходу до складних проблем прийняття рішень, розроблений американським математиком Томасом Сааті. Основне призначення методу – рішення задач вибору альтернатив за допомогою їх багатокритеріального рейтингування. Основне застосування методу – підтримка прийняття рішень за допомогою ієрархічної композиції завдання і рейтингування альтернативних рішень.

МАІ передбачає побудову моделі проблеми у вигляді ієрархії, що включає мету, альтернативні варіанти досягнення мети і критерії для оцінки якості альтернатив. Після цього слідує визначення приорітетів всіх елементів ієрархії з використанням методу парних порівнянь. Потім відбувається синтез глобальних приорітетів альтернатив шляхом лінійної згортки приорітетів елементів на ієрархії. Далі – перевірка суджень на узгодженість. І на закінчення – прийняття рішення на основі отриманих результатів [55].

Формально ієрархію можна представити таким чином. Нехай Н – кінцева множина, на якій задано бінарне відношення R, що володіє наступними властивостями:

– транзитивність: $\forall x, y \in H \mid (x, y) \in R \wedge (y, z) \in R \Rightarrow (x, z) \in R$;

– асиметричність: $\forall x, y \in H \mid (x, y) \in R \Rightarrow (y, x) \notin R$;

– антирефлексивність: $\forall x, y \in H \mid (x, y) \in R \Rightarrow x \neq y$.

Множину H назвемо ієрархією, якщо існує розбиття $H = L_0 \cup L_1 \ldots \cup L_n$ таке, що:

а) $\exists h_0 \mid \forall x \in H \ (h_0, x) \in R$ і $L_0 = \{h_0\}$;

б) $\forall i \neq j \, L_i \cap L_j = \varnothing$;

в) $\forall (i = 0,1,\ldots n-1) \ x \in L_i \Rightarrow \exists y \in L_{i+1} \mid (x, y) \in R$ і $\nexists z \in H \mid (x, z) \in R \wedge (z, y) \in R$;

г) $\forall (i = 1,\ldots n) \ x \in L_i \Rightarrow \exists y \in L_{i-1} \mid (y, x) \in R$ і $\nexists z \in H \mid (y, z) \in R \wedge (z, x) \in R$.

Множини L_i називають рівнями ієрархії, а елемент h_0 – вершиною.

Нехай маємо ієрархію $H = L_0 \cup L_1 \ldots \cup L_n$. Виділимо елемент h, який не входить в останній рівень ієрархії, тобто $h \in L_0 \cup \ldots \cup L_{n-1}$, і розглянемо множину його дочірніх елементів $h^- = \{x_1, \ldots, x_m\}$, для яких виконується умова в) наведена вище.

До кожного відношення $(h, x_j) \in R$ поставимо у відповідність w_j – приорітет елементу x_j відносно h. У MAI, як правило, накладають такі обмеження на значення w_j для елементу $h, w_j \in [0,1]$ і $\sum w_j = 1$.

Безпосередньо приорітети визначаються методом парних порівнянь, суть якого виражається в такий спосіб – будується зворотньо симетрична «матриця парних порівнянь» $D = \left[d_{ij} \right]$ порядку m, де d_{ij} – оцінка експертом ступеня переваги елемента x_i над x_j відносно h, причому $d_{ji} = 1 / d_{ij}$. Комірка d_{ij} може приймати значення, наведені в табл. 3.1 [56].

Таблиця 3.1 Оцінки переваги альтернатив в MAI

Ступінь переваги d_{ij}	**Смислове значення**
1	x_i = x_j
3	x_i слабо переважає x_j
5	x_i переважає x_j
7	x_i значно переважає x_j
9	x_i абсолютно переважає x_j
2, 4, 6, 8	Проміжні результати

В даний час розроблено декілька методів розрахунку на основі $D = \left[d_{ij} \right]$, що відрізняються різною складністю і точністю [57], які зводяться до виділення власного вектора матриці D. Однак на практиці можна визначати приорітет як середнє геометричне рядків матриці попарних порівнянь з наступною нормалізацією всіх складових отриманого вектора за формулою (3.1) [56]:

$$w_j = \frac{\sqrt[m]{\prod_{k=1}^{m} d_{jk}}}{\sum_{j=1}^{m} \left(\prod_{k=1}^{m} d_{jk} \right)} \qquad (3.1)$$

Після побудови пріоритетів для всіх вузлів ієрархії, крім $x_i \in L_n$ – нижніх елементів ієрархії (альтернатив), виконується етап ієрархічного синтезу, пов'язаний з отриманням приорітетів альтернатив щодо головної мети - вузла h_0, $L_0 = \{h_0\}$. При цьому вектор пріоритетів $A = \{a_i\}$ для $x_i \in L_n$ розраховується за формулою (3.2), де n - число рівнів ієрархії, а $W_{k-1,k}$ – матриця приорітетів елементів рівня k щодо рівня k-1.

$$A = \prod_{k=1}^{n} W_{k-1,k}` \qquad (3.2)$$

Число рядків матриці $W_{k-1,k}$ дорівнює числу вузлів ієрархії на рівні L_{k-1}, а число стовпців дорівнює кількості вузлів ієрархії на рівні L_k. Значенням елемента матриці $W_{k-1,k} = \left[v_{ij} \right]$ буде приорітет $h_j \in L_k$ відносно $h_i \in L_{k-1}$ і $v_{ij} = 0$, якщо h_i і h_j не пов'язані між собою.

2.3.2.2 Ієрархія факторів, що впливають на вибір проекту

Дослідження, проведені раніше в роботах [58, 59], виділяють ряд факторів, що входять в цілі бізнес-центрів (інкубаторів). Ці фактори наведені в табл. 3.2 і ранжовані за частотою їх згадки в заявах про мету створення (місії) бізнес-центру.

Таблиця 3.2 – Типові цілі бізнес-центру (інкубатору)

Ціль бізнес-центру (інкубатора)	**Частота**
Сприяння конкурентоспроможності місцевої економіки	26%
Створення робочих місць	20%
Підтримка середнього і малого бізнесу з високим потенціалом зростання	19%
Стимулювання підприємницького духу і просування інновацій	18%
Підтримка конкретних секторів і розвиток кластера промисловості	8%
Інше, включаючи: 1.Визначення потенціалу та розробка регіональної конкурентоспроможності. 2.Оцінка ризиків компанії. 3.Зміцнення зв'язків між університетами, науково-дослідними інститутами і бізнес-спільнотою. 4.Сприяння росту й успіху нових технологічних компаній. 5.Передача знань в промисловість і більш ефективне використання інтелектуальної власності університету	9%

Як видно з таблиці 3.2, на момент проведених досліджень розвиток академічного підприємництва та взаємодія університетів з бізнесом входило в цілі менше 9% досліджених бізнес-центрів. Це робить актуальним побудову ієрархії цілей для університетського бізнес-центру (інкубатору) ІТ-компаній.

В ході виконаного опитування ряду вчених, викладачів, студентів і представників фінансового та промислового бізнесу були виділені основні дійові особи і цілі кожного стейкхолдеру. Ці дані наведені на рис. 3.1.

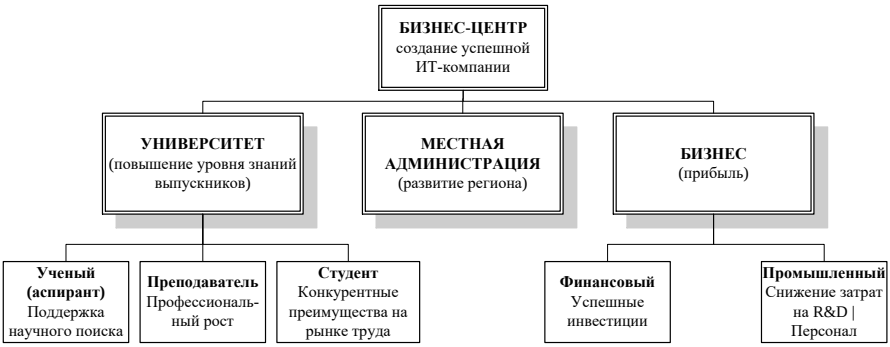


Рисунок 3.1 – Основні стейкхолдери та їх цілі

Так, для університету основною метою є підвищення рівня знань його випускників, оскільки вони не тільки формують привабливий імідж університету, але, оскільки випускники в основному є внутрішнім джерелом наукового та професорсько-викладацького складу, вони в кінцевому підсумку впливають на позиції в різних рейтингах, що визначають матеріально-фінансову забезпеченість університету.

Для вчених і викладачів основною метою є професійне зростання. Для дослідників це, в першу чергу, визначається доступом до ресурсів, що забезпечує їх наукові інтереси (наприклад, наукові лабораторії, обчислювальні центри і дослідно-конструкторські потужності).

Основною метою для студентів є отримання конкурентних переваг на ринку праці. При цьому в умовах України можна спостерігати парадоксальну ситуацію, коли студенти свідомо йдуть до вузів з більш слабким рівнем підготовки заради того, щоб мати можливість додаткового часу для роботи за майбутньою спеціальністю.

Що стосується бізнесу, то тут слід розділяти фінансовий бізнес, чия основна мета - успішні інвестиції в перспективний бізнес і промислові компанії, що йдуть на співпрацю з університетами заради зниження витрат на НДДКР. У будь-якому випадку основною метою бізнесу є отримання прибутку.

Необхідно розглянути державні, а також неурядові громадські організації, які формально не займаються отриманням прибутку від комерційної діяльності. У цьому випадку основною метою є ефективне витрачання бюджету для досягнення своїх цілей, що дозволяє їх розглядати аналогічно бізнес-структурам.

Університетський IT-бізнес-центр має за основну мету створення успішної IT-компанії (в разі створення IT-продукту або послуги) або формування успішного IT-колективу, здатного влитися у велику промисловість як необхідний персонал.

Визначивши основних стейкхолдерів і їх цілі, можна перейти до формування ієрархії відбору проектів для підтримки з боку бізнес-центру. Проведене опитування студентів, випускників, викладачів СФ НТУУ «КПІ» та ЧНТУ, представників бізнесу та органів місцевої влади дозволив сформувати наступну ієрархію, наведену на рис. 3.2.

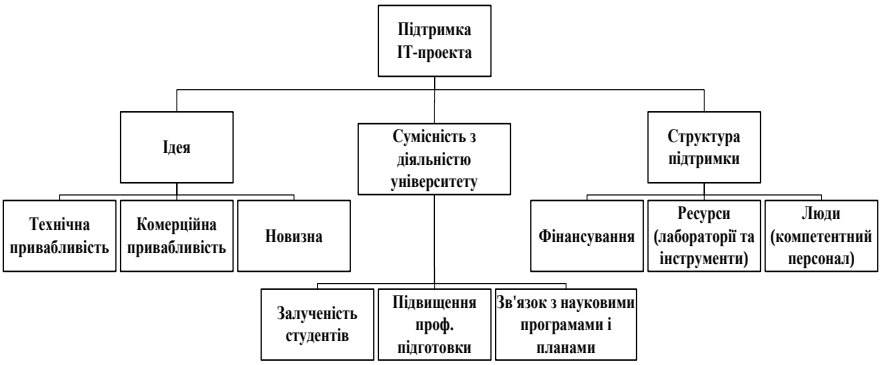


Рисунок 3.2 – Ієрархія факторів, що впливають на вибір проекту, який буде підтриманий з боку університетського бізнес-центру

Вочевидь, що в кожному конкретному випадку відбору проектів окремий фактор матиме свою власну вагу. Так, наприклад, в разі дефіциту ресурсів саме ці фактори будуть мати визначальну вагу в ухваленні рішення про вибір для підтримки конкретного проекту. Таким чином, при побудові інструменту відбору проектів і команд за допомогою MAI це визначає вимогу до можливості динамічної зміни приорітетів, а також безпосередньо самої ієрархії.

2.3.3 Основні проектно-технічні рішення

Інструмент відбору проектів був реалізований в якості додаткового модуля до раніше розробленого веб-порталу університетського бізнес центру (див. вище). Це наклало певні обмеження на вибір програмної платформи і набору інструментальних засобів, на користь технологій, сумісних з попереднім рішенням. А саме: Linux - в якості операційної системи сервера, програмного фреймворка Django і мови програмування Python. Так само, як зазначалося вище, в якості архітектурного рішення був застосований каркас або шаблон проектування «модель-уявлення-контролер» (MVC), що дозволив розділити функціональні блоки призначеного для користувача інтерфейсу, бізнес логіки і безпосередньо моделі даних.

У процесі рішення була побудована ER-діаграма сутностей для інструменту відбору проектів (див. Рис. 4.10), центральною сутністю якої є безпосередньо відбір проектів (на діаграмі сутність Review), що

характеризується термінами його проведення, а також отриманими результатами.

В окрему сутність винесена ієрархія критеріїв відбору проекту (на діаграмі ReviewHierarchy), яка характеризується належністю до конкретного відбору, переліком критеріїв і їх ієрархічною підпорядкованістю, а також вагою конкретного критерію.

Безліч експертів, які беруть участь у відборі проекту, характеризуються сутністю Expert, що має пряме посилання на користувача університетського веб-порталу. Відповіді експертів, отримані в ході відбору, зберігаються в сутності ExpertAnswer, що характеризується безпосередньо відбором, в якому бере участь експерт, критерієм, за яким він робить оцінку, посиланням на проект, який оцінюється і, нарешті, безпосередньо самою виданою оцінкою відповідності.

Сутність проекту (на діаграмі Project) - належить основній ER моделі розглянутій в п.4.1. веб-порталу та наведена тут для повноти сприйняття.

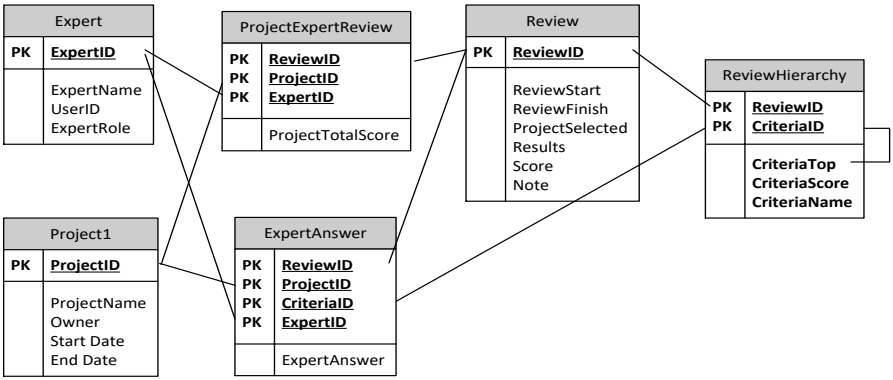


Рисунок 4.10 – Діаграма сутностей для інструментувідбору проектів.

Слід зауважити, що побудована ER діаграма однаково успішно підходить для реалізації відбору проекту, як за допомогою методології університету Вісконсін-Медісон (США) см. п. 2.3.1, так і методу аналізу ієрархій см. п. 2.3.2.

2.4 Вибір життєвого циклу

Модель життєвого циклу IT-продукту - це схема, яка пояснює, як будуть виконуватися дії з розробки IT-продукту, його експлуатації та супроводу (включаючи виведення з експлуатації) за допомогою опису вищеназваних дій. Така модель може бути лінійною або циклічною, її фази - послідовні або паралельні. В даний час процес пошуку оптимальних моделей і методологій розробки IT-систем триває, про що свідчать публікації в наукових виданнях і матеріалах міжнародних конференцій. У той же час всі ці моделі і методології можуть бути розділені на кілька категорій, які володіють загальними ознаками. Якщо взяти за критерій такого поділу послідовність виконання фаз і дій життєвого циклу, то можна буде розглянути класифікацію [60], наведену на рис. 2.6.

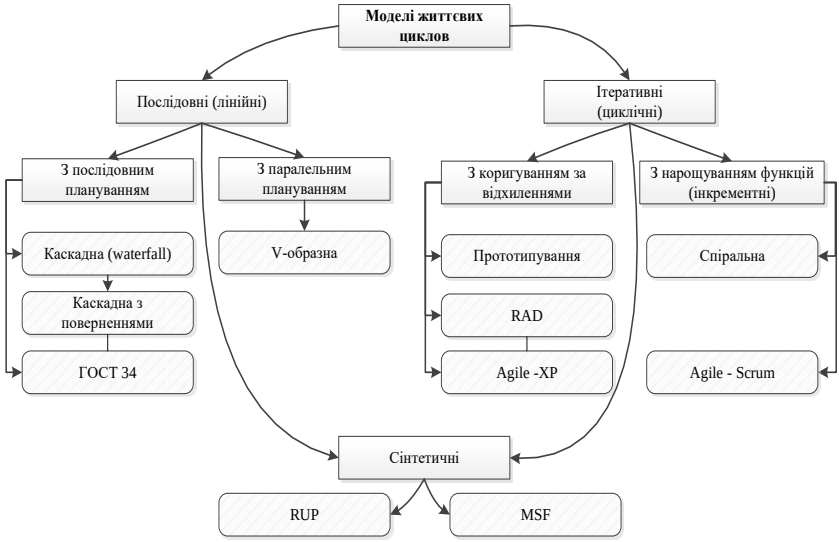


Рисунок 2.6 – Класифікація моделей життєвих циклів створення IT-продукту

Запропонована класифікація ділить моделі життєвих циклів на дві основні категорії - послідовні (лінійні), де фази життєвого циклу послідовно змінюють одна одну, і циклічні (ітеративні), коли в життєвий цикл

закладено циклічне повторення деяких фаз. У свою чергу, послідовні життєві цикли поділяються на моделі з послідовним і паралельним плануванням фаз.

До моделей з послідовним плануванням слід віднести класичну каскадну (водоспадну) модель [61], включаючи її модифікації, наприклад, методологію створення автоматизованих систем згідно ДСТУ 34 [62, 63].

До моделей з паралельним плануванням фаз слід віднести V-подібну модель і її модифікації. Основні ідеї, покладені в основу V-подібної моделі, знайшли своє відображення в міжнародному стандарті ISO / IEC 12207 [64]. Крім того, розроблена на її основі модель V Modell 97 і далі V Modell XT стала національним стандартом ФРН на розробку IT-систем [65].

Циклічні або ітеративні моделі можна розділити за принципом початку чергової ітерації. У загальному випадку таких принципів два:

• у зв'язку з виявленням відхилень в створюваному IT-продукті від очікуваного стану системи;

• у зв'язку з початком роботи над новою властивістю створюваного продукту.

До підкласу циклічних моделей ЖЦ з початком ітерації для створення нової якості або функціоналу системи можна віднести спіральну модель [66], запропоновану Баррі Боєм (Barry Boehm) в 1986 р.

До підкласу циклічних моделей ЖЦ з початком ітерації за виявленим відхиленням від очікуваного стану системи відносять різні варіанти моделі прототипування, включаючи Rapid Application Development, розроблену компанією IBM на початку 1990-х рр. [67]. Сюди входять різні «швидкі» [31] і «гнучкі» (Agile) методології, включаючи екстремальне програмування (англ. EXtreme Programming, XP) [30]. При цьому слід зазначити, що методологія Agile Scrum [36] вже відноситься до ЖЦ з початком ітерації для створення нової якості або функціоналу, саме тому «гнучкі» методології Agile віднесені відразу в обидва підкласу циклічних моделей.

До синтетичних моделям ЖЦ можна віднести Microsoft Solution Framework аж до версії 3 [68], Rational Unified Process [69].

Виходячи з того факту, що життєвий цикл стартап-компанії, як правило, закінчується виходом на ринок створеного продукту або послуги (далі компанія перетворюється в самостійний бізнес або поглинається більшим гравцем ринку), то життєвий цикл проекту, який підтримує бізнес-центр при університеті, буде побудований навколо життєвого циклу

створення такого продукту. Від правильності вибору даного життєвого циклу буде в значній мірі залежати успіх або провал створеної стартап-компанії. Але оскільки компанія створюється в університетському середовищі, її персонал часто має лише теоретичні уявлення про переваги і недоліки різних життєвих циклів.

Вплив зрілості організації та класу продукту, що розробляється, на вибір життєвого циклу показано в роботах [70, 71, 72]. Тому слід враховувати, що більшість стартапів, що опікаються бізнес-центром при університеті, буде організовано за участю молодих людей, які тяжіють до неформальної роботи, не знайомі на практиці з більшістю вищезазначених життєвих циклів і, більш того, не готові до їх використання.

У роботі Р. Т. Фатрелл, Д.Ф. Шафер і Л.Н. Шафер [21] вибір ЖЦ передбачається робити на основі такої методики:

1. Виходячи з вимог до проекту і продукту існуючої команди розробників, можливостей організації взаємин із замовником, а також на основі проектних ризиків, пропонується відповісти на ряд опитувань з двійковій системою відповідей (так/ні).

2. Здійснити пріоритизацію питань за ступенем важливості для проекту.

3. Вибрати модель, що має максимальне число позитивних відповідей з урахуванням обраних пріоритетів.

Наведемо вибірковий приклад із такого опитування (див. Табл. 2.1), який в оригінальній роботі [21] містить більше 30 питань, згрупованих за чотирма категоріями: 1) характеристики вимог до продукту; 2) характеристики колективу розробника; 3) характеристики кінцевого користувача; 4) типи і ризики проекту.

Таблиця 2.1 – Приклад питань опитування щодо вибору моделі ЖЦ

Питання	**Каскадна**	**V- подібна**	**Прото -тип**	**Спі- ральна**	**RAD**	**Інкре- ментна**
Чи легко визначаються вимоги?	Так	Так	Ні	Ні	Так	Ні
Часто будуть змінюватися вимоги?	Ні	Ні	Так	Так	Ні	Ні
Чи є інструменти новими для команди розробників?	Так	Так	Ні	Так	Ні	Ні

У даному опитуванні відповіді «Так» або «Ні» виходять на основі практик застосування різних ЖЦ командами розробників усього світу, що вивчаються Інститутом якості програмного забезпечення в Техасі, США.

Варто відзначити, що доступні автору на даний момент публікації вищеназваного інституту не містять досвіду застосування гнучких методів проектування IT-продуктів, як Agile (Scrum, XP, Kanban), що мають найбільшу популярність серед сучасних команд розробників. Однак міркування, висловлені у вищезгаданій роботі щодо інкрементної моделі ЖЦ можна поширити на Agile, виходячи із запропонованої вище класифікації життєвих циклів (розд. 2.3.2).

При своїй простоті запропонований метод не враховує приналежність системи, що створюється, до класу автоматизованих систем за їх призначенням, як це було зазначено вище. Але це може бути легко компенсовано шляхом введення додаткових питань в опитування, представлених в табл. 2.2.

Таблиця 2.2 – Додаткові питання, для урахування належності системи, що створюється, до класу АС

Питання: Чи належить система, що створюється, до **класу…**	**Каскадна**	**V-подібна**	**Прототипування**	**Спіральна**	**RAD**	**Інкрементна**
АСУ-ТП?	Так	Так	Ні	Ні	Ні	Ні
АСУ-П?	Так	Так	Так	Ні	Так	Так
ІС?	Ні	Ні	Так	Так	Так	Так

Запропонований у роботі [21] підхід може бути легко представлений у формальній формі, що дає можливість його реалізації у вигляді програмного інструменту. Наведемо нижче такий опис.

З кожною моделлю ЖЦ може бути асоційований двійковий вектор $C_k = \{c_1, \ldots c_i \ldots, c_n\}$, де 1 в позиції c_i означає позитивну, а 0 негативну відповіді на поставлене запитання, згідно табл. 2.1. Відповіді, дані користувачем при опитуванні, кодуються також у вигляді двійкового вектора $A = \{a_1, \ldots a_i \ldots, a_n\}$ за вищевикладеними правилами.

Тоді найбільш підходящою моделлю ЖЦ при реалізації проекту буде така, що відстань по Хеммінгу [73] між векторами буде мінімальною.

$$d_{C_k A} = \sum_{i=1}^{n} |c_i - a_i| \to min\,. \qquad (2.1)$$

Подібний підхід можна розширити для випадку, якщо в якості відповіді дається не тільки 0 або 1, а значення на інтервалі [0,1]. У такому випадку це завдання можна звести до пошуку мінімуму для узагальненої відстані між векторами A і C_k за Г. Мінковським (2.2), де для практичного застосування підходить $p=2$, що зводить формулу (2.2) до евклідової відстані.

$$\rho(x, y) = \left(\sum_{i=1}^{n} |x_i - y_i|^p \right)^{\frac{1}{p}}. \qquad (2.2)$$

2.5 Оцінка термінів робіт

Діяльність стартап-компанії, що знаходиться під опікою бізнес-центру при університеті, можна розглядати як обмежений в часі проект. Причому, з точки зору бізнес-центру, це буде проект зі створення успішної самостійної компанії, а, з точки зору самої компанії, це буде проект з виведення на ринок новостворюваного продукту (послуги) або проект по освоєнню компанією конкретних компетенцій, необхідних для її бізнесу. Вочевидь, що успіх або провал проекту опікуємої стартап-компанії стане аналогічним результатом проекту з точки зору бізнес-центру при університеті.

Життєвий цикл створення продукту (послуги) визначає узагальнені фази, в яких знаходиться проект з його початку і до його закінчення. Як правило, це фази планування, проектування, розробки, тестування та впровадження. Такі фази необов'язково лінійні, але можуть повторюватися і проходити одночасно. Кожна фаза характеризується цілями, які необхідно досягти, як, наприклад, розробити комплект проектно-технічної документації, виконати цикл випробувань, впровадити рішення на об'єкті замовника тощо. Фаза складається з набору робіт, пов'язаних між собою відношеннями черговості виконання. Наприклад, деяка робота не може бути розпочата раніше закінчення попередньої і разом вони складають мережеву модель, що описується графом переходів від однієї роботи до іншої. Фаза вважається завершеною за фактом успішного виконання всіх робіт. Робота

вважається успішно завершеною в разі отримання результату, що відповідає раніше заданим вимогам.

У проектному управлінні класичним підходом до моделювання проектів є метод PERT (англ. Program (Project) Evaluation and Review Technique) - техніка оцінки та аналізу програм (проектів) [74], де таке моделювання виконується за допомогою графів взаємозв'язків робіт (стрілки) і подій (вузли). Управління такими проектами здійснюється методом критичного шляху, де в результаті аналізу графа виявляється набір таких робіт, де зміна тривалості будь-якої з них призводить до зміни термінів реалізації всього проекту. При початковому плануванні робиться оцінка тривалості кожної роботи на підставі попереднього досвіду або типових галузевих даних, що дозволяє зі значною точністю не тільки оцінювати терміни проекту, але і здійснювати його управління.

Нижче наведено короткий опис даного методу [75]. Для кожної роботи i оцінюючий експерт формує три оцінки:

M_i – найбільш ймовірна оцінка трудовитрат;

O_i – мінімально можлива оцінка трудовитрат;

P_i – песимістична оцінка трудовитрат.

Тоді оцінку трудовитрат для роботи і розраховують за наступною емпіричною формулою (3.3).

$$E_i = \frac{P_i + 4M_i + O_i}{6}. \qquad (3.3)$$

Для розрахунку сумарної трудомісткості проекту, яка не буде перевищена з імовірністю 95%, застосовується формула(3.4).

$$E_{95\%} = \sum E_i + 2 * \sqrt{\sum \left[\frac{(P_i - O_i)}{6} \right]^2} \quad . \qquad (3.4)$$

У той же час для IT-проектів оцінка тривалості окремої роботи є нетривіальним завданням, оскільки такі оцінки вкрай суб'єктивні і залежать від кваліфікації виконавців, особливо для незрілих організацій, якими є стартап-компанії [6]. В роботі [75] відзначається, що вищезгаданий метод буде працювати тільки в разі, якщо план-графік робіт буде містити всі роботи, необхідні для досягнення проектних цілей. Крім того, там же зазначається, що отримані оцінки слід, як мінімум, помножити на

емпіричний коефіцієнт «4» - відомий більшості практичних керівників IT-проектів, який пояснюється тим, що роботи по безпосередньому кодуванню складають не більше 25% від усіх робіт по проекту.

Необхідно відзначити, що моделі проекту, побудовані класичним методом PERT, тяжіють за формою до каскадного життєвого циклу розробки, який не відображає ітеративну природу розробки інформаційних технологій. До того ж основним капіталом будь-якої компанії в постіндустріальному суспільстві є люди, які не тільки володіють необхідними компетенціями, але й здатні до безперервного самовдосконалення і освоєння нових знань. Розуміння переваги ролі людей над процесами було і в період індустріального стрибка в СРСР, а в XXI ст. прийшло в країни з ринковою економікою завдяки вибуховому розвитку інформаційних технологій і далі в інші галузі економіки. Це обумовлює недостатність класичних способів моделювання ходу проекту і вимагає розширення таких способів за рахунок технік, здатних враховувати людську природу, як один із чинників ентропії створення продуктів і послуг.

Такою технікою може стати агентне імітаційне моделювання (англ. Agent-based model, ABM) процесу розробки продукту (послуги), засноване на обраній проектною командою моделі життєвого циклу розробки продукту (послуги). У цьому випадку на перший план виводиться виконавець (агент), що володіє своїми властивостями і особливостями поведінки. Тільки в конкретний момент часу з кожною роботою пов'язаний виконавець (агент), який може її завершити з певною ймовірністю, яка визначається властивостями самого агента.

Призначення пропонованої імітаційної моделі - оцінка досяжності підтримуваною компанією її виробничих цілей (випуск продукту, розробка нової послуги, придбання специфічних компетенцій, підвищення ефективності праці і т.і.). Така модель повинна включати в себе:

• життєвий цикл виконання проекту по досягненню цілей компанії, наприклад, створення продукту (послуги);

• набори типових робіт, з яких складаються фази вищеназваного життєвого циклу;

• переліки компетенцій, необхідних для виконання робіт, і нормовані тривалості цих робіт за умови їх виконання компетентним фахівцем.

Вхідними даними такої моделі будуть: реальний набір фахівців, наявних у компанії; планована мережева модель реалізації проекту;

стратегія компанії по розподілу робіт конкретним фахівцям; імовірнісні оцінки виконання робіт і інша специфічна для конкретного проекту інформація.

- В результаті прогону модель повинна відповідати на наступні питання:
- 1. Які терміни виконання проекту при обраній підприємством стратегії реалізації проекту?
- 2. Чи достатні виділені ресурси для реалізації проекту?

2.5.1 Концептуальна модель

Згідно з В.М. Томашевським [76], розробка концептуальної моделі є наступним кроком створення імітаційної моделі після формулювання проблеми і загальної постановки завдання. Таку концептуальну модель, як правило, описують на природній мові в термінах предметної області і тільки потім переходять до опису моделі на мовах моделювання. Далі від мов моделювання можна перейти або до програмної реалізації, або математичної моделі і її формального рішення. Наведемо тут опис нашої концептуальної моделі.

Як було відзначено в підрозділі 2.3, життєвий цикл проекту визначає узагальнені фази, в яких знаходиться проект з його початку і до його закінчення. Для стартап IT-проектів такий ЖЦ, як правило, збігається з ЖЦ створення продукту і включає в себе фази планування, проектування, розробки, тестування продукту і закінчується виходом продукту на ринок. Такі фази необов'язково лінійні, але можуть повторюватися і проходити одночасно. Кожна фаза характеризується цілями, які необхідно досягти на цій фазі, і складається з набору робіт.

Роботи пов'язані між собою відношеннями черговості виконання. Наприклад, деяка робота не може бути розпочата раніше закінчення попередньої і разом вони складають мережеву модель, що описується графом переходів від однієї роботи до іншої. Одинична робота вважається виконаною, якщо в результаті з'являється артефакт (документ або його частина, програмний код, матеріальний об'єкт, структурна одиниця, система) з наперед заданими властивостями. Фаза вважається закінченою, якщо в результаті виконання робіт, що входять до неї, отримано набір всіх необхідних артефактів.

З кожною роботою пов'язаний виконавець, який здійснює виконання даної роботи, який, з одного боку, конкретний представник компанії, який має відповідну компетенцію для виконання даної роботи, а, з іншого боку, суб'єкт, який грає певну роль і виконує певну функцію. Такого виконавця можна назвати діючим суб'єктом (агентом) в термінології UML. У реальних проектах, як конкретний індивід може грати кілька ролей, так і безліч індивідів можуть грати одну роль. При цьому люди є обмеженим ресурсом компанії як в кількісному, так і якісному відношенні, але, з урахуванням чинника опіки стартапу з боку бізнес-центру при університеті, можна розглянути можливість залучення відсутніх людських ресурсів з боку бізнес-центру.

В якості необхідних компетенцій для IT-проекту слід розглянути набір областей знання з SWEBoK [77].

- Software Requirements — вимоги до ПЗ;
- Software Design — проектування ПЗ;
- Software Construction — конструктування ПЗ;
- Software Testing — тестування ПЗ;
- Software Maintenance — супровід ПЗ;
- Software Configuration Management — управління конфігурацією;
- Software Engineering Management — управління IT-проектом;
- Software Engineering Process — процес програмної інженерії;
- Software Engineering Models and Methods — моделі та методи розробки;
- Software Engineering Professional Practice — опис критеріїв професіоналізму та компетентності;
- Software Quality — якість ПЗ;
- Software Engineering Economics — економічні аспекти розробки ПЗ;
- ComputingFoundations — основи обчислювальних технологій, що застосовуються у розробці ПЗ;
- MathematicalFoundations — базові математичні концепції і поняття, що застосовуються в розробці ПЗ;
- Engineering Foundations — основи інженерної діяльності.

Для цілей моделювання можна покласти для кожного виконавця оцінку його компетенцій в інтервалі [0..1]. Тоді деяка нормована робота

буде виконана конкретним виконавцем за час, обернено пропорційний до цієї оцінки.

Кроком моделювання для даної моделі буде нормований умовний час виконання роботи. Іншими словами, в рамках моделі одна робота виконується за одну одиницю часу, якщо виконавець даної роботи має компетенцію для її виконання із значенням 1. Роботи, об'єктивна тривалість яких займає більше часу, в рамках моделі повинні бути розбиті на декілька більш дрібних робіт. Модель може включати в себе спеціальні роботи - «навчання», які змінюють рівень компетенції виконавця, який її «виконує», тим самим модель може моделювати витрати на підготовку фахівців.

Вхідними даними моделі буде обраний компанією життєвий цикл реалізації IT-проекту, перелік планованих робіт, компетенції фахівців компанії і стратегія їх розстановки по роботах. Результатом моделювання стане час, необхідний для реалізації модельованого IT-проекту.

На моделі з'являється певна множина одиничних робіт, котрі і будуть виконувати агенти. Зрозуміло, що швидкість виконання тієї чи іншої одиничної роботи повністю залежить від величини компетенції працівника. Варто зазначити, що при необхідності агенти матимуть змогу взаємодіяти один з одним, і в такому випадку робота буде виконуватися швидше. По факту виконання певної кількості робіт компетенція агентів буде підвищуватися у відповідних областях знань.

Для того, щоб провести перевірку даної поведінки, кращим варіантом буде зімітувати роботу персоналу за допомогою агентного підходу. Саме імітаційне моделювання надасть змогу відтворювати ті процеси, які протікають на реальних проектах (в нашому випадку - по розробці програмного продукту).

Таким чином, описана вище стратегія повинна визначати кількість часу виконання X робіт при різноманітних розстановках персоналу та його кількості, що в результаті допоможе заздалегідь визначити, чи впорається університетська команда зі своїми задачами в контрактний період. Виходить, якщо змодельований час не збігатиметься зі строками, то можна буде провести наступну перестановку персоналу, відправити працівника на підвищення кваліфікації (тим самим доповнити модель фіктивними задачами), призначити декількох осіб на одну задачу тощо. [17].

Звісно, таке дослідження передбачає планування та проведення експериментів, у ході яких будуть збиратися необхідні дані. Планування експерименту — це розробка такого плану експерименту, який дає

можливість за мінімальної кількості прогонів моделі і за мінімальних витрат ресурсів зробити статистично значимі висновки або знайти оптимальні рішення щодо функціонування системи. Під час планування експериментів, як правило, визначають [78]:

- вхідні дані для кожного експерименту і кількість прогонів імітаційної моделі;
- тривалість одного прогону моделі і перехідного процесу моделювання;
- стратегію збору даних під час кожного прогону моделі;
- методи оцінювання точності вихідних даних і побудови довірчих інтервалів;
- чутливість моделі до вхідних даних, різних видів розподілів випадкових величин, сценаріїв поведінки модельованої системи;
- умови і сценарії проведення експерименту;
- умови генерування потоків випадкових чисел у системі моделювання та імовірнісних вхідних даних;
- стратегію досягнення мети експерименту.

Організація експериментів з моделлю – одна із найважливіших задач. Так як в даному випадку в моделі будуть враховуватися випадкові фактори, то така імітаційна модель буде класифікована як стохастична. Це означає, що під час моделювання необхідно провести велику кількість прогонів моделі, як із різними вхідними даними, так і з різними послідовностями випадкових чисел. У ході експерименту з імітаційною моделлю буде одержано безліч даних, які мають бути структуровані та інтерпретовані для використання під час прийняття рішень стосовно реального проекту.

2.5.2 Математичний опис

В підрозділі розглядається одне із завдань комбінаторної оптимізації, пов'язане з виникаючою на практиці проблемою розстановки персоналу по безлічі робіт у разі, якщо на безлічі робіт можуть існувати обмеження на порядок їх виконання, а персонал має нерівнозначну підготовку [79].

Визначимо $A = \{a_1, ...a_i..., a_n\}$ як безліч робіт, необхідних для завершення фази життєвого циклу розробки продукту. Між деякими парами

робіт існує обмеження проходження $a_i \rightarrow a_j$. Це означає, що роботу a_j неможливо почати раніше роботи a_i. Таке обмеження можна задати за допомогою орієнтованого графа $G = (V,\ D)$, де кожній вершині з множини $V = \{1,\ ...,\ n\}$ буде відповідати одна робота з множини А, а множина дуг $D = \{(i,\ j)\,|\,i,\ j \in V;\ i \rightarrow j\}$ відповідає обмеженню проходження. Слід зазначити, що такий граф G має бути ациклічним.

Нехай $C = \{c_1,\ ...c_k ...,\ c_s\}$ – безліч співробітників компанії, де c_k – конкретний співробітник, який здатний виконати роботу a_i за час $t_{ik} > 0$, званої тривалістю виконання роботи. В одиничний момент часу один співробітник може виконувати одну роботу, переривання робіт не допускається. Очевидно, що такі тривалості виконання робіт вищеназваними співробітниками утворюють множину T – суть декартів добуток множин $A\ i\ \ C, T = A \times C = \{(a,c)\,|\,a \in A,\ \ c \in C\}$.

Тоді виникає завдання оптимальної розстановки наявних співробітників по роботах так, щоб загальний час виконання всіх робіт з множини А був мінімальним і не порушувалося відношення слідування.

Згідно з А.А. Лазарєвим і Е.Р. Гафаровим, ця задача відноситься до задачі комбінаторної оптимізації. Слід зазначити, що в такій постановці задача відрізняється від відомої задачі «про призначення», де вимагається мінімізувати час, витрачений на виконання усіх завдань, і яка має поліноміальну складність.

Очевидно, що загальна кількість усілякого розподілу співробітників по роботах складає m^n, де n – кількість робіт, m – кількість співробітників. Тобто нерозумно розв'язувати задачу простим перебором. У той же час для деяких випадків розв'язання очевидне. Отже, розглянемо два крайні випадки:

1. Існуючі обмеження слідування пропонують виконувати усі роботи послідовно від a_1 до a_n.

2. Відсутні обмеження слідування, тобто усі роботи від a_1 до a_n можуть виконуватися паралельно.

Неважко показати, що при розбитті потоку завдань на частини усі інші варіанти зводяться до цих двох крайніх випадків, де деякий набір робіт $a_i..a_j$ послідовний або паралельний.

Зважаючи на той факт, що одна робота може виконуватися тільки одним співробітником, введемо функцію $f(i,k)=1$, якщо робота a_i виконується співробітником c_k, і $f(i,k)=0$ в усіх протилежних випадках.

Тоді у разі існування обмеження на послідовне виконання робіт розв'язання задачі про розподіл персоналу зводиться до пошуку такої функції $f(i,k)$, коли $\forall i \in [1,n] \exists k \in [1,s] \mid f(i,k)=1$, вираз (1) буде мінімальним.

$$T_{\min} = \sum_{k=1}^{s}\sum_{i=1}^{n} t_{ik} f(i,k) \to \min . \quad (1)$$

Для того, щоб знайти дану функцію, будемо міркувати так. Розкладаємо вираз (1) на 2 доданки:

$$\sum_{k=1}^{s}\sum_{i=1}^{n-1} t_{ik} f(i,k) + \sum_{k=1}^{s} t_{nk} f(n,k) . \quad (2)$$

Оскільки $\forall i \in [1,n]$, $k \in [1,s] \mid t_{ik} f(i,k) > 0$, формула (2) буде мінімальна, якщо обидва доданки мінімальні. При цьому мінімум другого доданка досягається в тому випадку, якщо знайдено таке k, при якому значення t_{nk} мінімальне і $f(i,k)=1$. Іншими словами, знайдено співробітника з індексом k, який виконає останню роботу a_n за найменший час. Продовживши так міркувати для першого доданка, отримуємо, що для пошуку всіх значень k, де функція $f(i,k)=1$, необхідно знайти $\min(t_{ik})$ для кожної роботи a_i.

У разі відсутності обмеження проходження, що дає можливість виконувати роботи паралельно, завдання зводиться до пошуку такої функції $f(i,k)$, коли значення виразу (3) буде мінімальним.

$$T_{\min} = \max_{1 \le k \le s} \left(\sum_{i=1}^{n} t_{ik} f(i,k) \right) \to \min . \quad (3)$$

Графічно це можна проілюструвати таким чином (таблиця 1).

Таблиця 1 – Матриця призначень співробітників на роботи

	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8
c_1		+		+	+			
c_2	+						+	+
c_3			+			+		

Уявімо множину T у вигляді матриці, де рядки будуть співробітники, а стовпці – роботи. Тоді значенням осередку t_{ik} буде час, який витрачається співробітником для виконання роботи i. Відзначимо осередки в таблиці, де функція $f(i,k)=1$. Тоді сума таких зазначених осередків буде мінімальною по тривалості виконання всіх робіт для строго послідовного обмеження проходження, а максимум сум по рядках – мінімальним за тривалістю виконання всіх робіт при повній відсутності обмеження проходження.

Покажемо, що при відсутності обмеження проходження, що дає можливість виконувати роботи паралельно, дана задача відноситься до класу NP.

Дійсно, розглянемо задачу для двох виконавців, обидва з яких однаково компетентні. Мінімальний час виконання всіх робіт, при можливості діяти двом виконавцям паралельно, буде досягнуто тоді, коли роботи між ними можна розділити порівну, тобто $\forall if(i,1) \neq f(i,2)$ і $\sum_{i=1}^{n} t_{i1} f(i,1) - \sum_{i=1}^{n} t_{i2} f(i,2) \to 0$, але при умові, що $t_{i1} = t_{i2}$. Це означає, що задача зводиться до класичної «задачі про ранці» (англ. knapsackproblem), тобто відноситься до класу NP-повних задач.

Отриманий висновок робить обґрунтованим використання засобів імітаційного моделювання у вирішенні даного завдання.

2.5.3 Моделювання мережами Петрі строків завершення ІТ-проекту

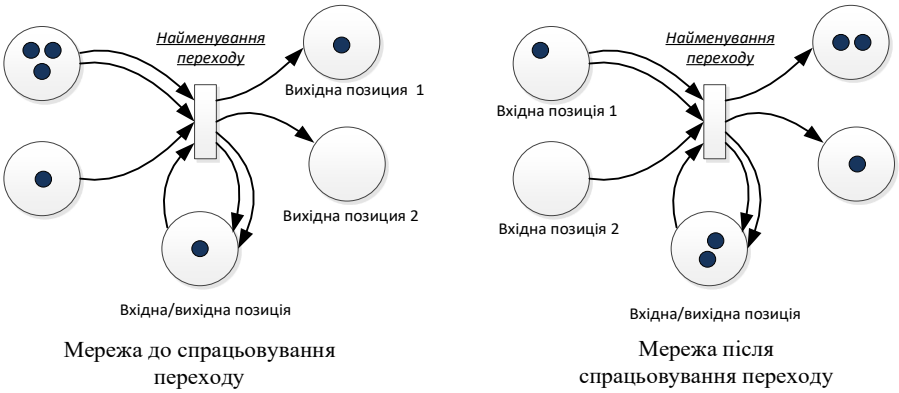
В даний час моделювання систем мережами Петрі [80] користується великою популярністю в академічному середовищі, завдяки своїй наочності і розвиненому математичного апарату. В Україні, значний внесок в дослідження мереж Петрі та побудови на їх базі систем моделювання внесла проф. І.В. Стеценко [81-83].

Мережа Петрі є двочастковим орієнтованим мультіграфом $G = \langle P, T, R \rangle$, що складається з вершин двох типів: позицій $P = \{ p_i \}$ і переходів $T = \{ t_j \}$, до того ж $P \cap T = \varnothing$, з'єднаних між собою ориентованними дугами $R = \{ (a,b) \}$, таким чином, що якщо $a \in P$, то $b \in T$ і якщо $a \in T$, то $b \in P$. Тобто вершини одного типу не можуть бутиз'єднані між собою.

У позиціях можуть бути розміщені маркери (мітки, фішки), які здатні переміщатися по мережі в результаті подій. Стан мережі однозначно визначається її маркуванням - розподілом маркерів по позиціях, яке можна представити у вигляді вектора

$M = \langle m_1, \ldots, m_i, \ldots, m_n \rangle$, де m_i відповідає кількості маркерів в позиції p_i.

Подією називають спрацьовування переходу, при якому мітки з вхідних позицій переміщаються в вихідні позиції. Обов'язковою умовою для спрацьовування переходу є наявність у всіх вхідних позиціях маркерів, кількість яких більше або дорівнює кількості дуг, що з'єднують вхідну вершину з переходом. Після спрацьовування переходу кількість маркерів у вхідних вершинах зменшується на кількість дуг, а у всіх вихідних вершинах додаються маркери в кількості, яка дорівнює кількості дуг, що з'єднують перехід із вхідною вершиною. На рис. 3.3 показаний приклад мережі Петрі в станах до і після спрацьовування переходу.



Мережа до спрацьовування переходу

Мережа після спрацьовування переходу

Рисунок 3.3 – Приклад мережі Петрі в станах до і після спрацьовування переходу

Мережу Петрі називають стохастичною, якщо на спрацьовування переходу накладається додаткове обмеження, пов'язане з ймовірністю такого спрацьовування. Висока залежність від суб'єктивних якостей ІТ-команди, таких, як рівні знань і компетенції, а також рівень невизначеності, який несе в собі кожна робота, дозволяє ввести поняття ймовірності завершення роботи в запланований термін і ймовірності неуспішного завершення окремої роботи. Що дає можливість успішного використання стохастичних мереж Петрі для моделювання моделей ЖЦ створення ІТ-продукту.

2.5.3.1 Моделювання мережами Петрі послідовних моделей життєвих циклів створення продуктів

Моделювання класичних послідовних моделей мережами Петрі не представляє собою складності завдяки їх лінійності. Для моделювання скористаємося методом поступової деталізації станів і переходів, який по суті є добре відомим способом проектування інформаційних систем методом зверху-вниз.

Початкова мережа Петрі для моделі проекту може бути представлена у вигляді двох послідовних станів «Проект розпочато» і «Проект закінчено», а так само одного переходу «Виконати проект» (рис. 3.4). Очевидно, що така мережа кінцева.

Виконаємо деталізацію станів і переходу початкової мережі. Так перехід «Виконати проект» можна розкрити до стану «Проект виконується», з попереднім переходом «Розпочати проект» і з подальшим переходом «Завершити проект». У той же час стан «Проект розпочато» можна деталізувати в такі стани: «Цілі <проекту> визначені», «Ресурси <на проект> виділені» і «Терміни <проекту> призначені». Аналогічні дії можна виконати для стану «Проект закінчено» (рис. 3.5).

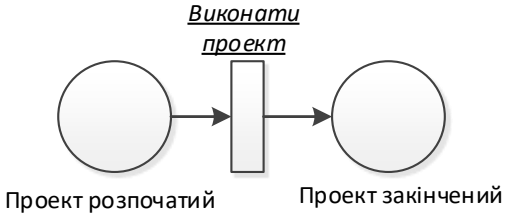


Рисунок 3.4 – Початкова мережа Петрі, що моделює проект створення ІТ-продукту

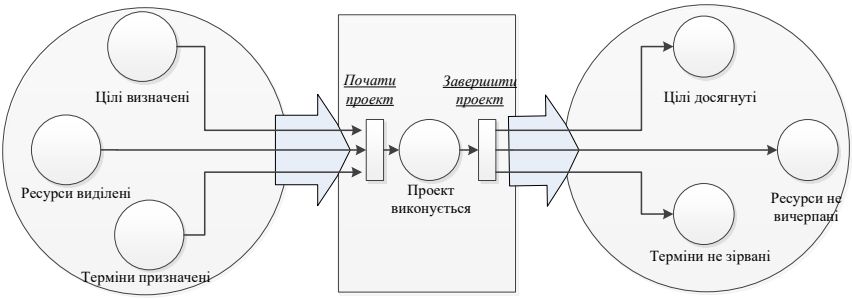


Рисунок 3.5 – Деталізація мережі Петрі, що наведена на рис. 3.4

Слід звернути увагу, що перехід «Розпочати проект» може спрацювати лише в разі знаходження маркерів мережі у всіх трьох попередніх станах. А спрацьовування переходу «Завершити проект» робить «істинними» його наступні стани.

Представлена на рис.2.2 мережа Петрі може бути базовою для будь-якої моделі життєвого циклу розробки ІТ-продукту, де стан «Проект виконується» деталізується у відповідності зі своєю моделлю.

2.5.3.2 Деталізація стану «Проект виконується» на прикладі моделі життєвого циклу розробки «швидке прототипування»

Як зазначалося в підрозділі 2.2, модель життєвого циклу швидкого прототипування вельми популярна серед студентів і молодих стартап-компаній. Тому моделювання даного ЖЦ становить практичний інтерес.

На рис. 3.6 представлена мережа Петрі, що деталізує стан «Проект виконується» для моделі життєвого циклу розробки «швидке прототипування»увикладі Р. Фатрела та ін. [21]. Як видно з рис. 3.6, перехід «Почати проект» ініціалізує стани «План-графік проекту не готовий» і «Аналіз не актуальний». Перехід «Виконати аналіз» враховує факт, що аналіз проекту не може бути початий за умови неготовності плану-графіка

проекту, тобто не враховано час і ресурси на виконання такого аналізу, а також наступних робіт. Перехід «Проектувати систему» дозволяє дії з розробки ЛМІ, БД, функцій системи та програмного коду, які в реальних проектах, як правило, виконуються паралельно. Стан «Система розглянута» дозволяє будь-який з двох переходів «Відхилити ...» і «Схвалити систему». Схвалення системи призводить до подальших переходів до закінчення проекту, в той час як відхилення системи повторює цикл планування проекту та аналізу системи (в даному випадку на предмет майбутніх змін).

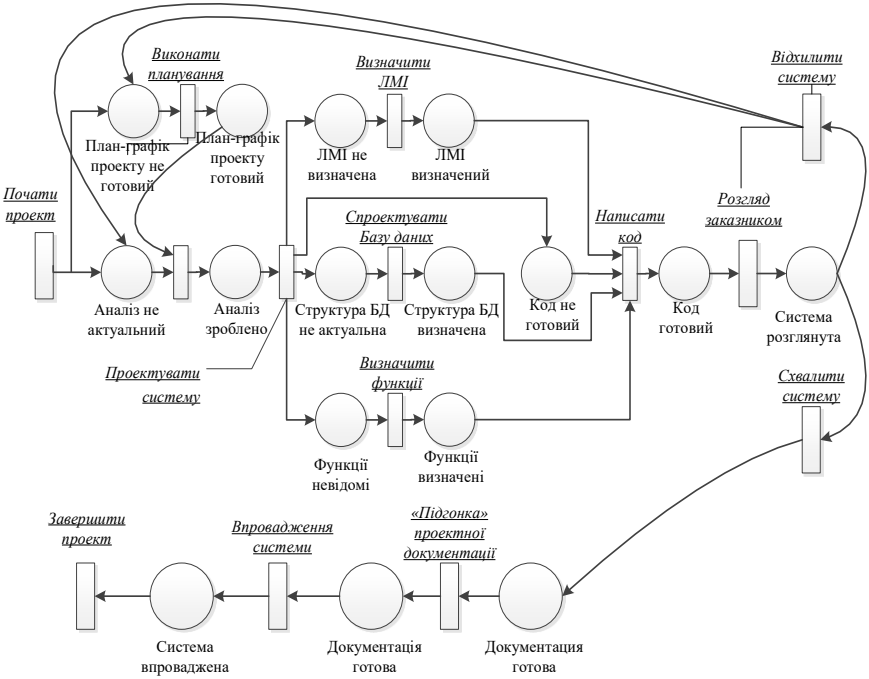


Рисунок 3.6 – Мережа Петрі для стану «Проект виконується» моделі ЖЦ швидкого прототипування

На відміну від описаної вище узагальненої моделі [21], де цикл прототипування не включає в себе необхідність перегляду планів-графіків проекту, повторення аналізу системи, перегляду бази даних (БД), функцій системи і людино-машинного інтерфейсу (ЛМІ), запропонована мережа

враховує цю необхідність, що робить її більш наближеною до реальних практичних IT-проектів.

У представленому вигляді мережа Петрі ще не враховує потребу в ресурсах і часі, а значить не може бути використана для заявлених цілей імітаційного моделювання.

2.5.3.3 Моделювання мережами Петрі моделі розробки Scrum

У підрозділі 2.3 зазначалося, що найбільш відомим і популярним методом "гнучкої" (Agile) розробки IT-продуктів є методологія Scrum [36]. На відміну від основних моделей життєвого циклу розробки, методологія Scrum не вказує певну послідовність розробки продукту і не визначає склад робіт. Замість цього дана методологія жорстко фіксує час ітерації («спринту»), за який повинні бути отримані конкретні результати, так званий «інкремент продукту». При цьому склад необхідних компонентів і функціональності «інкремент продукту» визначається командою безпосередньо перед початком «спринту».

Найбільш простий спосіб побудувати мережу Петрі для такого ЖЦ - це описати всі доступні стани, в яких перебуває проект, і з'єднати дані стани переходами. Особливістю такої мережі для методології Scrum буде те, що стан «Daily Scrum» може повторюватися циклічно, і те, що після огляду продукту може повторитися цикл розробки, так званий «спін» (рис. 3.7). Незважаючи на гадану тривіальність такої мережі, згодом можна деталізувати цю мережу розкриваючи окремі стани, або окремі переходи.

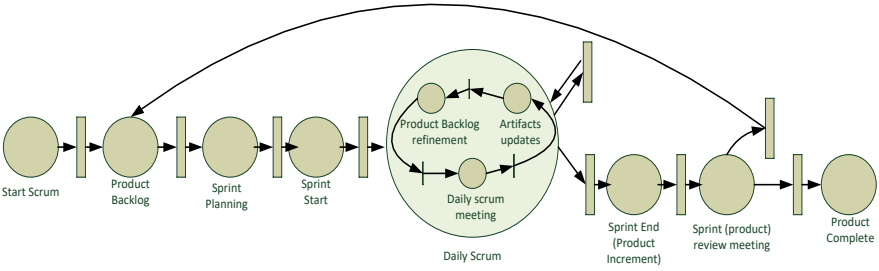


Рисунок 3.7 – Узагальнена мережа Петрі, що моделює виконання IT-проекту за методологією Scrum

Однак у наведеній вище мережі є істотний недолік: вона не в змозі моделювати проект в залежності від вихідних даних, таких як обсяги вхідних даних і доступних ресурсів, ймовірності успіху у виконанні окремих завдань і т.д. Таке моделювання в мережах Петрі можна виконати як за допомогою безлічі «фішок» в стартових позиціях мереж, так і безліччю дуг [81]. Приклад такої мережі, що моделює виконання IT-проекту за методологією Scrum, представлений на рис. 3.8.

Початковий стан даної мережі регламентується зазначенням:

- кількості «призначених для користувача історій» в стані «Product Backlog», який моделює перелік функцій і властивостей реалізованого IT-продукту;
- наявності фішки в стані «No Sprint running», що дозволить згодом реалізувати перехід «Start new Sprint»;
- ймовірності вибору конкретного переходу для станів з безліччю переходів.

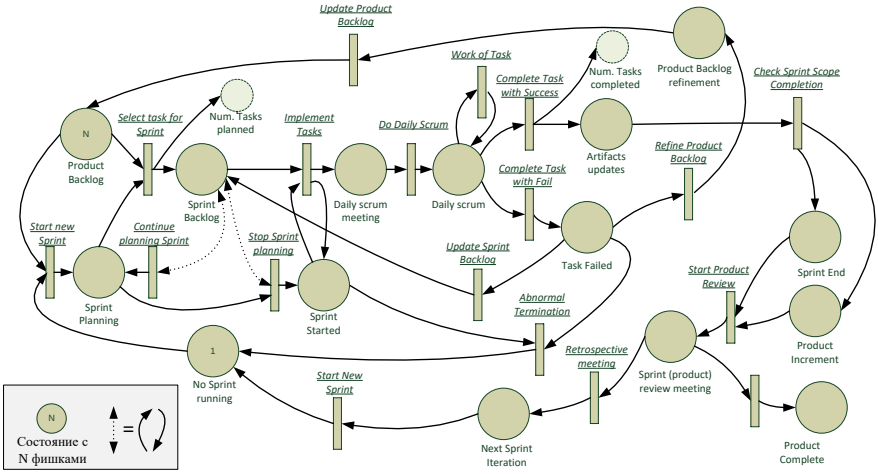


Рисунок 3.8 – Мережа Петрі, що моделює виконання IT-проекту за методологією Scrum

Зі стану «Start new Sprint» існує два переходи: або завершити планування і приступити до подальших кроків, або продовжити планування і розширити стан «Sprint Backlog» новою фішкою, що моделює передачу

завдання з множини «Product Backlog» в обсяг робіт на поточний спринт. При цьому мережа не дозволяє закінчити планування, якщо жодної роботи не було заплановано до виконання через «Sprint backlog».

Як тільки завдання додається в «Sprint backlog», вона може бути реалізована. І кожна така задача може бути завершена або успішно, або невдало, що моделюють відповідні стохастичні переходи. У разі неуспішного вирішення можливі три результати:

- перегляд вимоги Замовником і оновлення беклога продукту;
- оновлення беклога спринту, коли завдання переробляється в рамках поточного спринту;
- переривання спринту, коли спринт повністю скасовується і відбувається його перепланування.

Для наочності уявлення мережі, перехід, що забезпечує перевірку завершення спринту, показаний умовно. Втім, його реалізація очевидна. Перехід повинен спрацювати при досягненні двох умов: спринт знаходиться в стані виконання і кількість запланованих і завершених завдань рівна.

Слід зазначити, що стан «Sprint (product) review meeting» має два переходи, через які можна перейти або в стан завершення продукту (і проекту), або в стан прийняття рішення про необхідність в новій ітерації спринту. І в цьому немає суперечності, оскільки прийняття рішення про випуск продукту може бути прийнято до вичерпання всіх завдань з «Product backlog».

2.5.3.4 Мережа Петрі, що моделює роботу над проектними завданнями

Як зазначено вище в описі концептуальної моделі, проект характеризується безліччю робіт, спланованими строками закінчення і набором виконавців, причому всі три ці множини скінченні. У мережах Петрі це можна моделювати набором маркерів в станах і входами/виходами на переходах, див. [84].

Для реалізації окремого завдання у компанії, якою опікуються може бути 3 стратегії:

1. Використовувати власного співробітника з наявним рівнем компетенції.

2. Підвищити компетенцію співробітника до необхідного рівня і виконати роботу його силами.

3. Найняти на виконання ролі зовнішнього співробітника з бізнес-центру.

Кожна стратегія має власну вартість і різні наслідки для компанії, в першу чергу, в частині прямих і непрямих витрат. Так, наприклад, при виконанні робіт власними силами витрачається більше часу, проте ростуть компетенції власних співробітників, що надалі (при повторному використанні даних компетенцій на наступній фазі) може дати позитивний ефект за термінами. При залученні зовнішніх експертів, терміни можуть бути близькі до ідеальних, проте ростуть прямі витрати. Це може бути продемонстровано моделлю, виконаною на основі мереж Петрі [78].

Представлена на рис. 3.10 мережа Петрі ініціюється заданням кількості робіт, що входять в задане команді завдання, кількістю доступних фахівців і кількістю доступного часу. Це моделюється кінцевим набором «фішок» у відповідних станах, зазначених на моделі значком N.

Переходи «Вибрати ... провідний, лінійного, стажиста» визначають «вартість» виконання роботи спеціалістом з різним ступенем підготовки в одиницях часу. За фактом вибору (спрацьовування відповідного переходу) в станах «Кількість ... ведучих, лінійних, стажерів» відбувається облік кількості задіяних в проекті фахівців з відповідною кваліфікацією. З метою конкретного застосування моделі можна ставити «вартість» виконання роботи різними по кваліфікації фахівцями шляхом зміни кількості «стрілок», що з'єднують стан «Кількість доступного часу» з переходом, відповідним фахівцю, що моделюється.

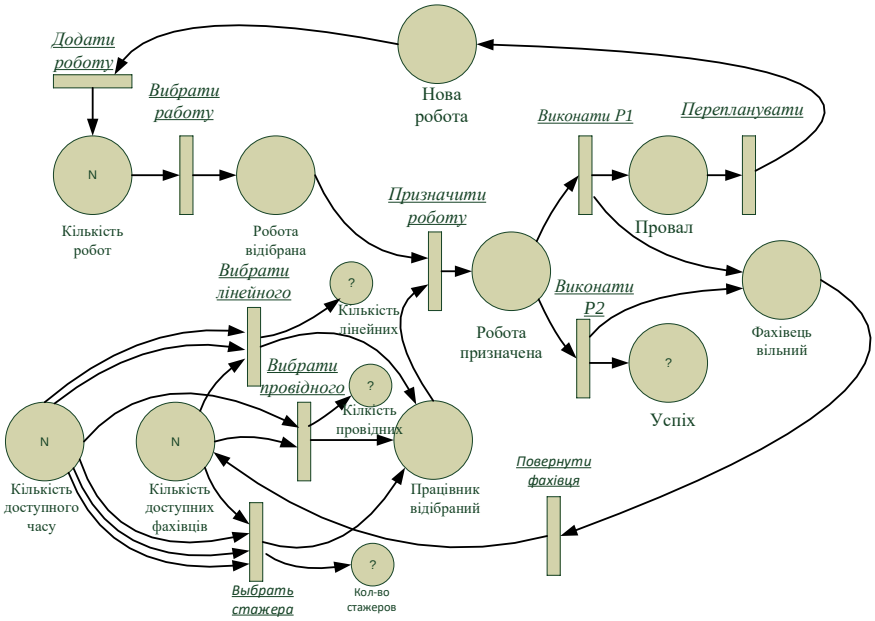


Рисунок 3.10 – Мережа Петрі, яка моделює роботу проектної групи над завданнями

У даному прикладі (рис. 3.10) провідний спеціаліст зменшує доступний для виконання проекту час на одну одиницю, лінійний на дві і стажист на три умовні одиниці проектного часу.

Перехід «Призначити роботу» пов'язує одиничну роботу з конкретним виконавцем, що фіксується станом «Робота призначена».

Дана мережа стохастична. Два переходи «Виконати» спрацьовують з певною ймовірністю успіху (P2) або провалу (P1) у виконанні роботи призначеним спеціалістом. У разі успішного завершення роботи в стан «Успіх» поміщається «фішка», а фахівець повертається в доступний пул фахівців. У разі провалу кількість «фішок» в стані «Кількість робіт» збільшується на одиницю, таким чином моделюється ситуація, коли невиконану роботу необхідно переробити.

Запропонована мережа скінченна.Мережа зупиниться, коли вичерпається доступний час або роботи. При цьому кількість маркерів у стані «Успіх» покаже кількість успішно виконаних робіт. В стані «Кількість

робіт» мережа покаже кількість невиконаних робіт. В стані «Кількість доступного часу» - залишок часу у разі вичерпання всіх невиконаних робіт або нуль, коли вичерпався увесь доступний час.

Слід зауважити, що дана мережа не моделює заданої черговості виконання завдань, її призначення зробити оцінку стратегії, яку обрала команда для виконання роботи у разі відсутності точних даних кваліфікації персоналу та складності виконуваних завдань.

2.5.3.5 Мережа Петрі, що моделює роботу над конкретним завданням

Розглянута вище мережа Петрі моделює час виконання проектного завдання шляхом прямого зв'язку між кваліфікацією спеціаліста та необхідним йому часом для виконання роботи. Для завдань початкового планування і оцінок термінів виконання проекту цього може бути достатньо. Однак для отримання більш точних оцінок часу, що витрачається на виконання роботи, може знадобитися інша модель. Приклад такої моделі наведено на рис. 3.11.

Початкове маркування даної мережі Петрі задається розстановкою маркерів наступним чином:

- маркер в стані «Робота розпочата» - визначає можливість розпочати роботи;
- терміни виконання роботи (в одиницях умовного часу планування) встановлюються кількістю маркерів в стані «Час визначено»;
- всі інші стани залишаються «порожні».

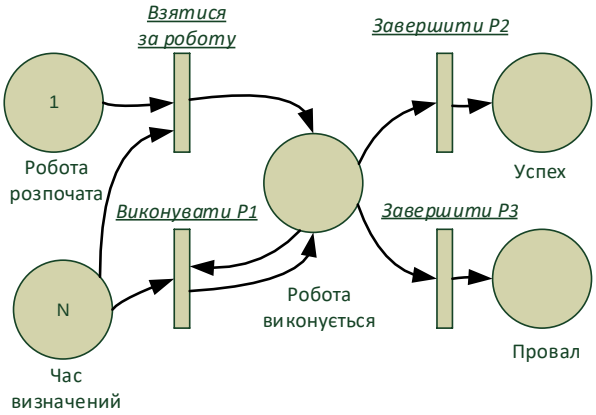


Рисунок 3.11 – Стохастична мережа Петрі, що моделює час, що витрачається при роботі над конкретним завданням

Перехід «До роботи приступити» спрацьовує за умови наявності часу ($N > 0$ в стані «Час визначено») і маркерів в стані «Робота розпочата».

Переходи «Виконувати $P_i(n)$» стохастичні, і спрацьовують з певною ймовірністю P_i. При цьому $P_1(n)$ – ймовірність продовження роботи, $P_2(n)$ – успішного завершення роботи і $P_3(n)$ – провалу виконання роботи на n-номукроці часу.

Стани «Успіх» або «Провал» відображають статус по завершенню роботи. При цьому мережа зупиниться в одному з цих двох станів, коли вичерпається доступний час, оскільки перехід «Виконувати P_1» більше не може бути виконаний і спрацює один з переходів P_2 або P_3.

2.5.4 Інструмент оцінки термінів досяжності цілей IT-проектів

Реалізація моделі виконана в середовищі агентного імітаційного моделювання NetLogo, розробленого в 1999 р Урі Віленським в Центрі Зв'язаного Навчання і Комп'ютерного Моделювання, США [85]. В даний час цей інструмент знаходиться в розвитку і позиціонується як середовище агентного моделювання природних і соціальних явищ. Середовище дозволяє моделювати незалежну поведінку тисяч агентів, що дає можливість досліджувати зв'язок між поведінкою моделі на мікрорівні і на макрорівні. Програмування агентів здійснюється на діалекті мови Logo. Наявність

прикладного інтерфейсу програміста (API) дозволяє інтегрувати моделі NetLogo в зовнішні додатки.

В реалізованій моделі поведінку агента задають правила, засновані на правилах, заданих мережею Петрі, яка описана в параграфі 2.4.3.5 (див. рис. 3.11), в той час як сама модель заснована на мережі Петрі з параграфа 2.4.3.4 (див. рис. 3.10).

Для цілей моделювання були задані 4 типи агентів - студенти, бакалаври, магістри та аспіранти.

```
to make-turtles
set-default-shape turtles "person"
ask turtles [
set heading 90
set size 1
set works-done 1
set workToDo 0
  ]
if-else use-level?
    [
repeat Aspirants [ make-team-member-a ]
repeat Masters [ make-team-member-m ]
repeat Bachelors [ make-team-member-b ]
repeat Students [ make-team-member-s ]
    ]
[ repeat team-size [ make-team-member ] ]
End
```

Кожен агент характеризуються кортежем випадкових чисел на інтервалі $[0,1]$, де кожне число умовно відповідає рівню професійної компетенції в рамках 15 областей знань з програмної інженерії відповідно до стандарту SWEBoKv3 [77] , як вказано в попередньому розділі:

```
turtles-own
[
energy ;; role have energy
works-done ;; total work done by turtle
;; SWEBOK Knowlage Areas
SWRequir   ;; Software Requirements
SWDesign   ;; Software Design
SWConstr   ;; Software Construction
SWTesting  ;; Software Testing
SWMainten  ;; Software Maintenance
```

```
      SWConfig   ;; Software Configuration Management
      SWManage   ;; Software Engineering Management
      SWProcess  ;; Software Engineering Process
      SWModels     ;; Software Engineering Models and
Methods
      SWQuality  ;; Software Quality
      SWPractice ;; Software Engineering Professional
Practice
      SWEconomy  ;; Software Engineering Economics
      CompFound  ;; Computing Foundations
      MathFound  ;; Mathematical Foundations
      EngFound   ;; Engineering Foundations

      ;; SWEBOK Related Disciplines
      CompEng  ;; Computer Engineering
      CompSci  ;; Computer Science
      GManage  ;; General Management
      Mathem   ;; Mathematics
      ProjMana ;; Project Management
      QA       ;; Quality Management
      SysEng   ;; Systems Engineering

      ]
```

На етапі налаштування початкових умов моделі для кожного типу агентів, студенти отримують характеристику для кожної з компетенцій на інтервалі [0, 0.25], бакалаври [0.25, 0.5], магістри та аспіранти на відрізках [0.5, 0.75] і [0.75, 1] відповідно. Закон розподілу для рівня компетенції - рівномірний. Кількість агентів кожного типу може здаватися довільно.

Для моделювання пасивних елементів моделі, в середовищі NetLogo існує поняття «патч» (англ. Patch - клаптик, клапоть) - що метафорично можна представити у вигляді клаптика території, на якій в даний час знаходиться агент. Даний «патч» так само можна наділяти власним набором властивостей, які можуть змінюватися за певними законами, в результаті взаємодії з агентом.

У запропонованій моделі, таким пасивним елементом є одинична робота по проекту, яка характеризується переліком необхідних компетенцій для її виконання, що так само можна представити у вигляді кортежу чисел, заданих в діапазоні [0, 1]. Наприклад, завдання розробки тесту для перевірки

програмного модуля, що робить розрахунок коренів полінома, може бути задане у вигляді наступного кортежу:

$$\langle 0,5;0;0,2;0,75;0,5;0,25;0;0;0;0,5;0;0;0,1;1;0 \rangle .$$

Цеозначає, що агент, який працює над цим завданням повинен мати переважні навички в області математики і тестування, і може не мати ніяких навичок в галузі управління, економіки і з інших, не важливих для цього завдання дисциплін. Такі кортежі не складно розробити для всіх типових завдань, що зустрічаються в IT-проекті, застосовуючи метод парних порівнянь.

Зустрічаючись з таким пасивним елементом, агент приступає до «реалізації завдання», що фактично реалізується зменшенням поточних значень характеристик завдання-патча, на величину власної компетенції за один інтервал модельного часу. Завдання вважається виконаним, коли всі її характеристики обертаються в нуль.

Середовище моделювання дозволяє здійснити моделювання наступних сценаріїв:

1. Коли для кожного агента визначається свій набір завдань.
2. Коли агенти можуть «допомагати» один одному, виконуючи одну задачу спільно.

У загальному вигляді моделювання роботи над проектом безлічі агентів відбувається за методом «голландського сиру», коли кожен агент випадково переміщається по «полю» в пошуках першого-ліпшого завдання, а виявивши його приймається за його рішення. Наближаючись до закінчення проекту починають з'являтися вільні агенти, які можуть допомагати іншим агентам, котрі ще не завершили свої завдання, що цілком відповідає стану справ в реальних проектах.

Вихідні дані по складу і зміст робіт по проекту беруться з веб-порталу, описаного в розд. 2.2, де для цілей планування і управління ходом робіт спроектована база даних, ER-діаграма якої представлена на рис. 4.3. Звідси ж можна взяти інформацію про планову прив'язку виконавця до роботи і відносинам проходження між роботами. У цьому випадку змінюється принцип поведінки агента, він не рухається хаотично в пошуку вільної роботи, але продовжує «рух» тільки по ланцюжку «пов'язаних патчів» - моделюючихпов'язані роботи.

На рис. 4.4 наведено користувацький інтерфейс інструменту з прикладом процесу моделювання. Агенти на рисунку показані у вигляді

антропоморфних фігурок, проектні роботи відображаються у вигляді зелених і сірих квадратів, де сірий колір показує, що по цій роботи вже приступили до виконання. Гістограма в правій стороні вікна показує, скільки робіт було «виконано» кожним агентом.

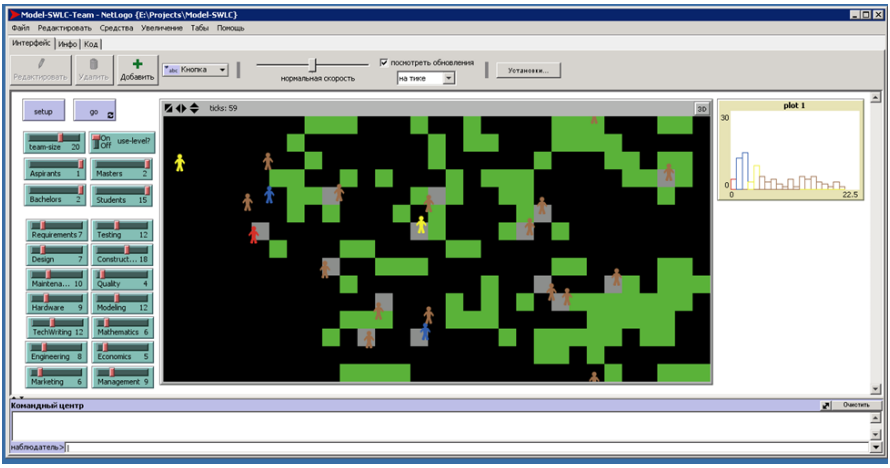


Рисунок 4.4 – Приклад процесу моделювання

Тут слід зазначити, що в близько половини випадків прогону моделі було отримано результат, що найбільше число робіт в проекті виконують агенти, що моделюють безпосередньо заступників керівника, а саме агенти з наступним після найвищого рівнем кваліфікації.

2.6 Оцінка зрілости

2.6.1 Модель SW-CMM і її нинішній розвиток

Модель зрілості здібностей для розробки програмного забезпечення (англ. Capability Maturity Model for Software, SW-CMM) розроблялася з середини 1980-х років Інститутом програмної інженерії університету Карнегі Меллон, м Піттсбург, штат Пенсільванія США (англ. Software Engineering Institute of Carnegie Mellon University, SEI) у відповідь на запит з боку Міністерства оборони США, пов'язаний з проблемою якості розроблюваного на їхнє замовлення ПЗ. Суть реальної проблеми полягала в питаннях управління великими проектами зі створення програмного

забезпечення, які завершувалися з перевищенням термінів і бюджету проектів.

Основною ідеєю було припущення, що якість (програмної) системи в основному визначається якістю виробничих процесів, використовуваних для її створення і супроводу. У своїх дослідженнях SEI виділив ряд критичних напрямків, на які повинні фокусуватися організації з метою поліпшення їх бізнесу. Згідно з їхніми дослідженнями, організації зазвичай фокусуються на людях, процедурах і методах, а також інструментах і обладнанні [86]. Для моделі SW-CMM це призвело до формулювання поняття процесу як засобу, за допомогою якого люди, процедури, методи, обладнання та інструменти інтегровані для отримання бажаного кінцевого результату [18].

У 1987 р з'явився прообраз майбутньої моделі - анкета, яка містила всього 85 процесних і 16 технологічних питань, відповіді на які і відносили оцінювану компанію до одного з п'яти рівнів зрілості. Уже в 1991 р американським інститутом SEI спільно з Mitre Corporation була випущена перша версія моделі зрілості організації, що випускає програмне забезпечення, заснована на ідеях, висловлених Ватсом Хамфрі в його книзі «Managing the Software Process», виданої в США в 1990 р [72].

Для створення цієї моделі був проведений аналіз ключової діяльності, виконуваної при розробці ПЗ, і пов'язаних з нею ризиків. Аналізувалися як позитивний досвід, який дозволив успішно уникнути або пом'якшити той чи інший ризик, так і негативний, який призводить до таких проблем як зриви термінів і перевищення бюджетів. Вся ключова діяльність була згрупована в так звані процесні області. Для кожної ключової діяльності (або мети) модель пропонує ряд дій, які дозволяють зняти або істотно зменшити відповідні проектні ризики.

У 2002 р, на базі інтеграції всіх вищеназваних моделей з'явилася інтегрована модель зрілості здібності (англ. Capability Maturity Model Integration, CMMI) [88]. Відповідно до запропонованого визначення на сайті організації Інституту CMMI, нова концепція є моделлю зрілості поліпшення процесу розвитку продуктів і послуг, складається з набору кращого досвіду (практик), який стосується діяльності в області розвитку і технічного обслуговування, охоплює весь життєвий цикл продукту від концепції до поставки і технічного обслуговування [87]. Появі CMMI сприяло виявлення неоднозначностей в підходах до досягнення більш високих рівнів зрілості для організацій різних розмірів і видів діяльності. З появою нових процесів

розробки ПЗ та «гнучких» (Agile) методів розробки програмного забезпечення [86] модель CMMI продовжує вдосконалюватися. В даний час опублікована версія 1.3 даної моделі [20].

При всій динаміці розвитку оригінальної моделі SW-CMM слід зауважити, що основа моделі - п'ять рівнів зрілості (англ. Maturity level) організацій практично не зазнала істотних змін. Далі наведено їх короткий опис:

1. Початковий. Найпримітивніший статус організації. Організація не має явно усвідомленого розуміння виробничих процесів, що відбуваються в організації. Самі процеси характеризуються хаотичністю, реактивністю, непередбачуваністю і сприймаються як щось аморфне. На даному рівні зрілості оцінки продуктивності виробничих процесів досить приблизні, планування роботи вельми умовне. На такому рівні зрілості якість продукту цілком визначається індивідуальними здібностями членів організації (розробниками і керівниками) і успіх одного проекту не гарантує успіх іншого. При завершенні проекту не фіксуються дані про трудовитрати, розклад і якість.

2. Повторюваний для SW-CMM або керований для CMMI. В деякій мірі відслідковуються виробничі процеси, їх розуміння досягає рівня чорного ящика. Ведуться записи про трудовитрати і плани. Функціональність кожного проекту описується в письмовій формі, контролюються вимоги замовників і проміжні продукти (артефакти проекту). На даному рівні зрілості якість продукту все ще залежить від індивідуальних здібностей членів організації.

3. Встановлений. Є певний документований і встановлений виробничий процес, який вже не залежить від окремих особистостей. Прояснюється розуміння процесу від моделі «чорного ящика» до «білого ящика», а також зв'язків і залежностей між процесами. Вводяться узгоджені професійні стандарти і практики. Такі організації вже в стані досить надійно прогнозувати терміни і витрати на проекти, аналогічно виконаним раніше. Як правило, на цьому рівні організації вже впроваджують у себе систему управління якістю.

4. Керований для SW-CMM або кількісно керований для CMMI. На даному рівні зрілості досягнуті всі цілі попередніх рівнів. Організації можуть точно передбачити терміни і вартість майбутніх робіт. Найголовніша відмінність цього рівня зрілості від попереднього полягає в передбачуваності ефективності процесів і можливості керувати цією

ефективністю. Є база даних накопичених вимірів показників діяльності. Розгорнута в організації система управління якістю ефективно працює. Але все ще недостатньо управляється реакція на зміни в оточенні організації, пов'язані з появою нових технологій і парадигм.

5. Оптимізований. В організації є постійно діюча процедура пошуку і освоєння нових і поліпшених методів, інструментів. На даному рівні зрілості організації мають точні характеристики оцінки ефективності бізнес-процесів, що дозволяє їм постійно і ефективно покращувати бізнес-процеси шляхом розвитку існуючих методів, технік і впровадження нових. Виробничі процеси та система якості сприйнятливі до появи нових технологій і парадигм.

Як було відзначено вище, процес є центральним поняттям в моделях SW-CMM / CMMI. Зрілість організації безпосередньо пов'язана зі станом процесів, і тут виникає важлива концепція інституціоналізації процесів, коли саме поняття процесу вкорінюється в підходах до способу виконання робіт. CMMI-DEV містить 22 області процесу. У додатку Б наведено список процесних областей з короткою розшифровкою назви кожної.

Згідно CMMI-DEV версії 1.3, з метою поліпшення процесів перед організаціями ставлять загальні та спеціальні цілі. Принципова відмінність між загальними та спеціальними цілями лежить в області їх застосування. Коли спеціальна ціль описує унікальну характеристику і спрямована на освоєння конкретної ключової практики, спільні цілі поширюються на всі процеси і організацію в цілому. Слід зазначити, що оцінка можливостей пов'язана з загальними цілями і практиками, тоді як оцінка зрілості більше пов'язана зі спеціальними цілями і практиками.

З метою застосування методів оцінки зрілості за моделлю CMMI були розроблені такі документи: «Вимоги до оцінювання по CMMI версії 1.3» [89] і «Стандартний метод оцінювання по CMMI для поліпшення процесів (SCAMPI) версії 1.3: Опис методу» [131]. Ці документи дають повний опис застосування методів CMMI для оцінки процесів організації. Однак необхідно відзначити, що CMMI, перш за все, концентрується на оцінюванні процесів організацій з розвиненим рівнем зрілості, якими не володіють компанії, створені в рамках академічного IT-підприємництва в університетському середовищі. Крім того, застосування стандартного методу вимагає відповідної підготовки та наявності необхідних ресурсів, наприклад, в [90] описано 5 ролей в експертній команді і 3 допоміжні ролі для виконання такої оцінки. Тому глибоке освоєння університетським

бізнес-центром стандартного методу оцінювання по CMMI v1.3 з єдиною метою його застосування для оцінки процесів новоствореної компанії стає економічно недоцільним з причин різниці масштабів оцінюваних організацій.

2.6.2 Модель інструменту оцінки зрілості IT-компанії на основі теорії нечітких множин

Як було відзначено в підрозділі 2.1.3, завдання бізнес-центру, який опікується стартап/спіноф-компаніями, полягає в доведенні їх до самостійного бізнесу. Що означає створення умов для формування у компанії другого «повторюваного» рівня зрілості і створення передумов до переходу на третій «встановлений» рівень в термінології SW-CMM. Це робить недоцільним розгляд старших рівнів моделі SW-CMM/CMMI і застосування класичних методів з [90], але навпаки ставить завдання побудови такої моделі оцінки зрілості, яка могла б стати основою для інструменту з видачі необхідної інформації наглядовій раді бізнес-центру, що забезпечує прийняття рішення про:

- початок підтримки колективу, який подав заявку на підтримку з боку бізнес-центру університету та формування нової стартап-компанії;
- перехід підтримуваної стартап-компанії в стан спіноф-компанії;
- припинення підтримки стартап/спіноф-компанії з боку бізнес-центру;
- випуск спіноф-компанії в самостійне функціонування.

Крім того, також формує завдання для вищеназваного інструменту по виявленню «вузьких місць» у діяльності підтримуваної компанії, в тому числі з метою розробки рекомендацій щодо усунення проблем і підвищення ефективності компанії.

Втім, виявлення «вузьких місць» в зрілості компанії не є проблемою, так як це, по суті, відсутність або слабке освоєння компанією ключових практик. Інша справа, що причини наявності таких «вузьких місць» можуть бути різні. В першу чергу, це відсутність необхідних компетенцій і практичного досвіду з боку співробітників компанії. У другу, - недостатність ресурсів, так як більш досконалі практики вимагають виділення окремих ролей на їх супровід.

Ретроспективний аналіз змін індикаторів діяльності компанії на основі раніше зібраних даних відіграє роль в оцінці динаміки зміни компанії. Він може бути непрямим індикатором зміни цінностей в команді, цілей проекту та іншої діяльності, складнощів, з якими зіткнулася команда. Крім того, стабільна негативна динаміка зрілості компанії є одним з критеріїв припинення підтримки «опікуємої компанії» з боку університетського бізнес-центру.

У моделі CMMI-DEV v1.3 рівні зрілості IT-компанії аж до 3-го рівня визначає наступна лінгвістична характеристика: «початковий» (I - від англ. Initial), «повторюваний» (R - від англ. Repeatable), «встановлений »(D - від англ. Defined).

Рівень зрілості IT-компанії виводиться на основі суб'єктивних оцінок ступені впровадження в компанії ключових практик, визначених моделлю зрілості. Список таких практик наведено в додатку В. Саме фактичне освоєння даних практик «опікаємою» компанією взято за індикатори стану компанії. Саме на них буде звертати увагу група експертів, яка проводить оцінку процесів в компанії. Освоєння ключових практик оцінюється експертами за допомогою лінгвістичних змінних, що приймають значення: «повністю впроваджено» (FI - від англ. Fully Implemented), «в більшості впроваджено» (LI - від англ. Largely Implemented), «частково впроваджено» (PI - від англ. Partially Implemented), «не впроваджено» (NI - від англ. Not Implemented).

Це дозволяє використовувати ідеї теорії нечітких множин [93, 94] для побудови математичної моделі інструменту оцінки зрілості IT-компаній, створених в рамках академічного підприємництва. У моделі зрілості CMMI ключові практики групуються у спеціальні цілі, далі цілі групуються в області процесу CMMI. Таким чином, для визначення рівня зрілості доцільно використовувати ієрархічні системи нечіткого логічного висновку [92]. Тоді зрілість IT-компанії буде визначатися співвідношенням (4) [96]

$$M = f(f_1(P_1, P_2, \dots, P_7), f_2(P_8, P_9, \dots, P_{18})) \quad (4)$$

де M – зрілістькомпанії, $f_1()$ і $f_2()$ – функції виведення зрілості для 2 і 3 рівнів, $P_1, P_2, \dots, P_{18}$ – засвоєння процесних областей у моделі CMMI-DEV.

У свою чергу, засвоєння кожної процесної області буде визначатися співвідношенням (5)

$$P_i = h_i(g_{i1}(p_{i11}, \dots, p_{i1l}), \dots, g_{in}(p_{in1}, \dots, p_{inm})) \quad (5)$$

де $h_i()$ – функція нечіткого виведення освоєності процесної област із індексом i, $g_{ij}()$ – функція нечіткого виведення досягнення IT-компанії j-ої спеціальної мети для i-оїпроцесноїобласті, p_{ijk} – освоєнняIT-компанією k-ої спеціальної практики моделі CMMI-DEV, що входить в j-у спеціальну мету для i-ої процесної області.

Визначимо оцінки NI, PI, LI, FI як нечіткі числа [95], задані нечіткими множинами N, P, L, F на множині дійсних чисел R. Носіями множин N, P, L, F визначимо інтервали на множині R – $[n_L, n_R]$, $[p_L, p_R]$, $[l_L, l_R]$, $[f_L, f_R]$ відповідно. Для кожного числа NI, PI, LI, FI задамо на R функції приналежності $\mu_i(x)$ такі, що $\forall x \in R \mid \mu_i(x) \in [0,1]$, де $i \in \{N, P, L, F\}$ і

$$\mu_i(x) = \begin{cases} 0 & x < a1 \\ left\left(\dfrac{x-a1}{a2-a1}\right) & a1 \le x < a2 - функція\ лівого\ краю\ \mu_{i\uparrow}(x) \\ 1 & a2 \le x \le a3 \\ right\left(\dfrac{a4-x}{a4-a3}\right) & a3 < x \le a4 - функція\ правого\ краю\ \mu_{i\downarrow}(x) \\ 0 & x > a4 \end{cases} \quad (6)$$

Для таких нечітких чисел визначено поняття метрики відстані між двома числами. В роботі П. Гжегожевського [91] показано, що найбільш підходящим буде розширення поняття евклідової відстані на множину нечітких чисел, що визначається таким виразом:

$$d^2(a,b) = \int_0^1 (A_L(a) - B_L(a))^2 da + \int_0^1 (A_U(a) - B_U(a))^2 da \quad (7)$$

де нечітке число задано через поняття α-перетину, такого, що $\forall a \in (0,1)\ \exists a1 \le x \le a4$ і $\mu_a(x) \ge a$, і $A_L(a) = \mu_{a\uparrow}^{inv}(x)$ – функція зворотна

$\mu_a(x)$ на інтервалі її зростання, а $A_U(a) = \mu_{a\downarrow}^{inv}(x)$ – функція зворотна $\mu_a(x)$ на інтервалі її убування.

Для визначення досяжності ІТ-компанії спеціальної мети і областей процесу CMMI розглянемо функцію $f(x_1, x_2, \dots, x_n) = x_1 \oplus x_2 \oplus, \dots \oplus x_n$, де операція $\oplus$ буде операцією додавання нечітких чисел згідно з формулою (8).

$$a \otimes b = \int_{a1 \otimes b1}^{a2 \otimes b2} \frac{\min(\mu_a(x), \mu_b(y))}{x \otimes y} \tag{8}$$

де $\otimes$ – операція +, –, *, /. Досягнення спеціальної мети G_i освоєння процесів оцінюваної ІТ-компанії опишемо у вигляді лінгвістичної характеристики: повністю досягнута (FR); частково досягнута (PR); недосягнута (NR).

У загальному випадку будемо вважати, що спеціальна мета G_i досягнута (FR), якщо сума відповідей експертів на питання про ступінь впровадження в компанії ключових практик, визначених моделлю зрілості CMMI-DEV v1.3, вираженої у вигляді нечіткого числа, рівного $f(x_1, x_2, \dots x_n) = x_1 \oplus \dots \oplus x_n$, «ближче» за П. Гжегожевським до нечіткого числа $FI * n = FI_1 \oplus \dots \oplus FI_n$.

Спеціальна мета G_i досягнута частково (PR), якщо $f(x_1, x_2, \dots x_n) = x_1 \oplus \dots \oplus x_n$ «ближче» до нечіткого числа $Mean = (LI * n \oplus PI * n) / 2$.

Спеціальна мета G_i не досягнута (NR), якщо $f(x_1, x_2, \dots x_n) = x_1 \oplus \dots \oplus x_n$ «ближче» до нечіткого числа $NI * n = NI_1 \oplus \dots \oplus NI_n$.

Освоєння області процесу виражається у вигляді суми оцінок G_i виду NR, PR, FR, отриманих на попередньому кроці для спеціальних цілей, що входять до конкретної області процесу CMMI. Освоєння областей процесу позначимо лінгвістичними змінними NF, FF і PF, що буде відповідати поняттям «область не освоєна», «освоєна частково» і «повністю освоєна». Вони будуть обчислюватися в загальному вигляді за такими правилами:

процесна область $F_j = NF$, якщо $\sum_{i=1}^{n} G_i$ «ближче» до $\sum_{1}^{n} NR$;

процесна область $F_j = PF$, якщо $\sum_{i=1}^{n} G_i$ «ближче» до $\sum_{1}^{n} PR$;

процесна область $F_j = FF$, якщо $\sum_{i=1}^{n} G_i$ «ближче» до $\sum_{1}^{n} FR$.

Далі слід визначити освоєння ІТ-компанією рівнів зрілості. Для цього треба визначити функції $f()$, $f_1()$ і $f_2()$ з виразу (4). Але якщо для функцій $f_1()$ і $f_2()$ можна застосувати аналогічні міркування, висловивши їх результат через суму освоєних процесних областей, то для функції $f()$ такі міркування вже не підійдуть. У моделі CMMI вважається, що компанія досягає 3-го «повторюваного» рівня (R), якщо вона досягла 2-го «встановленого» рівня (D) і в більшості освоїла процесні області, специфічні для 3-го рівня зрілості. Якщо значення функцій $f_1()$ і $f_2()$ визначаються терм-безліччю {NM, PM, FM}={«незрілий», «частково зрілий», «повністю зрілий»}, тоді можна сформувати такий набір нечітких правил:

ЯКЩО $(f_1(P_1, \ldots, P_7) = NM)$, ТОДІ $M = I$.

ЯКЩО $(f_1(P_1, \ldots, P_7) = PM)$, ТОДІ $M = D$.

ЯКЩО $(f_1(P_1, \ldots, P_7) = FM)$, ТОДІ $M = D$.

ЯКЩО $(f_1(P_1, \ldots, P_7) = PM)$ І $(f_2(P_8, \ldots, P_{18}) = NM)$, ТОДІ $M = D$.

ЯКЩО $(f_1(P_1, \ldots, P_7) = PM)$ І $(f_2(P_8, \ldots, P_{18}) = PM)$, ТОДІ $M = R$.

ЯКЩО $(f_1(P_1, \ldots, P_7) = FM)$ І $(f_2(P_8, \ldots, P_{18}) = PM)$, ТОДІ $M = R$.

ЯКЩО $(f_1(P_1, \ldots, P_7) = FM)$ І $(f_2(P_8, \ldots, P_{18}) = FM)$, ТОДІ $M = R$.

Отримані правила дозволяють побудувати програмний інструмент оцінки зрілості ІТ-компанії.

2.6.3 Калькулятор оцінки зрілості ІТ-компанії

Теоретичні основи інструментального засобу, що оцінює рівень зрілості ІТ-компанії за сукупністю індикаторів, які базуються на п'ятирівневій моделі оцінки зрілості CMMI, описані вище. Основне призначення такого калькулятора полягає у виявленні «вузьких місць» у діяльності підтримуваного стартапу з метою вироблення рекомендацій щодо усунення проблем і підвищення ефективності ІТ-компанії, що розвивається. Результати оцінки даного інструментального засобу використовуються для

прийняття рішення про продовження підтримки компанії, переведення її зі стану стартап в спіноф і далі випуск її з-під опіки бізнес-центру.

Калькулятор структурно складається з наступних модулів:

- «опитувальника», що формує базу експертних оцінок індикаторів діяльності компанії;
- модуля оцінки рівня зрілості компанії, що здійснює розрахунок на основі апарату нечіткого висновку;
- модуля виявлення «вузьких місць» в компанії, який заснований на зіставленні актуальних оцінок освоєння процесних областей належному рівню зрілості;
- модуля ретроспективного аналізу змін індикаторів діяльності компанії на основі раніше зібраних даних.

2.6.4 Приклади застосування

Створений інструмент був випробуваний для оцінки зрілості IT-стартапу, створеного студентами Славутицької філії НТУУ «КПІ», а також на місцевій, вже існуючій 4 роки, малій IT-компанії, яка успішно працює на ринку аутсорсингу програмного забезпечення. Оцінка проводилася як самостійно персоналом компанії, так і експертною групою, сформованою з фахівців в галузі управління проектами зі значним практичним досвідом роботи. Результати такої оцінки для успішно існуючої компанії наведені на рис. 4.5.

На представленій діаграмі наведені результати оцінки за стандартним алгоритмом, описаним в документах моделі CMMI [130, 131], а також результати, отримані за допомогою інструменту, заснованого на нечіткому висновку. Тут слід зазначити, що отримані лінгвістичні оцінки зрілості обома способами збігаються. Однак відсутність можливості по дефаззифікації отриманого результату для стандартного способу не дозволяє відстежувати динаміку змін в зрілості компанії в рамках одного рівня оцінки, що робить запропонований в роботі метод оцінки зрілості більш привабливим щодо стандартного. Крім того, метод оцінки зрілості за допомогою нечіткого висновку зблизив підсумкові результати для експертної оцінки і самооцінки компанії, в той час як фінальний результат, отриманий стандартним способом в рамках самооцінки компанії, виявився завищеним.

На рис. 4.6 наводяться порівняльні оцінки зрілості для IT-стартапу і самостійної IT-компанії.

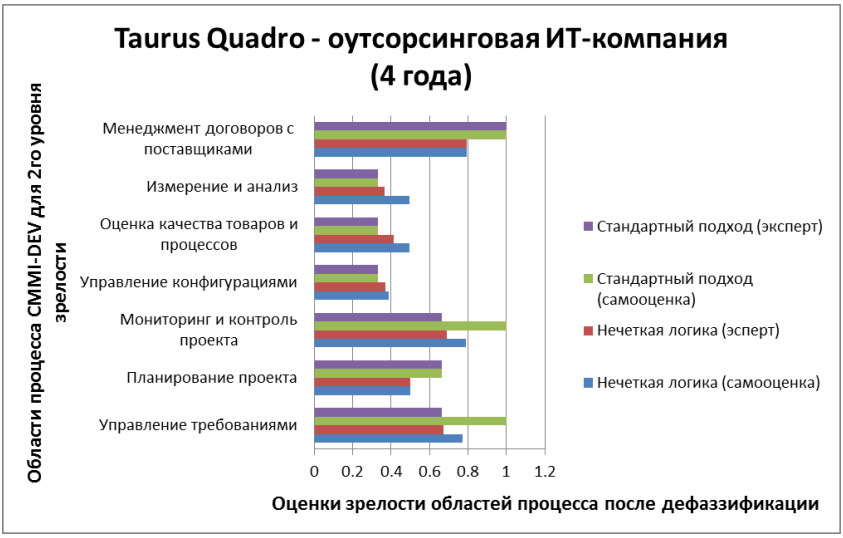


Рисунок 4.5 – Результати оцінки зрілості місцевої IT-компанії з 4-річним стажем роботи

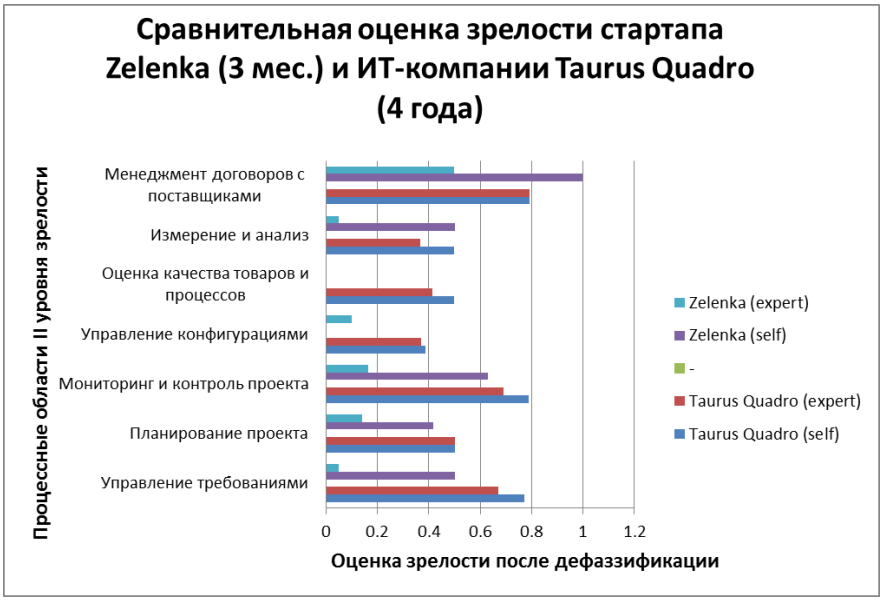


Рисунок 4.6 – Порівняння зрілості малої ІТ-компанії з 4-річним стажем роботи і ІТ-стартапу

Інформація на даній діаграмі показує, що ІТ-стартапам властиво значно завищувати власну самооцінку, в той час як реальна організаційна зрілість ІТ-стартапів є вкрай низькою. Тому оцінка зрілості ІТ-стартапу повинна проводитися експертами університетського бізнес-центру. Одну з корисних можливостей, що дає створений інструментальний засіб оцінки зрілості, можна продемонструвати на прикладі оцінки стартапа DDPRO, описаного в п.4.2. Це ретроспективний аналіз змін індикаторів діяльності компанії на основі раніше зібраних даних. На рис. 4.7 наводяться порівняльні оцінки зрілості для ІТ-стартапу DDPRO, отримані на етапах:

(1) оцінки доцільності підтримки ідеї та колективу майбутнього ІТ-стартапу;

(2) прийняття рішення про переведення колективу в юридичну особу ТОВ «ДДПРО»;

(3) оцінки компанії на етапі випуску першого прототипу системи.

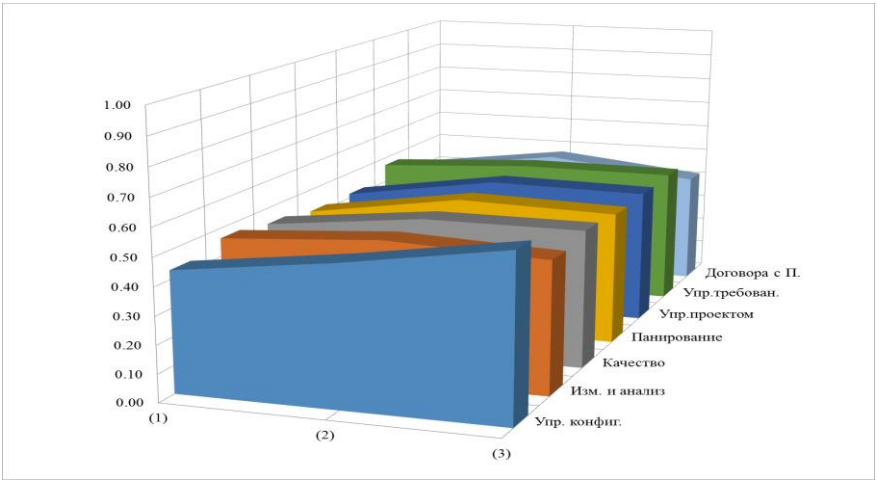


Рисунок 4.7 – Динаміка зрілості IT-стартапу DDPRO

Початковий високий рівень зрілості колективу DDPRO пояснюється наявністю досвіду роботи ключових співробітників в іноземних компаніях до організації стартапу. У той же час деяке зниження показників зрілості по ряду організаційних показників на етапі випуску першого прототипу системи пояснюється зміщенням у команди фокусу уваги на проектування продукту.

РОЗДІЛ 3. МОДЕЛІ ТА ІНСТРУМЕНТАЛЬНІ ЗАСОБИ В СИСТЕМАХ АВТОМАТИЗОВАНОГО НАВЧАННЯ

3.1 Історія розвитку автоматизації в освіті

Перші експерименти по застосуванню комп'ютерів в освіті відносяться до кінця 50-х років минулого століття. Незважаючи на те, що технічна база ЕОМ та програмне забезпечення того часу явно не відповідали успішному вирішенню поставленої проблеми в цілому, дослідження в цій області почалися у всіх розвинених країнах. Виділимо найбільш значущі етапи розвитку робіт у цій галузі і простежимо за зміною цілей і завдань, які ставили перед собою дослідники і розробники.

Перший етап дослідження можливостей створення навчальних систем припадає на 50-60-ті роки двадцятого сторіччя. Ще в 1954 році професор Б. Ф. Скіннер висунув ідею, що отримала назву програмованого навчання [97]. Вона полягала в спробі підвищити ефективність управління навчальним процесом шляхом побудови його в повній відповідності з психологічними знаннями про нього, що фактично означає впровадження кібернетики в практику навчання. Цей напрям почав активно розвиватися в США, а потім і в інших країнах. І вже тоді однією з основних ознак програмованого навчання вважалася автоматизація процесу навчання.

Автоматизація програмованого навчання почалася з використання навчальних і контролюючих пристроїв різного типу. Вони досить широко застосовувалися в 60-70-ті роки [98-100], хоча, за обмежених можливостей, не забезпечували достатньої ефективності та адекватності результатів контролю реальному рівню знань студента. Фактично застосування таких пристроїв як в нашій країні [101], так і за кордоном [102] не вийшло за рамки навчання різним навичкам, а також найпростіших методів контролю, в основному вибіркового типу.

В цей же час почали розвиватися ідеї штучного інтелекту. Були розроблені основні моделі подання знань, з'явилися перші системи, що використовують методи штучного інтелекту.

У процесі створення перших прототипів систем автоматизованого навчання (САН) стало очевидним, наскільки складними є завдання представлення предметних знань, організації зворотного зв'язку зі студентом (у тому числі, повноцінного діалогу, для якого явно не вистачало лінгвістичних знань).

В 60-ті роки було розроблено велику кількість спеціалізованих пакетів програм, орієнтованих на створення і супровід прикладних навчальних програм — автоматизованих навчальних курсів (АНК) на базі ЕОМ третього покоління. Деякими з найвідоміших у нашій країні проектів використання обчислювальної техніки та засобів комунікації у навчанні є проект PLATO, в найбільш розвиненій версії — PLATO-IV, а також вітчизняні автоматизовані навчальні системи АНС-ВНЗ, АНС-СПОК, АСТРА, САДКО [103] та інші.

Ці та інші навчальні системи були системами селективного типу. У таких системах визначенням методики навчання займався педагог, а реалізація та оцінка результатів проводилася засобами САН.

Крім систем селективного типу були створені навчальні системи, в яких діалог зі студентом не програмувався, а формувався за кількома алгоритмами відповідно до набору операцій і фактів, закладених в систему. Подібні навчальні системи призначалися для деяких специфічних предметних областей, які з тих чи інших причин опинилися виключно підходящими для такого типу програмування. В якості прикладів можна привести систему Ліклайдера для навчання аналітичній геометрії [104] і систему Бітена і Лейна для навчання вимові слів іноземної мови [105].

Наступний, другий етап у розвитку автоматизованого навчання розпочався з початку 70-х до середини 80-х. До цього часу ідея створення інтелектуальних систем фактично зазнала тимчасового фіаско. Системами автоматизованого навчання почали називати будь-які програми, призначені для інформаційної або функціональної підтримки процесу навчання: тести, електронні підручники, лабораторні практикуми і т. ін.

Незважаючи на ослаблення вимог до навчальних систем, тривали дослідження можливості використання ідей і методів представлення знань, розроблених до того часу в області штучного інтелекту при створенні САН. Але якщо для подання знань про предметні області ці розробки підходили в значній мірі, то для вирішення двох інших завдань — управління навчальним процесом і контролю знань — були потрібні більш складні методи і засоби. Саме ці проблеми перебували в полі зору розробників навчальних систем в кінці даного періоду і все ще є предметом сучасних досліджень в області створення навчальних програм.

На початку цього періоду основні зусилля теоретиків автоматизованого навчання були спрямовані на пошук і перевірку більш глибоких моделей навчання на основі когнітивної психології. Як наслідок

цих робіт стали з'являтися експериментальні навчальні системи продукційного типу, де методика навчання обиралася не педагогом, а визначалася алгоритмом функціонування системи і генерувалася в залежності від цілей навчання і поточної ситуації. При цьому передбачалося, що в навчальній системі представлені знання про те, чому навчати, як навчати і знання про студента.

Третій етап — друга половина 80-х і 90-і роки. Цей період характеризується двома основними тенденціями. З одного боку, широке поширення персональних комп'ютерів (ПК) і розвиток обчислювальних мереж орієнтує навчальні системи на роботу в мережі з використанням загальноприйнятих стандартів подання та передачі даних. З іншого боку, швидко розвиваючі апаратні можливості призвели до того, що одним з основних напрямів розвитку навчальних систем стало застосування в них нових комп'ютерних технологій, а саме гіпертексту і мультимедіа [106].

На сьогоднішній день системи автоматизованого навчання (САН) — це інструментальний комплекс, що включає методичне, математичне та програмне забезпечення, робота якого спрямована на забезпечення автоматизованої підтримки процесу навчання студентів за участю викладача (тьютора), підготовки навчальних матеріалів, управління процесом навчання і контролю його результатів.

3.2 Класифікація програмних навчальних засобів

Аналізуючи існуючі зарубіжні і вітчизняні адаптивні та інтелектуальні системи навчання (КАДИЗ, КОБРА, АНСМІКРО, АНА!, WebCOBALT, MLTutor та ін.), було встановлено такі особливості:

- врахування моделі студента в процесі навчання, адаптація навчання під його потреби та знання;
- можливість налаштування тривалості та форм навчання;
- контроль знань студента забезпечується проведенням тестового контролю знань.

Слід зауважити, що за способом реалізації навчального процесу в начальних системах розрізняють дві принципово різних моделі навчання:

- лінійна;
- адаптивна.

Лінійна модель характеризується чітко визначеною послідовністю проходження навчального матеріалу (блоків, модулів), яка попередньо

обумовлена розробником системи (адміністратором, викладачем, тощо). Незважаючи на те, що дані послідовності, як правило, розрізняються на рівні категорій (школярі, студенти), груп (за спеціальностями), і навіть окремих студентів (за рівнем підготовки), тим не менш процес навчання полягає в переходах від одного навчального блоку (модуля) до другого, згідно раніше заданої траєкторії.

В якості переваг даної моделі можна виділити високий рівень стандартизації (підготовка універсальних програм) та простоту побудови. Але, основним недоліком лінійної моделі є відсутність можливості врахування індивідуальних особливостей студента, як психофізичних, так і розумових. Саме спроби усунення цього недоліку призвели до побудови моделі навчання, спрямованої на адаптацію до конкретного студента.

Суть адаптивної моделі навчання полягає в наступному:

1. Розробка навчального матеріалу виконується з урахуванням спеціалізації груп студентів (елемент лінійної моделі навчання).

2. Для кожного модуля, який є важким для сприйняття чи містить значну кількість інформації, розробляється дві його версії: спрощено-поверхнева та деталізована.

3. При переході до вмісту модуля студент проходить попередній тест, який дозволяє визначити рівень його знань з нової теми (деяка частина студентів до початку навчання вже володіє необхідними знаннями, навиками, вміннями).

4. За результатами тесту студент:

   а) перенаправляється до наступного модуля (пропускає тему, якщо тест виявляє достатній рівень знань з теми);

   б) переходить до спрощеної версії модуля (якщо рівень підготовки студента виявився вищим за середній; одночасно існує можливість отримати більш детальні дані з модуля, що включає деталізовану інформацію);

   в) направляється до деталізованого модуля, якщо його знання виявлялися недостатніми.

5. Окрім вхідного тестування для навчальних модулів, що мають різний рівень викладення матеріалу, для всіх без виключення модулів мають бути реалізовані процедури підсумкового тестування, які дозволяють повернути студента до початку розділу/теми при недостатньому рівні знань, отриманих в ході вивчення модуля [107].

Враховуючи той факт, що під САН частіше за все розуміють програмні засоби навчання, розглянемо різні класифікації цих засобів.

1. За принципами взаємодії програмних навчальних засобів і студента ці програмні засоби можна розділити на два великі класи [108]:

- навчальні середовища;
- навчальні програми.

Жорсткої межі між навчальними середовищами та навчальними програмами немає. Дійсно, системи, що забезпечують демонстрацію навчального матеріалу у своєму розвитку «йдуть» в напрямку навчальних середовищ. А навчальні середовища-тренажери в певній галузі в кінцевому рахунку наближаються до навчальних програм-тренажерів. Єдина відмінність, яка залишається між навчальними системами цих класів — відсутність контролю фіскального типу в навчальних середовищах і наявність його в навчальних програмах.

При роботі в навчальному середовищі передбачається, що студент має певну мету, а система асистує йому в досягненні цієї мети. У системах даного класу відсутній (принаймні явно) етап контролю студента з боку системи, оскільки система не знає мети, з якою студент звернувся до неї.

2. Оскільки деякі науковці і на сьогодні ототожнюють системи автоматизованого навчання з розподіленою системою, в якій є клас контролюючих програм і клас з комп'ютерними засобами, що допомагають в процесі навчання, то доречно розглянути їх класифікацію. Наприклад, Кривошеєв А. О. [109] пропонує ділити комп'ютерні навчальні засоби на такі класи:

- комп'ютерні (або електронні) підручники — забезпечують можливість самостійно освоїти навчальний курс або його розділ;
- предметно-орієнтовані середовища — це навчальні пакети програм, що дозволяють оперувати з об'єктами певного класу;
- лабораторні практикуми, що дозволяють автоматизувати виконання лабораторних робіт з різних дисциплін;
- тренажери призначені для відпрацювання і закріплення технічних навичок вирішення завдань;
- контролюючі програми, призначені для перевірки (оцінки) якості знань студентів;
- інструментальні системи — це програмні комплекси, призначені для створення різних програм навчального призначення;

- довідники, бази даних навчального призначення, що забезпечують зберігання і надання студенту різноманітної навчальної інформації довідкового характеру.

3. Савельєв А. Я. виділяє наступні два класи навчальних систем [110]:

- автоматизовані навчальні системи (АНС) — їх ядром є так звані авторські системи, що дозволяють викладачеві вводити свій навчальний матеріал в базу даних системи і програмувати алгоритми вивчення за допомогою спеціальних засобів;
- окремі програми, пакети програм, елементи автоматичних систем (АСУ, САПР, АСНІ, АСУВ, тощо), призначені для автоматизації трудомістких розрахунків, оптимізації, дослідження властивостей об'єктів і процесів на математичних моделях і т. ін. Особливе місце серед таких систем займають експертно-навчальні системи (ЕНС) [111,112], які зазвичай представляють собою промислові експертні системи (ЕС), адаптовані до цілей навчання.

4. Соловов А. В., автор однієї з відомих навчальних систем (КАДИЗ — Комплекс Автоматизованих Дидактичних Засобів [113]) пропонує умовно ділити навчальні системи на дві множини:

- комп'ютерні навчальні програми, під якими розуміється будь-який програмний засіб, спеціально розроблений або адаптований для застосування в навчанні;
- інструментальні системи, призначені для розробки комп'ютерних навчальних програм і створення навчальних курсів.

5. Останнім часом робляться спроби систематизації навчальних комп'ютерних засобів. Наприклад, Б. Х. Кривицький [114] пропонує проводити їх класифікацію («розмежування») за двома ознаками:

а) вид навчальної діяльності:

засоби для групової роботи;

засоби для самостійної роботи;

б) педагогічне (або дидактичне) призначення:

інформаційні, що забезпечують прямий канал передачі;

контролюючі, що забезпечують зворотний канал передачі;

навчальні, що забезпечують замкнутий цикл управління.

6. З точки зору управління навчальним процесом всі навчальні системи можна розділити на два класи [115]:

навчальні системи, в яких управління процесом навчання покладено на користувача. Містять викладення навчальної дисципліни або її розділу на машинному носії у текстовому і графічному форматах відповідно до її логіки. Навчальні системи даного класу відрізняються між собою функціональністю, властивостями, способами їх реалізації і поділяються на наступні підкласи:

- електронний підручник чи методичний посібник з послідовною структурою — можна розглядати як електронну копію традиційного друкованого підручника чи посібника, тобто структура представлення матеріалу на машинному носії є послідовною;
- електронний підручник чи методичний посібник з гіпертекстовою структурою — представлення навчальної дисципліни на машинному носії має гіпертекстову структуру;
- повнотекстова база даних — надає можливість звернення за посиланнями з авторського викладення навчальної дисципліни до оригінальних текстів інших авторів, при цьому як авторський текст, так і тексти інших авторів можуть мати гіпертекстову структуру подання на машинному носії;
- електронна бібліотека — система, що керує комплексом електронних навчально-методичних матеріалів різного класу з різних навчальних дисциплін, дозволяє навчати виконувати пошук інформації (пошук за ключовими словами, пошук по предметній області), де простір пошуку має допускати розширення, причому необхідна організація взаємодії з відповідною бібліографічної системою;
- мультимедійний електронний підручник чи методичний посібник — виклад навчальної дисципліни повністю виконано або доповнено викладенням в аудіо, відео форматах, за рахунок цього дана система дозволяє студенту спостерігати динаміку досліджуваних явищ і змінювати параметри цієї динаміки. Така система може володіти всіма або декількома властивостями повнотекстових баз даних;
- електронний підручник чи методичний посібник із засобами рубіжного контролю — після кожного розділу навчальної дисципліни системою формується оцінка, яка є основою для самоконтролю студента. Ця система може володіти всіма або декількома властивостями мультимедійних систем;
- навчальні системи, які самостійно керують навчальним процесом. Містять викладення навчальної дисципліни або її розділу на

машинному носії у текстовому, графічному, аудіо, відео форматах відповідно до її логіки. В кінці кожного фрагменту викладення навчальної дисципліни в даних системах студенту надаються перевірочні завдання. На відміну від систем першого класу, в даних системах відповіді і дії студента впливають на подальший хід процесу навчання. Ступінь управління навчальним процесом безпосередньо залежить від ступеня адаптації системи під конкретного студента, тому навчальні системи даного класу поділяються на підкласи за ступенем їх адаптивності та способами реалізації адаптації:

a. САН з лінійною моделлю навчання — структура подання матеріалу на машинному носії є послідовною. Залежно від результатів перевірки студенту надається черговий (наступний) фрагмент навчального матеріалу, або він повертається до додаткового вивчення попереднього фрагменту;

b. САН з розгалуженою моделлю навчання — для кожного фрагменту навчальної дисципліни в системі задано декілька варіантів викладення матеріалу, що розрізняються за ступенем деталізації, глибиною викладу, а також кілька варіантів пропонованих в кінці кожного фрагменту перевірочних завдань з різними рівнями складності. Дана система адаптується за глибиною, ступенем деталізації викладу досліджуваного матеріалу і складністю перевірочних завдань, що дозволяє їй формувати індивідуальну траєкторію навчання;

c. САН з адаптацією за формою викладення — студент має можливість вибирати форму викладення навчальної дисципліни: переважно або текстова, або графічна, або аудіо, або відео форма;

d. САН з адаптацією за логікою викладення — контроль знань студента здійснюється на основі зіставлення моделей про предметну область викладача (еталонної моделі) і студента;

e. Мультиагентна САН з адаптацією до об’єкту та цілей навчання — управління навчальним процесом здійснюється колективом агентів, кожен з яких окремо має всі властивості навчальних систем попередніх підкласів. Колектив агентів складається щоразу під конкретного студента, під його цілі навчання. [116].

Узагальнюючи вище наведені класифікації, можна зробити наступні висновки:

- оптимального варіанту класифікації САН немає, частіше за все один клас перетинається з іншим;

- автоматизація в цих системах найчастіше полягає у підтримці традиційної форми навчання комп'ютерними засобами;
- поява адаптивних систем навчання коригує процес отримання знань під конкретного студента.

3.3 Проектування формалізованого представлення предметної області, виходячи з її текстово-графічного опису

На сьогодні практично всі САН для ознайомлення студента з матеріалами навчального курсу використовують структуровану інформацію у вигляді розділів текстово-графічних матеріалів, підрозділів, пунктів і т. ін. (конспектів лекцій, методичних вказівок до лабораторних робіт тощо). Така структуризація предметної області не дає можливості визначити ступінь охоплення контрольними тестами всіх матеріалів навчального курсу, а отже процес контролю знань студента не можна вважати повним та ефективним. В процесі навчання при наданні студенту інформації для ознайомлення, як правило, не враховується ступінь його попередньої підготовки.

Формалізація предметної області навчального курсу формує новий погляд на його структуру та може надати можливість виправити вказані вище недоліки. Це впливає на підготовку контенту САН, процес навчання та проведення контролю знань студентів.

Знання в системах навчання розглядаються як знання про предмет, і визначається як впевнене розуміння предмета, вміння самостійно звертатися і розбиратися в ньому, а також використовувати його для досягнення намічених цілей [117].

Відповідно до поглядів Ю. І. Кликова, знання — це сукупність даних, фактів і правил виведення про світ, які включають в себе інформацію про властивості об'єктів, закономірності процесів, явищ, а також правил, які використовуються цією інформацією для прийняття рішень [118].

Текстово-графічне представлення знань, виражене у вигляді сукупності речень на природній мові, є найбільше поширеним. Але існує ряд більш спеціалізованих та структурованих штучних мов, що утворюють різні розділи наукового сленгу, які базуються на використанні різних моделей формалізації знань. Вони дозволяють більш точно і компактно описати явища та процеси реального світу.

В даний час отримали найбільшу популярність наступні моделі формалізованого представлення знань:

Логічні моделі. Основна ідея підходу при побудові логічних моделей представлення знань — вся інформація, необхідна для вирішення прикладних завдань, розглядається як сукупність фактів і тверджень, які представляються як формули в деякій логіці. Знання відображаються сукупністю таких формул, а отримання нових знань зводиться до реалізації процедур логічного виводу. В основі логічних моделей представлення знань лежить поняття формальної теорії, що задається кортежем: $S=<B,F,A,R>$, де B — зліченна множина базових символів (алфавіт), F — множина, яку називають формулами, A — виділена підмножина апріорі істинних формул (аксіом), R — кінцева множина відношень між формулами, яку прийнято називати правилами виведення [119].

Семантичні мережі. Модель представлення знань за допомогою семантичних мереж складається з вершин (вузлів), що відповідають об'єктам, концепціям або подіям, і дуг, що їх зв'язують, які описують відношення між розглянутими об'єктами. Дуги можуть бути визначені різними методами. Зазвичай для представлення ієрархії використовуються дуги типу «IS–A» (відношення «є») і «HAS–PART» (відношення «має–частина»). Вони також встановлюють ієрархію спадкування в мережі, тобто елементи нижчого рівня в мережі можуть успадковувати властивості елементів більш високого рівня, що економить пам'ять, оскільки інформацію про успадковані властивості не потрібно повторювати в кожному вузлі мережі [120].

Фрейми. Модель представлення знань за допомогою фреймів запропонована Марвіном Мінським, який описує їх наступним чином [121]: «Фрейм — це структура даних, що представляє стереотипну ситуацію начебто знаходження всередині деякого роду житлової кімнати або збирання на вечірку з приводу народження дитини». До кожного фрейму приєднуються кілька видів інформації. Частина цієї інформації про те, як використовувати фрейм, частина про те, чого можна очікувати далі, частина про те, що слід робити, якщо ці очікування не підтвердяться. Фреймова модель за своєю організації багато в чому схожа на семантичну мережу. Вона є мережею вузлів і відношень, організованих ієрархічно: верхні вузли представляють загальні поняття, а підлеглі їм вузли — окремі випадки цих понять. У системі, заснованій на фреймах, поняття в кожному вузлі визначається набором атрибутів-слотів і значеннями цих атрибутів. Кожен слот може бути пов'язаний зі спеціальними процедурами, які виконуються,

коли інформація у слотах (значення атрибутів) змінюється. З кожним слотом можна пов'язати будь-яке число процедур.

Продукційні системи. У цій моделі знання представляються у вигляді сукупності правил типу «якщо–то» [122,123]. Системи обробки знань, що використовують таке подання, отримали назву продукційних систем. До складу систем продукційного типу належать база правил (знань), робоча пам'ять і інтерпретатор правил (вирішувач), який реалізує певний механізм логічного висновку. Факти представлені у вигляді пари: атрибут–значення, а у випадках, коли є кілька об'єктів з ідентичними атрибутами, — у вигляді трійки: об'єкт–атрибут–значення. Операційні знання представляються правилами виду умова–дія або умова–мета. Правило інтерпретується таким чином: для досягнення мети правила необхідно досягти цілей умови. При вирішенні завдання керуюча програма вибирає мету і намагається досягти її.

У більшості форм представлення знань центральною ідеєю є фіксація множини концептів і відношень між ними, які добре поєднуються зі структурними моделями.

Структурні моделі, як правило, носять статичний характер, фактор часу в них не відіграє суттєвої ролі. Для опису таких моделей розроблено безліч спеціалізованих формалізмів завдання динаміки поведінки систем, до яких відносяться системи диференціальних рівнянь, кінцеві автомати [124], марківські і напівмарківські процеси [125], агрегативні моделі [126], мережі Петрі [127], логіко-динамічні моделі [128], та ін.. Всі ці засоби опису динаміки в явній або неявній формі мають справу з множиною станів системи і переходами між станами. Вони використовуються як для опису фактів, так і знань, які задають класи. Моделі, що відображають динаміку поведінки, часто називають динамічними моделями.

Динамічні моделі характеризуються наявністю множини станів і можливих переходів між ними в часі. Виходячи з цього, в перелік завдань, що вирішуються за допомогою динамічних моделей, входять завдання прогнозу або досяжності станів, завдання допустимості траєкторій поведінки, завдання визначення моментів обриву траєкторії і т. ін.

Якісне навчання має здійснюватися на актуальних моделях. Звідси безперечна важливість функції підтримки знань в актуальному стані в САН. Її складність сильно залежить від форми подання знань. Найпростіше вона реалізується при природно-мовній текстово-графічній формі шляхом редагування старого тексту і введення нового.

При формалізованої формі зберігання знань є два способи актуалізації:

- пряма робота інженера зі знань з формалізованим представленням шляхом редагування старих формалізованих представлень і додавання нових;
- навчання системи.

Перший спосіб підходить практично при всіх формах формалізованого представлення знань, другий — практично тільки при представленні знань у вигляді нейронної мережі.

Розв'язання задач з використанням моделей фактично є процесом трансформації первинних знань і фактів у вторинні (похідні) знання. Трансформація в навчальній системі може підтримуватися або створенням індивідуальних процедур обчислення похідних показників, або використанням для цих цілей стандартизованих процедур. Останні, як правило, прив'язані до методу представлення знань, точніше, до стандартних процедур, характерних для того чи іншого формалізму.

Розглянемо стандартні процедури, які використовуються для отримання вторинних знань для деяких формалізмів.

Формальні граматики — процедури граматичного розбору і процедури генерації речень мови з використанням граматики.

Логічні обчислення — процедури логічного виводу теорем, наприклад, метод резолюцій, або алгоритми формульних перетворень.

Мережі Петрі — алгоритми перевірки досяжності розмітки мережі з початкової.

Нейронні мережі — процедури обчислення вихідних значень мережі за заданими вхідними.

Кінцеві автомати — процедури перевірки досяжності заданого стану з початкового.

ER-моделі — процедури перевірки коректності моделі або процедури оцінки показників якості моделі [129].

Розглянуті представлення знань можуть використовуватися при вивченні предметної області, як окремо, так і в сукупності, тобто комбінування і використання моделей з різними формалізмами.

Виходячи з вище сказаного, можна зробити висновок, що єдиного ефективного способу формалізованого представлення знань не існує, а вибір способу формалізації залежить від задач, що вирішуються в системі.

Аналізуючи мови представлення знань, раціонально звернути увагу на багатомовні засоби для формалізованого представлення знань, до яких відноситься Unified Modeling Language (UML).

Як відомо з [130], UML являє собою багатомовну систему з можливостями задання структурних моделей (діаграми класів), сценаріїв взаємодії елементів моделей (діаграми послідовностей і діаграми взаємодії об'єктів), завдання динаміки функціонування системи (діаграми послідовностей, діаграми станів), взаємодії паралельних процесів (діаграми активностей) та ін. Зазначені вище діаграми грають вирішальну роль в формалізованому представленні предметної області і, як показали дослідження в області програмної інженерії [131], можуть бути отримані з текстового опису об'єкту формалізації природною мовою.

Завдання формалізації предметної області є досить специфічним і вимагає високого рівня знань і умінь, які не завжди є необхідними і доступними таким користувачам САН як тьютор, експерт в предметній області і студент. Саме тому для реалізації завдань формалізації предметної області і необхідно ввести додаткового користувача — інженера зі знань.

У САН в функціональному модулі формування знань про предметну область вся робота залежить від двох категорій користувачів — експерта в предметній області і інженера зі знань [132,133]. Головним напрямком роботи експерта в предметній області є підготовка курсу з предметної області, а саме виконання наступних базових функцій (рис. 3.1.):

- відбір і вибір матеріалів, які необхідно вивчити;
- розробка супроводжуючих ілюстративних матеріалів;
- розробка таблиць, графіків та інших допоміжних навчальних матеріалів;
- представлення знань з предметної області в текстовій формі та її первинна структуризація на розділи, підрозділи та пункти;
- виділення фрагментів предметної області, які підлягають формалізації;
- визначення ступеня покриття предметної області формалізованими представленнями.

Рис. 3.1. Діаграма прецедентів (Use Case) основних функцій експерта в предметній області

Очевидно, що від роботи експерта в предметній області залежить подальша робота зі знаннями про предметну область навчального курсу, які необхідно формалізувати.

Вибір способу формалізації та необхідна робота для підтримки знань в актуальному стані залежить від інженера зі знань.

Серед основних функцій інженера зі знань виділяють наступні (рис. 3.2.):

- вибір способу формалізації представлених компонентів знань;
- формалізація виділених представлених компонентів знань;
- встановлення прямих посилань між формалізованими знаннями і структурованими текстово-графічними і навпаки;
- підтримка знань в актуальному стані.

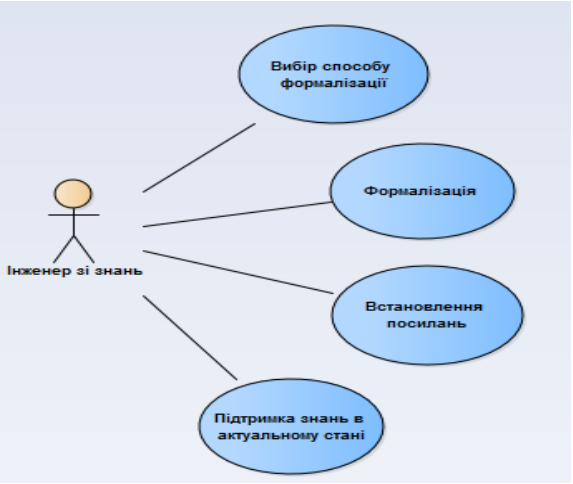


Рис. 3.2. Діаграма прецедентів (Use Case) основних функцій інженера зі знань

Очевидно, що робота над формалізованими представленнями предметної області, виходячи з її текстово-графічного опису, залежить від роботи експерта в предметній області та інженера зі знань (рис. 3.3.).

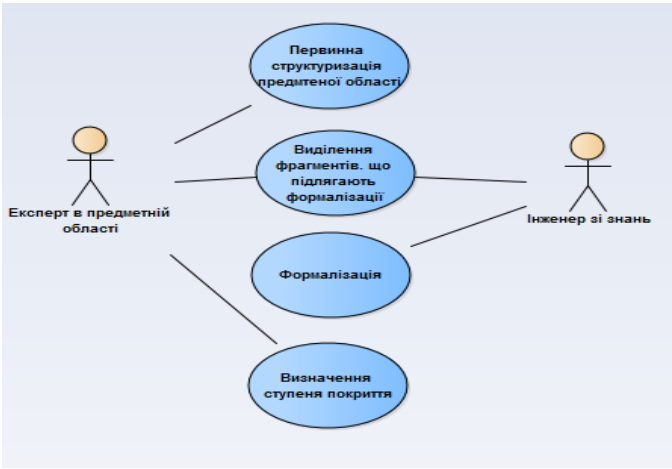


Рис. 3.3. Діаграма прецедентів (Use Case) взаємодії експерта в предметній області та інженера зі знань при формалізації предметної області, виходячи з її текстово-графічного опису

3.4 Порівняльний аналіз використання різних видів представлення знань в САН

Як зазначалося раніше, є широкий спектр методів представлення знань предметної області. Це і природно-мовні подання навчального курсу, і формалізовані представлення фрагментів предметної області у вигляді діаграм UML, формалізовані представлення, що базуються на певному обчисленні, такому як числення предикатів 1-го порядку, фрейми, продукційні правила та ін. Кожне з цих представлень має свої переваги та недоліки.

3.4.1 Природно-мовні представлення

Природно-мовні представлення орієнтовані на людину (на тьютора, студента, експерта в предметній області). Текст навчального курсу розбивається на розділи, підрозділи, пункти, і т. ін. Навігація між окремими частинами текстового представлення здійснюється за допомогою змісту, і як у багатьох видах текстових документів, є термінологічний словник.

Перевагою природно-мовного текстово-графічного представлення предметної області навчального курсу є орієнтація на людину, що забезпечує використання його у всіх автоматизованих навчальних системах.

Недоліками природно-мовних представлень знань навчального курсу є:

- відсутність формалізації в структурі природної мови;
- наявність великої кількості невизначеностей, синонімів, омонімів;
- структуризація текстового документу обмежується його змістом і при наявності сторінкового розбиття тексту в якійсь мірі, термінологічним словником;
- інформація, що включається в розділи, підрозділи може не носити цілісний характер, що не властиво предметній області.

3.4.2 Формалізоване UML-представлення предметної області

Формалізоване UML-представлення предметної області має графічний інтерфейс, зрозумілий інженерам зі знань, фіксовану семантику елементів мови, більш строгу семантику мовних конструкцій. Фактично UML — це багатомовна система, здатна описувати як статику (класові діаграми), так і динаміку предметної області (діаграми послідовностей,

діаграми станів, діаграми активностей). Незважаючи на солідний теоретичний фундамент для деяких підмов, UML-семантика мовних конструкцій залишає бажати кращого. Проте, за допомогою UML можна досягти компактного опису предметної області. Досвід використання UML, набутий при об'єктно-орієнтованому проектуванні програмних систем, пов'язаний з перетворенням текстового опису системи в UML-діаграми, надає можливість його використання як засобу організації баз знань в САН.

Формалізація опису предметної області, що базується на використанні певних математичних теорій, визначається властивостями цих теорій. Процес такої формалізації предметної області, відштовхуючись від зрозумілих людині текстових описів, є досить трудомістким і вимагає залучення експертів в даних теоріях. Наприклад, побудова динамічної предметної області у вигляді мережі Петрі або напівмарківського процесу при описі реальних систем є складним завданням. Складність опису виникає і при спробі використання для формалізації представлення предметної області різних варіацій математичної логіки, включаючи теорію предикатів першого порядку, різні системи на основі логічних правил, і т. ін.

Однак ці формалізми дозволяють використовувати при аналізі формалізованих представлень предметної області розвинений математичний апарат, включаючи механізми логічного виводу, імітаційне моделювання, та ін.

Зведена інформація про можливості різних варіантів представлень предметної області, отримана на базі експертних оцінок надана в табл. 3.1.

Таблиця 3.1. Порівняльні характеристики різних представлень знань про предметну область

	Критерії для порівняння:			
Види представлень знань про предметну область	*Можливість легкого сприйняття предметної області навчального курсу*	*Рівень формалізації предметної області навчального курсу*	*Можливості зручного опису структури системи*	*Можливості зручного опису динамічних властивостей предметної області*
Природно-мовні представлення предметної області	5	0	1	1
UML-діаграми	4	3	4	4
Логічні обчислення у вигляді предикатів	1	5	4	3
Логічні обчислення у вигляді правил	2	5	2	3
Фрейми	1	3	1	1

З вище сказаного слідує, що найкращої форми представлення знань навчального курс не існує. Більш того, є необхідність в САН використовувати кілька його представлень відповідно до цілей використання:

- при роботі тьютора і студента — текстові представлення;
- при автоматизованій оцінці знань студента — UML-діаграми, логічні числення;
- при адаптації навчального матеріалу до рівня знань студента — текстові представлення, UML-діаграми.

Серед наведених у табл. 3.1 критеріїв для порівняння методів представлень знань необхідно розглянути більш детально наступні два:

- критерій можливості легкого сприйняття предметної області навчального курсу кінцевим користувачем;
- критерій рівня формалізації предметної області навчального курсу.

Очевидна важливість першого критерію: чим ближче опис предметної області до рідної природної мови користувача, тим більше можливостей для сприйняття, аналізу і використання навчального матеріалу, як для студента, так і для осіб, які готують інформацію курсу для САН.

Для оцінки ступеня важливості другого критерію необхідно враховувати, що основним для контролю знань при навчанні є проведення опитувань з пройденого матеріалу (іспит). Можливість контролювати засвоєння навчального матеріалу напряму пов'язаний з цим критерієм.

3.5 Архітектура бази знань САН

В якості архітектури бази знань (БЗ) САН можна запропоновувати структуру, представлену на рис. 3.4 [134]. Її особливостями є:

1. Запропонована БЗ САН використовує три способи представлення предметної області, а саме природно-мовне, UML-представлення та логічні представлення. Перший спосіб призначений для студента і тьютора, а в частині формування — експертам в ПО та інженерам зі знань, другий призначений для роботи інженера зі знань, третій використовується для забезпечення повного контролю знань по навчальному курсу. Кожному способу представлення знань відповідає притаманний йому шар в БЗ.
2. В БЗ передбачено засоби автоматизованого формування шару структурного представлення за даними шару текстового представлення предметної області. Ці засоби в основному використовуються експертом в предметній області та інженером зі знань.
3. В БЗ передбачено засоби автоматизованого формування шару логічного представлення за даними шару структурного представлення.
4. БЗ в САН підтримує роботу модуля контролю знань за рахунок модулів генерації простих питань і задач відповідно для 2-го і 3-го шарів.
5. В БЗ передбачено модулі розв'язання типових задач, як на рівні другого, так і для третього шарів.

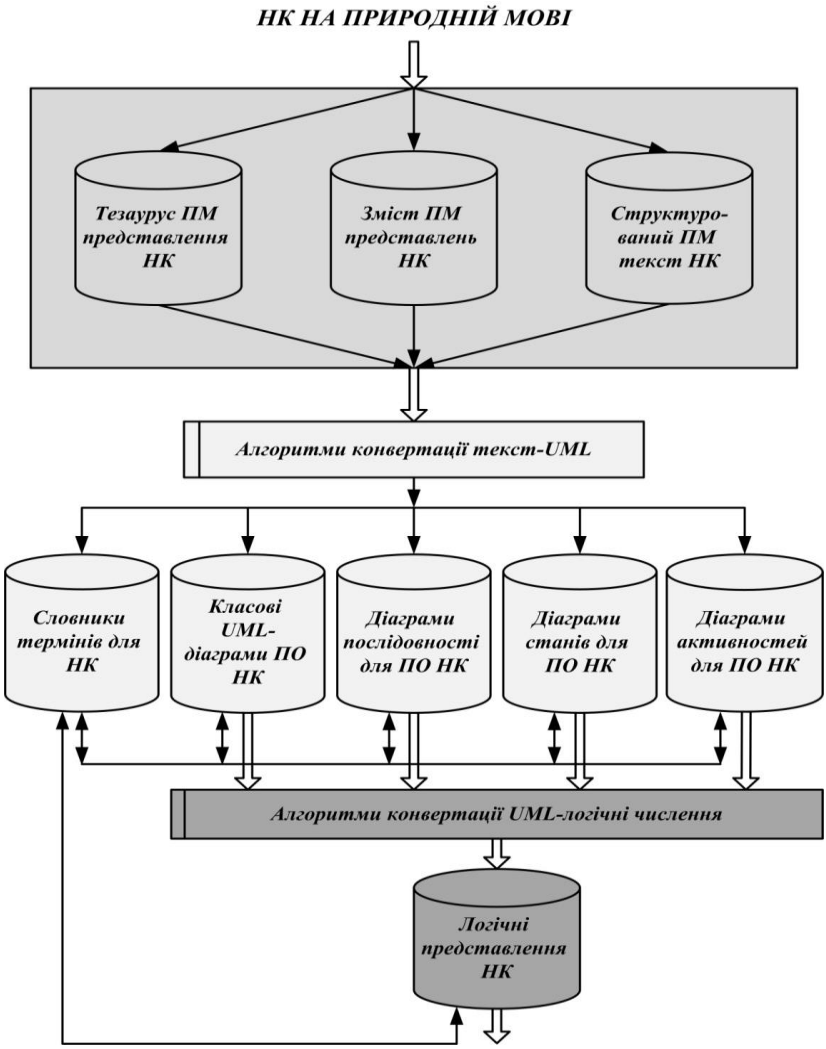


Рис. 3.4. Архітектура БЗ САН

Така архітектура дозволяє зберегти і ефективно використовувати переваги кожного з шарів представлення знань в загальній системі.

При даній архітектурі важливу роль в забезпеченні ефективності роботи системи відіграють алгоритми конвертації ПМ-текст → UML і UML → Логічні представлення.

РОЗДІЛ 4. ПОБУДОВА ФОРМАЛІЗОВАНОЇ МОДЕЛІ ПРЕДМЕТНОЇ ОБЛАСТІ

Із запропонованої структури БЗ САН слідує, що однією з найважливіших задач підтримки БЗ є задача розробки методів формалізації предметних областей навчальних курсів. Згідно з описаним вище трьох-шаровим представленням БЗ задача конвертації природно-мовних представлень в UML-діаграми є актуальною. Це надасть можливість більш ефективно застосовувати знання в процесі навчання та контролю знань студента.

4.1 Підходи для виділення об’єктів і зв’язків, які використовуються для побудови формалізованої моделі предметної області

Для вирішення задачі раціонально використовувати підходи, що розроблені в об’єктно-орієнтованому проектуванні програмних систем. Зокрема для виділення об’єктів і зв’язків предметної області використовують наступні підходи [135]:

- підкреслення іменників;
- ідентифікація причинних об’єктів;
- ідентифікація сервісів (пасивні об’єкти);
- ідентифікація на базі об’єктів реального світу;
- ідентифікація об’єктів на базі фізичних пристроїв;
- ідентифікація ключових концептів;
- ідентифікація об’єктів на базі транзакцій;
- ідентифікація інформації, що зберігається;
- ідентифікація візуальних елементів;
- використання сценаріїв;
- ідентифікація зв'язків між об’єктами.

Перераховані вище підходи виділення об’єктів і зв’язків можуть використовуватися як окремо, так і в сукупності при текстовому аналізі. Однак слід зазначити слабкий ступінь формалізації даного процесу, як щодо побудови класових діаграм, так і у відношенні до діаграм, що описують сценарії поведінки.

При роботі з природно-мовним представленням предметної області найчастіше використовується підхід підкреслення іменників. Визначення

потенційних об’єктів здійснюється шляхом підкреслення кожного іменника або фраз з іменниками в тексті. Таким чином, формується словник потенційних об’єктів предметної області.

Претенденти, ідентифіковані подібним чином, можуть потрапити в одну з чотирьох категорій:

- об’єкти, що представляють інтерес;
- актори;
- об’єкти, які не представляють інтересу;
- атрибути об’єктів.

При застосуванні цього підходу необхідно враховувати деякі особливості:

1. актори зазвичай вже ідентифіковані в моделі прецедентів, але іноді виявляються деякі нові актори, які не мають прямого відношення до нашої системи;
2. в описі проблеми атрибути також представлені як іменники;
3. іноді атрибут може трактуватися як властивість об’єкту;
4. якщо є сумнів, необхідно класифікувати іменник як об’єкт;
5. якщо подальший аналіз показує, що об’єкт не є суттєво цікавим, тоді він може бути включений як атрибут в деякий інший об’єкт.

Для подальшої формалізації предметної області навчального курсу, необхідно вирішити такі задачі:

1. задачу обробки природно-мовних текстів з метою виділення об’єктів, атрибутів, класів об’єктів і відношень між ними;
2. задачу інтеграції класових діаграм для різних розділів навчального курсу в єдину предметну область навчального курсу;
3. задачу оцінки показників якості класових діаграм навчального курсу;
4. задачу еквівалентних перетворень класових діаграм навчального курсу.

4.2 Обробка природно-мовних текстів навчального матеріалу з метою виділення об’єктів, атрибутів, класів об’єктів і відношень між ними

Узагальнений алгоритм обробки тексту для формалізації предметної області в термінах UML має наступний вигляд [132]:

1. Виділення з тексту T множин словосполучень, що позначають групи іменника N_t, що утворюють словник первинних імен сутностей.
2. Переробка N_t і тексту T з метою позбавлення від синонімів та омонімів. Останні перейменовуються, а для синонімів в словнику залишається єдиний. В результаті утворюється словник N_t' і текст T'.
3. Групування речень тексту T' на термінологічні групи G_n, де $n \in N_t'$. В принципі, речення тексту T' може включатися в кілька груп.
4. Співставлення кожному елементу $n \in N_t'$ інформації про кількість сутностей в системі, відповідному текстовому опису T'.

Таким чином, визначається функція $F: N_t' \to L$. Тут в якості елементів множини L можуть виступати цілі позитивні числові значення і невизначеність.

5. На елементах m множини N_t' виділяються атрибутивні зв'язки r_m, які визначаються наступним чином: сутність $n \in N_t'$ є атрибутом сутності $m \in N_t'$, $((n, m) \in r_m)$, якщо n не містить інших сутностей і її можна розглядати, як властивість сутності m. Виділення атрибутивних зв'язків раціонально вести з використанням онтологій [83] або словників-глосаріїв з даної предметної області.

Множину атрибутів множини N_t' позначають N_t^a, а множину атрибутивних зв'язків — R_a.

6. На елементах m множини N_t' виділяються зв'язки узагальнення. Вони визначаються в такий спосіб: сутність $n \in N_t'$ є прямим узагальненням сутності $m \in N_t'$, якщо n є узагальненням m і не існує відмінної від n і m сутності l, яка є узагальненням m і для якої n є узагальненням. Множину прямих зв'язків узагальнення позначається через R_o. Виділення зв'язків узагальнення раціонально вести з використанням онтологій або словників-глосаріїв.
7. На елементах m множини N_t' виділяються зв'язки агрегації $R_{ag} = \{r_n^m\}$, які розглядаються, як входження однієї сутності $n \in N_t'$ в іншу $m \in N_t'$, в якості елемента. При цьому, розрізняються два види зв'язків агрегації: зв'язки, для яких об'єкти сутності n породжуються, знищуються і не можуть існувати поза об'єкту що породив m (зв'язки композиції) і об'єкти з тимчасовим входженням до складу інших об'єктів. Множини таких зв'язків позначаються R_{ag}^1 і R_{ag}^2 відповідно, $R_{ag} = R_{ag}^1 \cup R_{ag}^2$.

Сутності n і m не повинні мати зв'язок агрегації. У термінологічних групах G_n і G_m повинні бути прийменниково-дієслівні форми, що відображають відношення включення n в m.

8. Для речень кожної термінологічної групи G_n, $n \in N_t'$ можуть існувати дієслівно-прийменникові форми, що відображають відношення слідування подій, в яких беруть участь об'єкти сутностей n і m. Ці зв'язки носять назву асоціативних зв'язків r_{nm}, а множина асоціативних зв'язків позначається як $R_{as} = \{r_{nm}\}$. Розрізняють два види асоціативних зв'язків: однонаправлені, які трактуються як те, що подія, в якій бере участь n передує події, в якій бере участь m, і двонаправлені, які трактуються як два однонаправлені зв'язки, що діють в протилежних напрямках.
9. Інші види зв'язків визначаються дієслівно-прийменниковими формами для сутностей n і m в реченнях термінологічної групи G_n, $n \in N_t'$ або для предметної області не мають значення, чи відносяться до категорії «залежність».

4.3 Елементарні операції, що використовуються при побудові формалізованих моделей предметних областей на базі UML

Описаний вище алгоритм обробки тексту для формалізації предметних областей в термінах UML має істотні *недоліки*, а саме:

- важко піддається автоматизації;
- розрахований на роботу з невеликими за обсягом текстами;
- крім, власне, тексту вимагається залучення додаткового матеріалу, такого як глосарії, термінологічні словники, онтології.

При цьому залучення експерта в предметній області є неодмінною умовою. Максимум, на що можна розраховувати при цьому підході — це на створення кістяка формалізованого представлення предметної області або її невеликих фрагментів. Разом з тим, для створення реальних формалізованих предметних областей потрібно як залучення багатьох експертів, так і виділення необхідного часу.

Подолання вище перерахованих недоліків може бути зроблено за рахунок:

- введення елементарних операцій перетворення формалізованих схем;

- використання стратегій побудови предметних областей, їх поступового формування за рахунок деталізації або узагальнення;
- застосування операцій інтеграції формалізованих фрагментів предметних областей в навчальний курс.

Першим кроком є введення множин елементарних операцій діаграм. Будь-яку **елементарну операцію** можна розглядати як функцію *f*, визначену на множині діаграм (схем) *S* в множину схем *S*, *f: S→S* [132].

В якомусь сенсі набори елементарних операцій для формування UML діаграм в початковому вигляді є в будь-яких об'єктно-орієнтованих CASE системах [131]. Наприклад, при побудові ER моделей в [129] також використовуються елементарні операції.

В даному підході при побудові формалізованого представлення предметної області з орієнтацією на UML ми вводимо два види елементарних операцій:

- елементарна операція «зверху-вниз»;
- елементарна операція «знизу-вгору».

Перші раціонально використовувати при побудові предметної області проводячи ієрархічну деталізацію зверху-вниз. Другі — навпаки, при побудові предметної області знизу-вгору, за рахунок введення узагальнень складових сутностей.

4.3.1 Елементарні операції «зверху-вниз»

Елементарні операції «зверху-вниз» характеризуються такими властивостями:

- початкова схема являє собою єдиний концепт, а результуюча складається з невеликого набору концептів;
- всі імена концептів перетворюються в нові імена, що описують вихідний концепт на більш низькому абстрактному рівні;
- логічні зв'язки повинні успадковуватися єдиним концептом результуючої схеми.

Наведемо можливий набір елементарних операцій «зверху-вниз» для перетворення класових UML діаграм, що описують предметну область:

1. Група елементарних операцій T1, перетворює сутність в відношення між двома сутностями. Операції цієї групи можуть утворювати відношення наступних типів:

- двонаправлена асоціація, з кардинальними числами або без них $T_{1,1}$, $T_{1,2}$;
- однонаправлена асоціація з кардинальними числами або без них $T_{1,3}$;
- пряма або зворотна агрегація $T_{1.4,}$ $T_{1,5}$;
- пряме або зворотне узагальнення $T_{1,6}$, $T_{1,7}$;
- пряма або зворотна композиція $T_{1,8}$, $T_{1,9}$.

2. Група елементарних операцій T2,1(n), що деталізують сутність в ієрархію з n+1 сутностей з використанням відношення узагальнення.

3. Група елементарних операцій T3,1(n) розбиває сутність на множину з n незалежних сутностей.

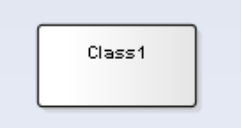
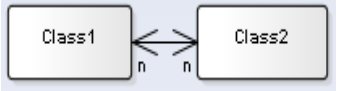
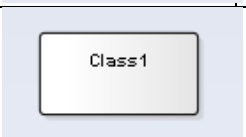
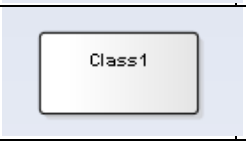
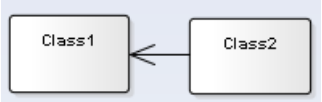
4. Група елементарних операцій T4,1(n) розбиває відношення асоціації між двома сутностями на дві, або більше відношень (n) між тими ж сутностями.

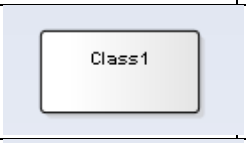
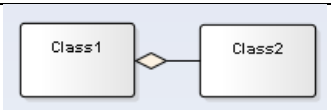
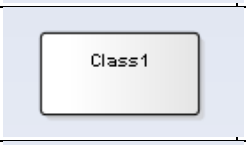
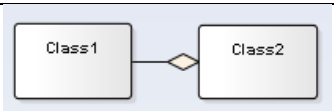
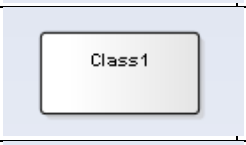
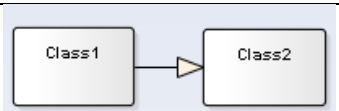
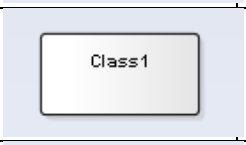
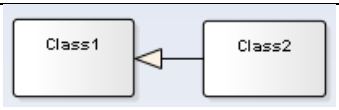
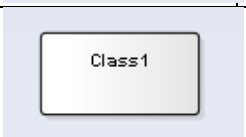
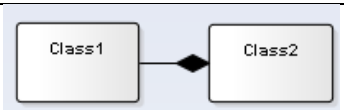
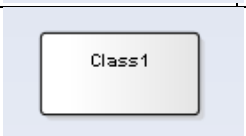
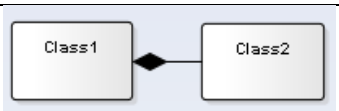
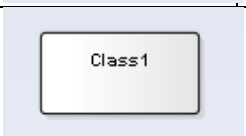
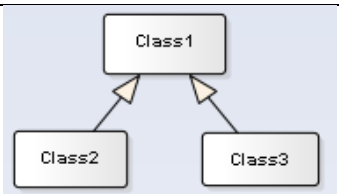
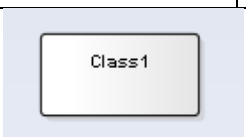
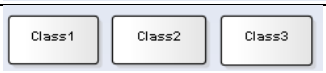
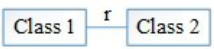
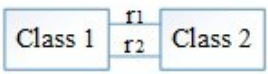
5. Група елементарних операцій T5,1(n) перетворює відношення асоціації в ненаправлений асоціативний шлях між тими ж сутностями, що проходить між новими n проміжними сутностями.

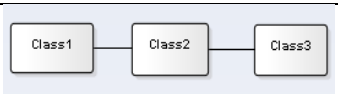
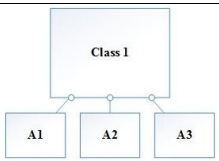
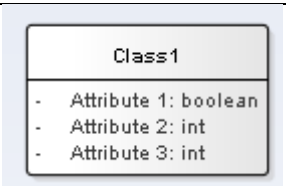
6. Елементарна операція T6,1 вводить в сутність її атрибути.

Графічне представлення елементарних операцій «зверху-вниз» наведено в таблиці 4.1.

Таблиця 4.1. Елементарні операції «зверху-вниз»

№	Група	Елементарні операції	Вхідний фрагмент	Результат
1	T_1	$T_{1,1}$		
2	T_1	$T_{1,2}$		
3	T_1	$T_{1,3}$		

№	Група	Елементарні операції	Вхідний фрагмент	Результат
4	T_1	$T_{1,4}$		
5	T_1	$T_{1,5}$		
6	T_1	$T_{1,6}$		
7	T_1	$T_{1,7}$		
8	T_1	$T_{1,8}$		
9	T_1	$T_{1,9}$		
10	T_2	$T_{2,1}(n)$		
11	T_3	$T_{3,1}(n)$		
12	T_4	$T_{4,1}(n)$		

№	Група	Елементарні операції	Вхідний фрагмент	Результат
13	T_5	$T_{5,1}(n)$		
14	T_6	$T_{6,1}$		

Зазначені вище елементарні операції «зверху-вниз» не дозволяють створити всі допустимі класові діаграми, оскільки побудова моделі предметної області відбувається шляхом вертикальної деталізації виділеного абстрактного концепту.

4.3.2 Елементарні операції «знизу-вгору»

Елементарні операції «знизу-вгору» вводять нові концепти і властивості, які були відсутні в попередніх версіях діаграми, або модифікують деякі її концепти.

Елементарні операції «знизу-вгору» використовуються при проектуванні діаграми, при розкритті тих особливостей предметної області, які не було виявлено на будь-якому рівні абстракції попередньої версії діаграми. Вони використовуються також тоді, коли діаграма перетворюється в більш загальну схему.

Наведемо можливий набір елементарних операцій «знизу-вгору» для перетворення класових UML діаграм, що описують предметну область.

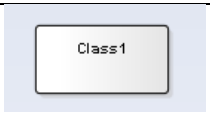
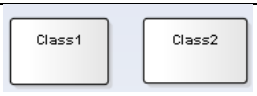
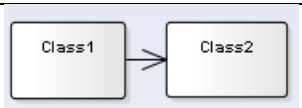
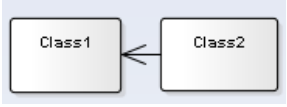
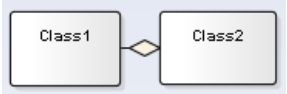
1. Елементарна операція B_1 породжує в діаграмі нову сутність. Вона вводиться, коли інженер зі знань виявив новий концепт зі специфічними властивостями, який не з'являвся в попередній діаграмі.
2. Група елементарних операцій B_2 породжує нове відношення між раніше визначеними сутностями діаграми:
   - елементарна операція $B_{2,1}$ для відношення двонаправленої асоціації;

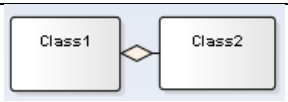
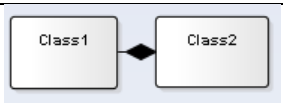
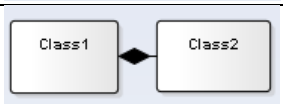
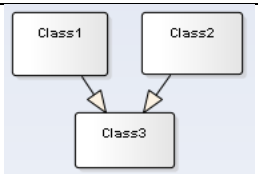
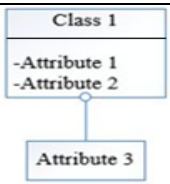
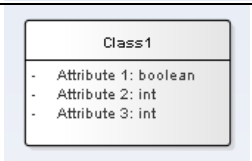
- елементарна операція $B_{2,2}$ для відношення прямої однонаправленої асоціації;
- елементарна операція $B_{2,3}$ для відношення зворотної однонаправленої асоціації;
- елементарна операція $B_{2,4}$ для прямого відношення;
- елементарна операція $B_{2,5}$ для зворотного відношення агрегації;
- елементарна операція $B_{2,6}$ для прямого відношення композиції;
- елементарна операція $B_{2,7}$ для зворотного відношення композиції.

3. Елементарна операція B_3 породжує нову сутність, яка розглядається як узагальнення (вершина ієрархії) для раніше визначених в діаграмі сутностей.
4. Елементарна операція B_4 породжує новий атрибут для раніше визначеної сутності.

Графічне представлення елементарних операцій «знизу-вгору» наведено в таблиці 4.2.

Таблиця 4.2. Елементарні операції «знизу-вгору»

№	Група	Елементарна операція	Вхідний фрагмент	Результат
1	B_1	B_1		
2	B_2	$B_{2,1}$		
3	B_2	$B_{2,2}$		
4	B_2	$B_{2,3}$		
5	B_2	$B_{2,4}$		

№	Група	Елементарна операція	Вхідний фрагмент	Результат
6	B_2	$B_{2,5}$		
7	B_2	$B_{2,6}$		
8	B_2	$B_{2,7}$		
9	B_3	$B_{3,1}$		
10	B_4	$B_{4,1}$		

4.4 Стратегії, що використовуються при побудові формалізованих моделей предметних областей на базі UML

Загальні принципи використання елементарних операцій при побудові моделей предметної області назвемо *стратегіями* побудови предметної області.

Для побудови класових діаграм, що описують предметну область, будемо розрізняти чотири стратегії:

- «зверху-вниз»;
- «знизу-вгору»;
- «зсередини-назовні»;
- змішана

Кожна з них характеризується використанням певних елементарних операцій.

4.4.1 Стратегія «зверху-вниз».

При використанні стратегії «зверху-вниз», класова діаграма, що описує предметну область, будується шляхом деталізації за рахунок використання тільки елементарних операцій «зверху-вниз». Кожне використання операції додає нові деталі в діаграму. На кожному кроці перетворення відбувається деталізація одної сутності або зв'язку, в той час як інша частина діаграми залишається незмінною.

Таким чином, процес побудови предметної області, представленої у вигляді класової UML діаграми в випадку використання стратегії «зверху-вниз» представляється у вигляді послідовності застосувань до схеми діаграми елементарних операцій «зверху-вниз»: $\langle T^1, T^2, \dots, T^n \rangle$, де $T^i \in T, i = 1, \dots, n$, при чому вихідна діаграма, S_0, що оброблюється T^1, складається з єдиної сутності, а кінцева класова діаграма S_n, що отримується в результаті застосування операції. T^n — є шукане формалізоване представлення предметної області. При цьому $S_0 < S_1 < S_2 \dots < S_n$.

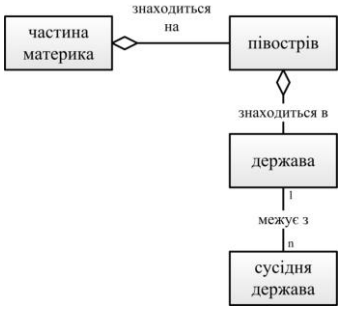
Для прикладу побудови предметної області, виконаної відповідно до стратегії «зверху-вниз» було вибрано фрагмент предметної області, представлений природно-мовним способом. За допомогою підходу підкреслення іменників було встановлено основні об'єкти, які в процесі аналізу було віднесено або до класів, або до їх атрибутів. Нижче наведено приклад застосування стратегії «зверху-вниз».

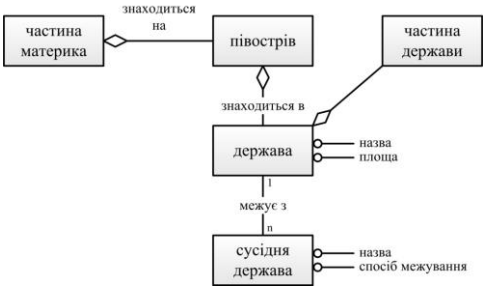
«Офіційна назва Швеції — Королівство Швеція. Держава знаходиться в Північній Європі на Скандинавському півострові. Межує з Норвегією та Фінляндією, також з'єднана з Данією мостом-тунелем через протоку Ересунн. Займаючи територію у 450 295 км², Швеція є третьою за площею країною Європейського Союзу, і має населення чисельністю 9,7 мільйонів осіб. Країна загалом має низьку щільність населення у 21 людину на квадратний кілометр, з найвищою концентрацією у південній половині країни. Приблизно 85 % населення живе у міський районах. Південна Швеція є переважно сільськогосподарською, у той час, як її північна частина — густо покрита лісами. Швеція є частиною географічного регіону Фенноскандія.»

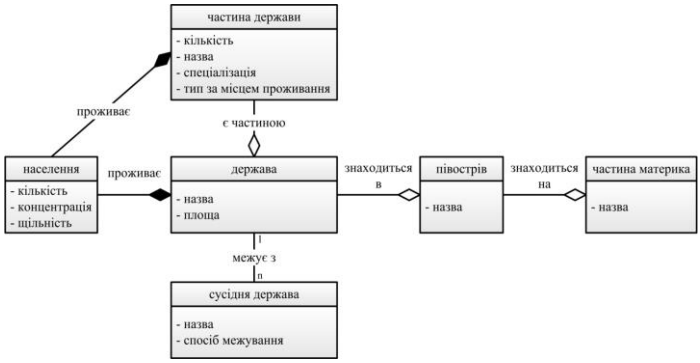
У процесі аналізу було виділено абстрактний концепт «півострів». Далі, деталізуючи поняття було встановлено відношення агрегації між «півостровом» та «частиною материка» та між «півостровом» і «державою». На наступному кроці було виділено клас «сусідня держава», яка має

асоціативне відношення з «державою». Після цього, аналізуючи текстово-графічне представлення предметної області було з'ясовано, що іменники «назва», «площа» та «спосіб межування» є атрибутами відповідно класів «держава» та «сусідня держава». На наступних кроках встановлено відношення агрегації між «державою» та «частиною держави», виділено атрибути «частини держави». Далі, використовуючи елементарну операцію $T_{6,1}$, в сутності було введено її атрибути. На наступних кроках було введено клас «населення» та визначено його атрибути, встановлено відношення композиції між «населенням» та класами «держава» і «сусідня держава». Графічне відображення цього процесу наведено в таблиці 4.3. Деякі кроки було об'єднано, щоб не нагромаджувати інформацію.

Таблиця 4.3. Приклад процесу деталізації предметної області, виконаний відповідно до стратегії «зверху-вниз»

Крок	**Графічне відображення**
1	
2	
3	

Крок	**Графічне відображення**
4	
5	
6	

Крок	**Графічне відображення**
7	

4.4.2 Стратегія «знизу-вгору».

При використанні стратегії «знизу-вгору», класова діаграма, що описує предметну область, будується шляхом деталізації за рахунок використання тільки елементарних операцій «знизу-вгору». Процес починається з елементарних концептів, на базі яких з використанням операцій будуються більш складні концепти, і т. ін.

Якщо процес побудови формалізованого представлення предметної області розглядати, як послідовність перетворень, які виконуються над діаграмами за допомогою елементарних операцій «знизу-вгору» $\langle B^1, B^2, \dots, B^n \rangle$, де $B^i \in B, i = 1, \dots, n$, то тут на вхід B^1 надходить діаграма, що складається з низькорівневих сутностей, що мають відображення в первинному словнику термінів-іменників, а кінцева класова діаграма S_n, виходить в результаті застосування елементарної операції B^n. B^n — є шукане формалізоване представлення предметної області. При цьому $S_0 < S_1 < S_2 \dots < S_n$.

Поступове додавання абстрактних сутностей може привести до часткової реструктуризації діаграми за рахунок перенесення властивостей сутностей шляхом, який визначається відношеннями узагальнення.

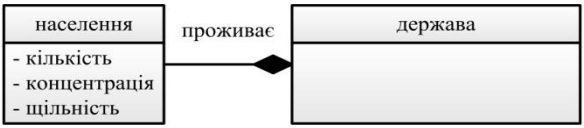
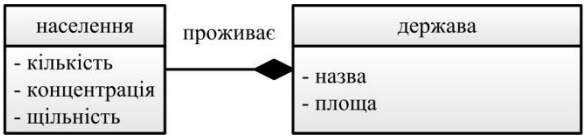
Для прикладу побудови предметної області, виконаної відповідно до стратегії «знизу-вгору» було використано аналогічний фрагмент предметної області, представлений природно-мовним способом.

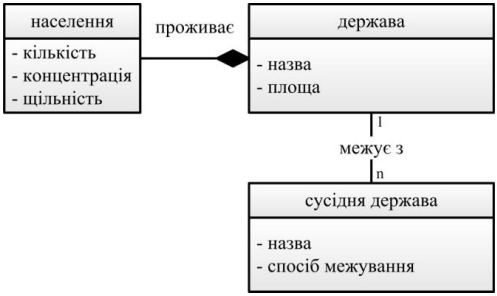
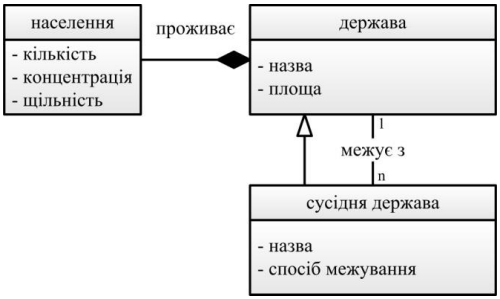
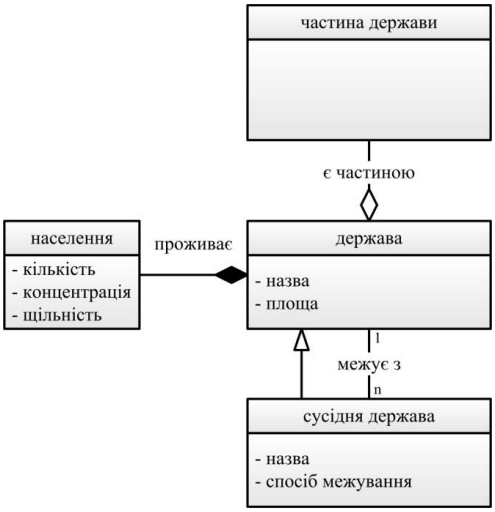
При цьому в процесі аналізу було визначено клас «населення», та його атрибути — «кількість», «концентрація» та «щільність». На наступних кроках було виділено класи «держава» та «сусідня держава», введено атрибути в класи та встановлено відношення між класами.

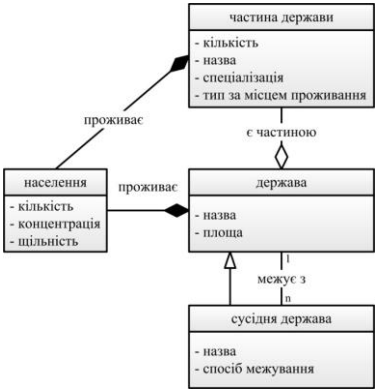
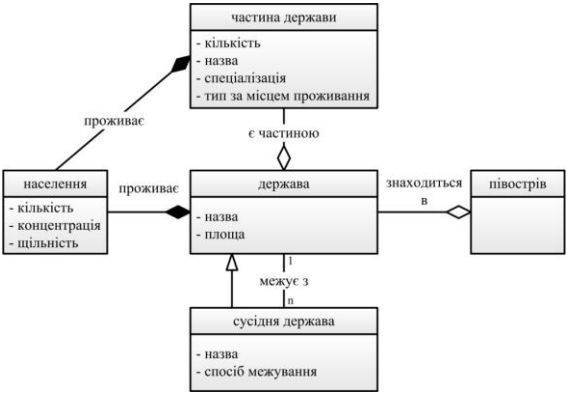
Далі було введено клас «частина держави» та визначено його атрибути. Після цього було встановлено відношення композиції між «населенням» та «частиною держави», виділено клас «півострів» та введено його атрибути. На наступних кроках було виділено клас «частина материка» та введено його атрибути.

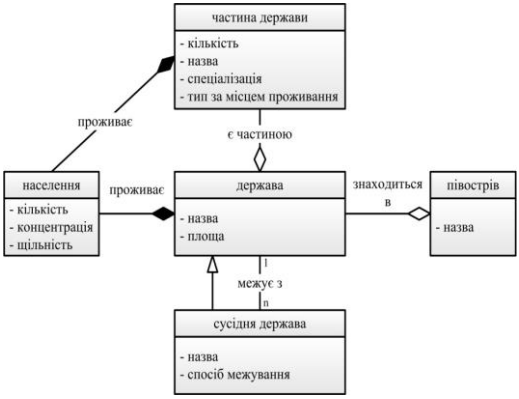
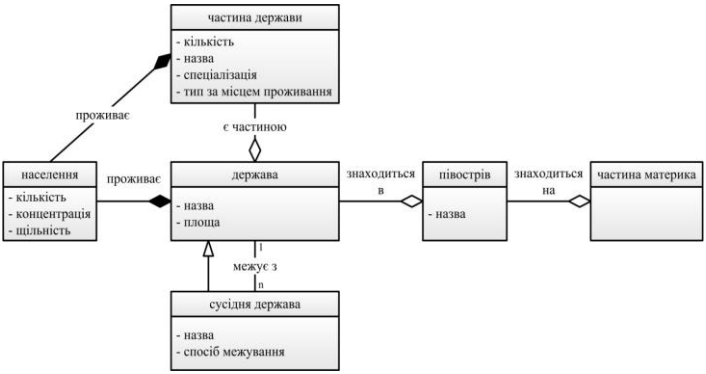
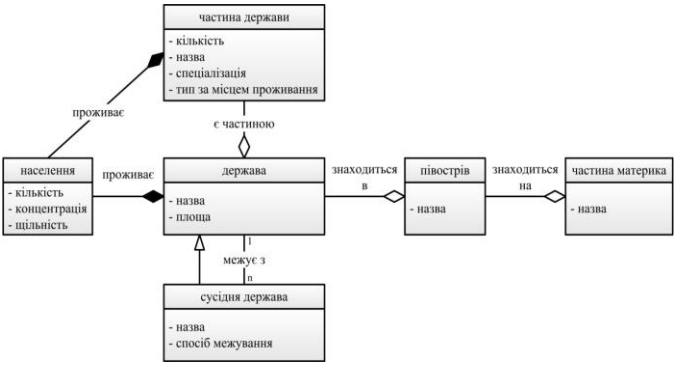
Графічне відображення цього процесу наведено в таблиці 4.4. Деякі кроки було об’єднано, щоб не нагромаджувати інформацію.

Таблиця 4.4. Приклад процесу побудови класової діаграми предметної області, виконаний відповідно до стратегії «знизу-вгору»

Крок	Графічне відображення
1	
2	
3	

Крок	Графічне відображення
4	
5	
6	

Крок	Графічне відображення
7	
8	
9	

Крок	Графічне відображення
10	
11	
12	

Послідовне застосування елементарних операцій «знизу-вгору» може привести до утворення на діаграмі багатьох кластерів пов'язаних сутностей, між якими в подальшому можна встановити відношення.

Перевагою стратегії «знизу-вгору» є простота побудови кластерів, що дає можливість швидко формувати прототипи предметних областей.

Недоліками стратегії «знизу-вгору» є труднощі інтеграції фрагментів складних діаграм і необхідність частої реструктуризації діаграми. Стабілізація складу сутностей і відношень в діаграмі настає тільки в кінці процесу проектування.

4.4.3 Стратегія «зсередини-назовні» або «розповсюдження масляної плями»

Використовуючи стратегію «зсередини-назовні» при проектуванні класової діаграми для формалізованого представлення предметної області, перш за все, обираються найбільш важливі і найбільш очевидні концепти (сутності). Зафіксувавши такі концепти необхідно знайти інші концепти, які є найбільш концептуально близькими до перших. Далі процес повторюється. У ER моделюванні предметних областей цей процес носить назву «процесу розповсюдження масляної плями» [129]. Приклад застосування стратегії «зсередини-назовні» показаний на рис. 4.1.

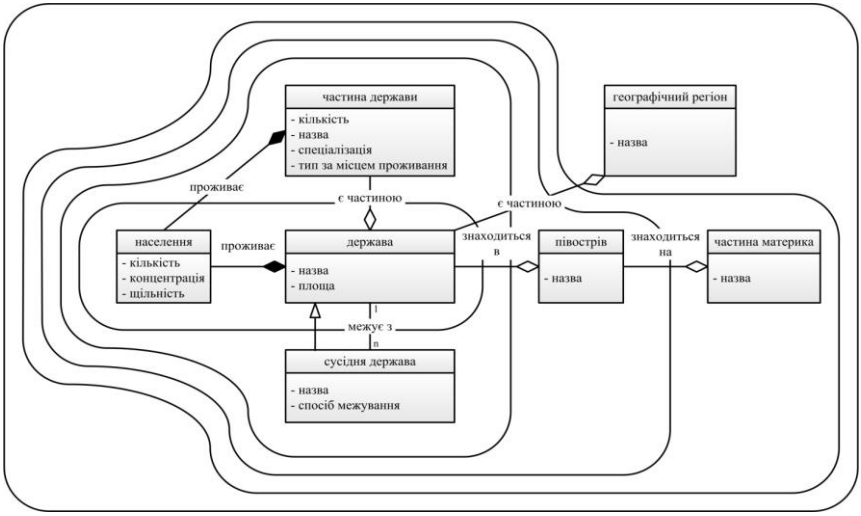


Рис.4.1. Використання стратегії «зсередини-назовні» при побудові моделі фрагменту предметної області

В природно-мовному представленні фрагменту предметної області попереднього прикладу було встановлено найбільш важливі та очевидні сутності — «населення» та «держава». За результатами аналізу тексту було визначено атрибути цих класів та встановлено відношення композиції між класами. Найбільш концептуально близькими концептами до виділених є «сусідня держава» та «частина держави», отже наступними кроками є деталізація цих класів шляхом введення в них атрибутів та визначення відношень між ними. Далі процес продовжується як «розповсюдження масляної плями» доки не буде зафіксовано останній концепт та не буде з'ясовано його атрибути та вид відношень з іншими класами.

Таким чином, стратегія «зсередини-назовні» може розглядатися як окремий випадок стратегії «знизу-вгору». Рівні абстракції концептів, представлених в сусідніх версіях діаграми аналогічні. Однак втрачені переваги процесування, починаючи з абстрактних рівнів.

4.4.4 Змішана стратегія

Головною ідеєю змішаної стратегії є те, що коли предметна область є досить складною, інженер зі знань ділить її текстове представлення на частини, які далі розглядаються окремо.

Одночасно розробник будує скелет діаграми, який містить найбільш важливі концепти і включає зв'язки між ними. Наявність скелета діаграми полегшує інтеграцію різних піддіаграм. На рис. 4.2. наведено приклад побудованої моделі фрагменту предметної області, використовуючи змішану стратегію.

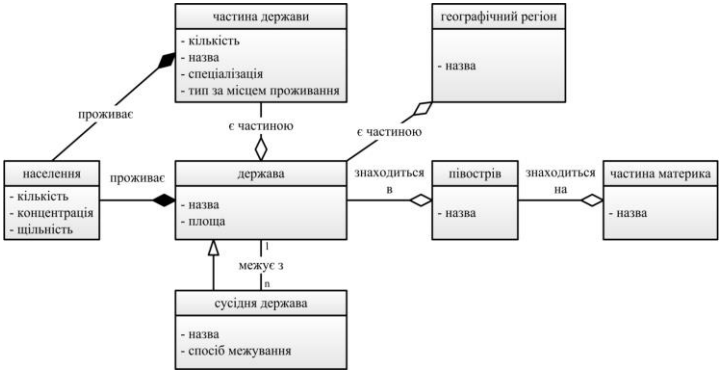


Рис.4.2. Використання змішаної стратегії при побудові моделі фрагменту предметної області

4.5 Життєвий цикл проектування предметної області

Описані вище стратегії проектування формалізованого представлення предметної області дозволяють зробити певні висновки про характер області застосування даних стратегій:

1. стратегії «зверху-вниз» і «знизу-вгору» застосовні для проектування невеликих предметних областей;
2. стратегія «зсередини-назовні» застосовна для проектування формалізованих представлень предметних областей середніх розмірів;
3. змішані стратегії цілком можна застосувати для проектування формалізованих представлень предметних областей цілих навчальних курсів.

Однак в останньому випадку потрібно врахувати і той факт, що окремі фрагменти загальної великої класової діаграми можуть проектувати різні люди.

З урахуванням перерахованих вище зауважень можна представити життєвий цикл розробки формалізованого представлення навчального курсу таким чином:

1. Аналіз тексту курсу T_1 і розбиття його на розділи $T_{1,1}$,$T_{1,2}$,… $T_{1,n}$.
2. Складання скелету формалізованого представлення моделі предметної області S_{sk}.
3. Розробка формалізованих представлень фрагментів предметної області S_1,S_2,…,S_m.
4. Інтеграція фрагментів формального опису предметної області в загальний формальний опис S.
5. Аналіз якості формального опису моделі предметної області S і її коригування з метою поліпшення якості.

Цей різновид життєвого циклу спирається на змішану стратегію розробки, про що свідчать етапи 2, 3, 4.

Результатом відпрацювання 1-го етапу є:

- термінологічний словник для сутностей предметної області курсу (включаючи предметний покажчик) L_1;
- термінологічний словник для дієслівних і прийменникових форм в текстовому представленні курсу L_2;

- стандартизоване представлення тексту курсу, вільне від синонімів, омонімів і перероблене на предмет стандартизації синтаксичних структур T_2;
- існуючі онтологічні описи предметної області курсу S^o.

Результатом відпрацювання 2-го етапу служить скелет S_{sk} діаграми предметної області курсу, що містить множину найбільш важливих концептів і зв'язків між ними:

$$S_{sk} = \langle N_{sk,}, R_{sk} \rangle. \quad (4.1)$$

Результатом відпрацювання 3-го етапу є класова піддіаграма S_i для кожного $i \in N_{sk}$.

Виконання 4-го етапу призводить до отримання неоптимізованого інтегрованого варіанту S формального представлення предметної області курсу, причому процес інтеграції ведеться шляхом використання піддіаграм S_i для кожного концепту скелета S_{sk}.

Виконання 5-го етапу призводить до оптимізованого за показниками якості варіанту класової діаграми S^1, який представляє формалізований варіант представлення предметної області курсу або його розділу.

4.6 Методи інтеграції класових діаграм для різних розділів навчального курсу в єдину предметну область навчального курсу

У процесі інтеграції окремо створених, можливо, різними людьми класових діаграм, піддіаграми зливають в одну. Головною метою інтеграції є знаходження всіх частин вхідних діаграм, які означають опис одного і того ж фрагменту реального світу, уніфікація його формального представлення. Труднощі цього процесу обумовлені тим, що є різні шляхи побудови формалізованого представлення фрагменту.

Труднощі злиття класових діаграм викликані наступними причинами:

- різні точки зору різних розробників на один і той же предмет реального світу;
- еквівалентність конструкцій в моделях предметної області;
- несумісні формальні представлення: помилки під час проектування щодо імен, структур, обмежень.

Кожна з цих причин призводить до конфліктів, тобто до різних представлень одного і того ж концепту. Рекомендується виявляти конфлікти

на ранніх стадіях інтеграції. Під час вирішення конфлікту одна або всі діаграми модифікуються з залученням експерта в предметній області. Під час злиття діаграм виходить попередньо зінтегрована діаграма.

Для подолання цих проблем запропоновано застосовувати методи інтеграції, серед яких виділяють крупноблочну інтеграцію та інтеграцію в малому.

4.6.1 Метод крупноблочної інтеграції

При *крупноблочній інтеграції* можуть існувати десятки, а іноді і сотні фрагментів формалізованих представлень, які необхідно інтегрувати. Це вимагає встановлення відповідного порядку проведення інтеграції. На рис.4.3. і рис.4.4. наведено варіанти даного процесу.

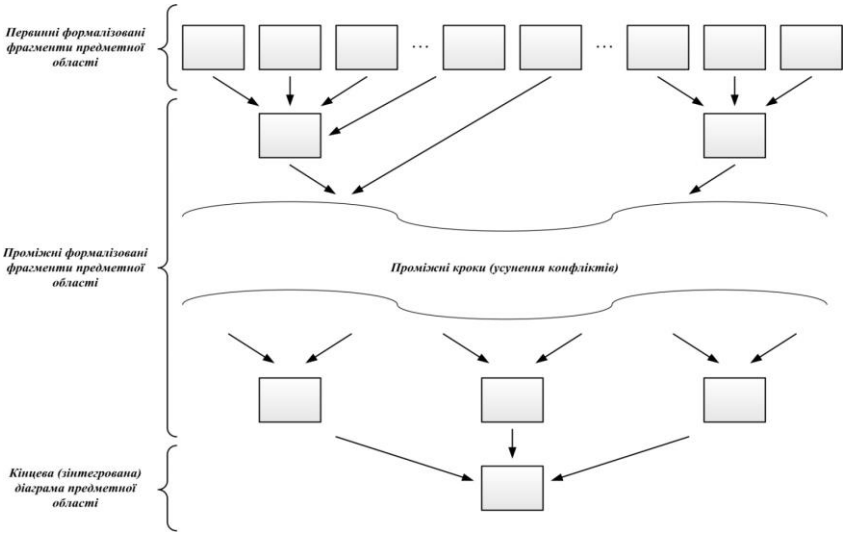


Рис.4.3. Загальна схема крупноблочної інтеграції

Найбільш раціональним варіантом вважається варіант, представлений на рис.4.4.

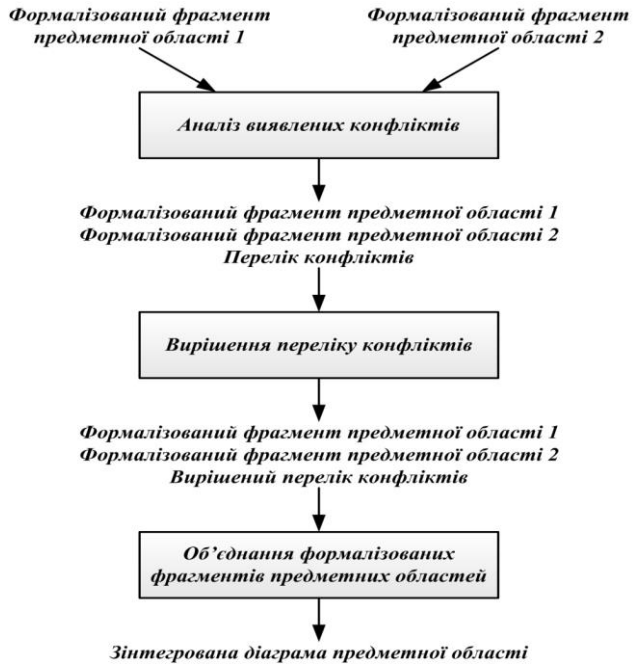


Рис.4.4. Вдосконалений порядок проведення крупноблочної інтеграції

4.6.2 Аналіз та вирішення конфліктів (метод інтеграції в малому)

Метод інтеграції в малому — це аналіз і вирішення конфліктів між парою формалізованих фрагментів. Даний вид інтеграції виявляє всі відмінності в представлення однієї і тієї ж частини реального світу.

Розрізняються два види інтеграційного аналізу в малому:

- аналіз для виявлення конфлікту імен, при якому імена концептів в діаграмах порівнюються і уніфікуються;
- аналіз для виявлення конфлікту в структурах, при якому представлення концептів в діаграмах порівнюються і уніфікуються.

Є два джерела конфлікту імен — синоніми і омоніми. При виявленні їх інженер зі знань керується ступенем подібності або відмінності серед концептів. Ступінь подібності або ступінь відмінності двох концептів

визначається збігом чи відмінностями властивостей і обмежень в діаграмах. Властивості визначаються наборами атрибутів.

Обмеження представляються правилами обмеження числа об'єктів, кардинальними числами, обмеженнями для відношень асоціативності і узагальнення. За результатами ступеня збігу і відмінності концептів можна визначити набір можливих модифікацій діаграм (модифікаційних сценаріїв). Модифікаційний сценарій для конфлікту найменувань включає перейменування концепту. Сценарії можуть включати додавання міждіаграмних властивостей, які визначають взаємні обмеження між концептами, що з'являються в різних діаграмах.

Після виконання аналізу конфліктів імен досягнута уніфікація імен. Під час проведення аналізу конфліктів в структурах концепти з однаковим іменем в початкових піддіаграмах порівнюються для визначення, чи можуть вони бути поєднаними.

Можливе використання наступних категорій:

- ідентичні концепти, які мають ідентичну структуру і властивості;
- сумісні концепти, які мають різні представлення структури або сусідні властивості, які не є суперечливими;
- несумісні концепти, які мають суперечливі властивості.

Джерела несумісності повинні бути видалені, перш ніж зливати піддіаграми в єдину схему. Деякі з можливих несумісностей:

- різні кардинальні числа для сутності;
- різні ідентифікатори: ідентифікатор в одній діаграмі відсутній в інший;
- зворотне входження: в 1-й діаграмі А входить в В, а в 2-й діаграмі В входить в А.

Можливе вирішення несумісностей включає вибір одного представлення замість іншого або побудову загального представлення, в якому всі обмеження двох діаграм підтримуються в зінтегрованій схемі.

4.6.3 Суміщення діаграм

Суміщення двох діаграм приводить до діаграми, всі сутності якої містяться в суміщених. На цій стадії формалізації всі конфлікти вирішені, отже, результуюча діаграма є суперпозицією суміщених. Сутності, які відповідають одна одній в двох діаграмах суперпозиціюються прямо.

Сутності, які відповідають одному і тому ж об’єкту реального світу, але мають різні атрибути, суперпозиціюються шляхом об'єднання атрибутів. Всі ідентифікатори в початкових діаграмах знаходять відображення в результуючій діаграмі. Якщо відношення узагальнення присутні в міждіаграмних властивостях, вони додаються на даному етапі. Після завершення процесу суперпозиціювання всіх сутностей, всі відношення і узагальнення вхідних діаграм мають відповідність в результуючій діаграмі.

4.7 Задача оцінки показників якості класових діаграм навчального курсу

4.7.1 Якість формалізованого представлення предметної області та оптимізація діаграм

Існує цілий ряд показників, що характеризують якість діаграм. За аналогією з ER діаграмами [129] наведемо наступний набір, повністю придатний для UML-діаграм:

- завершеність;
- коректність;
- мінімальність;
- виразність;
- читабельність.

Діаграма вважається *завершеною*, коли вона представляє всі відповідні функції (особливості) предметної області. Завершеність може бути перевірена двома способами:

- перегляд і аналіз тексту на наявність всіх особливостей його деталей, і відображення їх на діаграмі;
- перевірка тексту, на предмет присутності кожного поняття (концепту), яке згадується в ньому.

Діаграма вважається *коректною*, коли всі поняття UML моделі правильно використовуються в ній. Будемо розрізняти два типи коректності: синтаксичну і семантичну.

Синтаксична коректність визначається граматикою мови, на якій представлена діаграма (в нашому випадку UML). Семантична коректність визначається тим, наскільки концепти діаграми використовуються у відповідності з їх визначеннями.

Наведемо список семантичних помилок, що найчастіше зустрічаються:

- використання атрибута замість сутності;
- виключення з використання відношення узагальнення;
- не врахування властивості наслідування при узагальненні;
- виключення використання *n*-арних відношень;
- не врахування кардинальних чисел, та ін.

Діаграма є *мінімальною*, якщо жоден концепт не може бути видалений з діаграми без втрати деякої інформації. Приклад оптимізації діаграми за показником мінімальності наведено на рис. 4.5.

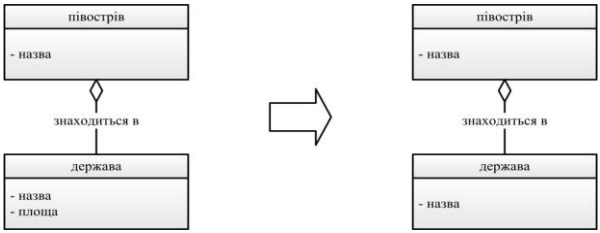


Рис. 4.5. Приклад оптимізації діаграми за показником мінімальності

Діаграма є *виразною*, коли вона легко зрозуміла в термінології UML. Приклад оптимізації діаграми за цим показником наведено на рис.4.6.

Діаграма є *читабельною*, коли в ній дотримані певні критерії, які роблять її витонченою. Головними критеріями цього показника якості є:

- діаграма повинна бути намальована в сітці так, щоб блоки, що представляють сутності мали приблизно однаковий розмір, були вирівняні, як по вертикалі, так і по горизонталі;
- акцентується увага на симетричність структур;
- загальна кількість перетинів зведено до мінімуму (часті переходи знижують пропускну здатність сприйняття користувача);
- загальна кількість вигинів вздовж з'єднань повинна бути зведена до мінімуму;
- в ієрархії узагальнення батьківська сутність повинна розташовуватися над сутностями нащадків, а сутність нащадка повинна бути симетрично розташована по відношенню до батьків.

Приклад оптимізації діаграми за показником читабельності наведено на рис. 4.7.

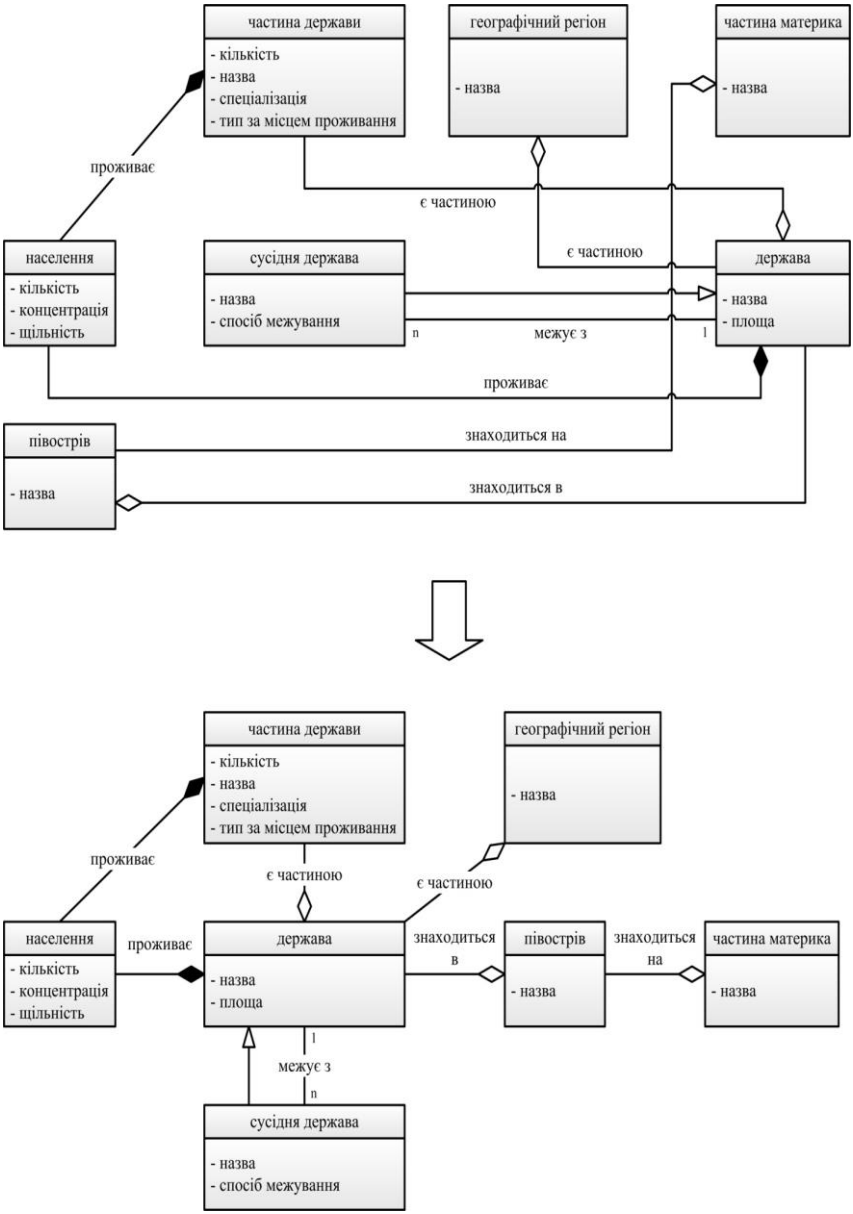


Рис. 4.6. Приклад оптимізації діаграми за показником виразності

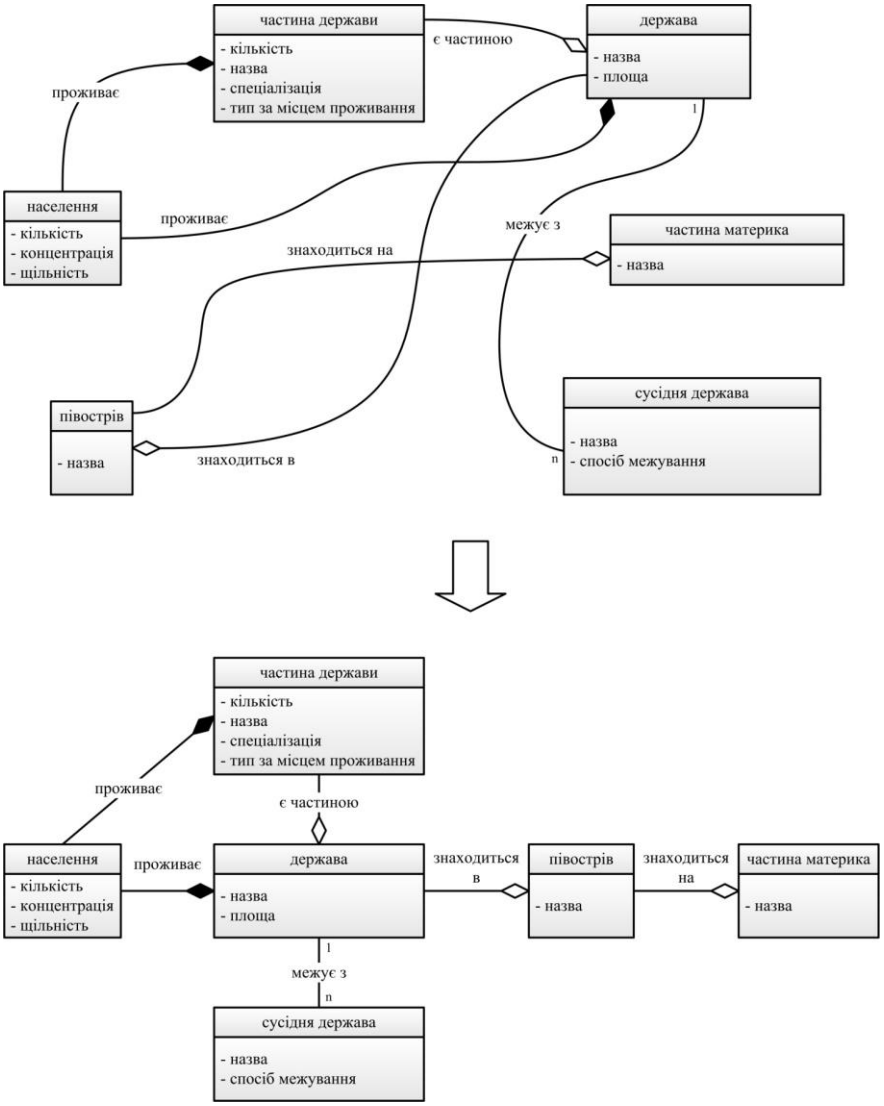


Рис. 4.7. Приклад оптимізації діаграми за показником читабельності

Для отримання значень наведених вище показників якості використовуються методи експертного оцінювання, наприклад PERT- метод [136].

4.7.2 Оптимізація діаграм за показниками якості

Розглянуті вище показники дають можливість з різних точок зору поглянути на отриману діаграму. Відхилення від необхідного рівня якості часто супроводжується появою небажаних ознак, які можуть з'являтися одночасно та іноді носять якісний характер. Тому використовуючи показники при виборі кращого варіанту діаграми можна намагатися отримати вираження ступеня близькості діаграми для заданого показника якості через систему ознак.

Далі, розглядаючи характеристики якості як критерії, можна звести задачу вибору найкращого варіанту діаграми до класичної задачі багатокритеріального вибору.

Враховуючи вищесказане, сформулюємо нижче загальну задачу оптимального вибору варіанта.

Нехай $S = \{S_i, i = 1, \dots, n\}$ — множина варіантів формалізованого представлення однієї і тієї ж предметної області; $C(s) = \langle C_1(s), C_2(s), \dots, C_m(s)\rangle, s \in S$ — вектор функцій оцінки варіанту s за критеріями якості $C_1, C_2, \dots, C_m$, $F(s) = F(C_1(s), C_2(s), \dots, C_m(s))$ — інтегральний критерій якості діаграми $s \in S$.

Необхідно знайти такий $s_i \in S$, для якого при будь-якому $j \neq i$, $i, j \in \{1, \dots, n\}$, $F(s_i) \geq F(s_j)$.

Для вирішення цього завдання на наш погляд ідеально підходить метод аналізу ієрархій. Детальніше про MAI див. 2.3.2.

4.8 Зауваження з приводу «еквівалентних» перетворень діаграм в технології оптимізації діаграм

У наведеній вище формалізованій постановці задачі використовувалося множина варіантів діаграм *S*. Разом з тим, з вищевикладеного виходить, що побудова варіанту формалізованого представлення предметної області — трудомісткий процес, що вимагає залучення як машинних, так і людських ресурсів. Фактично на кожному етапі отримання формалізованого представлення можливе отримання альтернативних рішень. На наш погляд, для формування елементів множини S цікавий інший підхід — отримання за вищеописаною схемою першого елемента $s_1 \in S$ і подальше його «еквівалентне» перетворення з отриманням відповідно $s_2, s_3, \dots, s_n$..

Отже, перетворення діаграм застосовуються до s_i, в результаті чого отримуємо діаграму $s_{i+1}, i = 1, \dots, n-1$.

Вони можуть відрізнятися у відповідності з тими показниками якості, поліпшення яких вони викликають. Однак відмінною рисою всіх цих перетворень є прагнення досягти ідентичності інформаційного контенту схем.

Відомо, що будь-яке формалізоване представлення предметної області породжує множину запитань до неї та множину пов'язаних з ними відповідей і, отже, S непрямо визначає функціональне відображення $F_s: Q \to A$, де Q — множина питань з предметної області, A- множина правильних відповідей на них.

Це відображення визначає інформаційний контент формалізованого представлення предметної області.

Вимогою до «еквівалентного» перетворення діаграм є ідентичність контенту для s_i та $s_{i+1}, i = 1, \dots, n-1$.

Поглинання одного контенту іншим розглядається як відношення <або> між ними. Звідси на множині контентів для діаграм, що служать для опису предметної області можна ввести відношення часткового порядку. Це дає можливість серед множини перетворень діаграм виділити:

1. перетворення, що не порушують контенту;
2. перетворення, які звужують контент;
3. перетворення, що розширюють контент;
4. перетворення, що призводять до контенту, який не можливо порівнювати.

Важливо відзначити, що під час проектування формального представлення предметної області, незважаючи на бажання збереження незмінного інформаційного контенту, часто використовуються перетворення діаграм 3-го або 4-го типів, наприклад, при вирішенні конфлікту імен або при інтеграції піддіаграм.

Що стосується перетворень діаграм, пов'язаних з якістю, то тут особливе місце займають перетворення 1-го типу. Перетворення 2-го типу використовуються тоді, коли текстові представлення не містять концептів, що містяться в діаграмі.

Як правило, надлишковість в діаграмах породжують цикли відношень. В цьому випадку надлишковість з великою часткою впевненості існує, коли одне відношення R між двома сутностями має той же інформаційний контент, що і шлях з відношень $R_i, R_{i+1}, \dots, R_{i+m}$.

Коли цикл містить кілька відношень, які взаємно надлишкові, можливо видалити одне з них. Це стосується і властивостей послідовностей відношень. Наприклад, відношення, отримане комбінацією двох або більше відношень типу «один-до-одного» є теж відношенням типу «один-до-одного».

4.9 Зауваження щодо отримання текстового опису предметної області навчального курсу, виходячи із її представлення у вигляді класових UML-діаграм

Представлення навчального курсу у вигляді класових UML-діаграм більш корисне для кінцевого користувача, коли не тільки з тексту можна будувати UML-представлення предметної області, але і будувати текст на базі отриманих UML-діаграм, тобто:

$$t \to S(t) \to t', \tag{4.2}$$

при чому, не обов'язково $t = t'$.

Основною складністю є включення в UML-діаграми одночасно декількох асоціативних зв'язків, що стосуються одного класу. Вирішення цієї проблеми може стати за рахунок введення канонічної форми представлення предметної області. Ця канонічна форма будується з використанням структур, кожна з яких є парою класів з інтерпретацією «суб'єкт-об'єкт» зв'язаних між собою асоціативним відношенням, де пара розглядається, як аналог простого речення. Зв'язки цих структур між собою можна будувати шляхом введення нових типів відношень між класами, які будемо трактувати як відношення-посилання. Ці відношення іменуються шляхом введення підтипу відношення («який», «котрий», «хто», «що», «чий») та індексації зв'язків одного підтипу.

Отже, якщо один і той же клас зустрічається в різних парах, то в канонічну форму класової діаграми слід включати тільки одне описання класу, а для інших структур необхідно використовувати *stub* класу. Stub та клас пов'язуються відношенням-посилання. Імена таких посилань задаються як <підтип відношення>, <індекс>.

Канонічна форма дозволяє легко відновити за класовою діаграмою текст, що є важливим при організації людино-машинного діалогу при контролі знань студентів.

4.10 Розробка способів побудови логічних представлень контенту САН на базі UML-представлень навчального курсу

Методам логічного представлення природно-мовних текстових документів присвячено ряд робіт [137-139]. Серед найбільш провідних робіт можна вважати роботи Вавіленкової А.І. [138,139], яка запропонувала використовувати для цілей порівняння текстових документів логіко-лінгвістичну модель речення, що поєднує в собі як загальні властивості тексту і його складових частин, так і взаємозв'язки між ними.

Формально логіко-лінгвістична модель речення на природній мові описується за допомогою лінгвістичної та семантично-синтаксичної складових. Лінгвістична складова включає типи текстів, множину складних, синтаксичних частин тексту, текстову базу, що включає ключові слова та словосполучення, а також множину абзаців, що характеризуються типами зв'язків між реченнями, типами тематичних прогресій, множиною домінант в абзаці. Семантично-синтаксична складова розглядається як кон'юнкція логіко-лінгвістичних моделей речень $t' = \bigwedge_{g \in S} L_g(S_g)$. Тут $L_g(S_g)$ — логіко-лінгвістична модель речення (S_g). Логіко-лінгвістична модель речення має вигляд $L(S) = \bigwedge_{\mu=1}^{\vartheta(S)} L_\mu(S)$, де $L_\mu(S)$ — предикат, що описує частину речення, яке має закінчений зміст:

$$L_\mu(S) = P\big(x_1, c(x_1), x_2, c(x_2), x_3, z, \vartheta(p), w(z)\big), \quad (4.3)$$

де *X(S)* — множина сутностей, що входять в речення *S*;

X_1 — множина суб'єктів, що входять в речення *S*, $X_1 \subseteq X$;

x_1 — суб'єкт, що входить в речення S, $x_1 \in X_1$;

$c(x_1)$ — кортеж характеристик суб'єкта x_1;

X_2 або $X_2(x_1)$ — множина об'єктів, над якими виконується дія суб'єкта x_1;

x_2 — об'єкт, над яким виконується дія суб'єкта $x_1, x_2 \in X_2$;

$c(x_2)$ — кортеж характеристик об'єкта x_2;

X_3 або $X_3(x_1)$ — множина об'єктів, пов'язаних з суб'єктом x_1;

x_3 — об'єкт, пов'язаний з суб'єктом x_1, $x_3 \in X_3(x_1), x_1 \neq x_3$;

$P(x_1, x_2)$ — множина відношень між суб'єктом x_1 і об'єктом x_2, $x_1 \in X_1, x_2 \in X_2, x_1 \neq x_2$;

$z(x_1, x_2, p)$ — множина зв'язків між суб'єктом x_1 і об'єктом x_2 за відношенням p, $p \in P(x_1, x_2)$;

$c(z)$ — кортеж характеристик зв'язків p –го відношення між суб'єктом x_1 і об'єктом x_2;

$\vartheta(p)$ — параметри p-го відношення;

$w(z)$ — параметри z-го зв'язку p-го відношення між суб'єктом x_1 і об'єктом x_2, $p \in P(x_1, x_2), x_1 \in X_1, x_2 \in X(x_2), z \in (x_1, x_2, p)$.

Процесування тексту з метою отримання його логіко-лінгвістичної моделі значно спрощується, якщо виходити не із обробки тексту, а із вже отриманої з неї UML-діаграми предметної області. При наявності класової діаграми її предикати утворюються як аналоги класів, що мають структуру:

$$P_i(<назва>,<список\ атрибутних\ параметрів>) \quad (4.4)$$

або як аналоги відношень:

$$q_j(<назва\ сутності\ 1>, <назва\ сутності\ 2>, <тип\ зв'язку>, <назва\ відношення>, <направленість\ зв'язку>, <список\ параметрів>, <кардинальна\ характеристика>), \quad (4.5)$$

де P_i – предикат, що зв'язує клас та його атрибути,

q_j – предикат зв'язку двох сутностей в класовій діаграмі,

<*назва сутності 1*> — назва класу-суб'єкта при направляючому відношенні в діаграмі,

<*назва сутності 2*> — назва класу-об'єкта при направляючому відношенні в діаграмі,

<*тип зв'язку*> — може належати одному з типів відношень: {асоціація, узагальнення, агрегація, композиція},

<*назва відношення*> — для асоціативних зв'язків є кортежем, що містить назву асоціації та назву ролей; для неасоціативних зв'язків:

- агрегація: *А*+<індекс>;
- композиція: *К*+<індекс>;
- узагальнення: *У*+<індекс>;

<*направленість зв'язку*> — {0,1},

<*список параметрів*> — відображає атрибути класу з якими зв'язок має відношення залежності,

<*кардинальна характеристика*> — список чисел, що характеризують інформацію про кардинальні числа зв'язку.

Загальний вигляд предикату, що характеризує діаграму, має вигляд:

$$L(S) = P(S) \wedge Q(S), \quad (4.6)$$

де $P(S) = \bigwedge_{i=\overline{1,I_k}} P_i$ — предикати, які відповідають множині класів в класовій діаграмі;

P_i — предикат, який відповідає *i*-тому класу в і-тій діаграмі;

I_k — кількість класів в діаграмі;

$Q(S) = \bigwedge_{j=\overline{1,I_c}} q_j$, — предикати, які відповідають множині зв'язків в класовій діаграмі;

q_j — предикат, який відповідає *j*-тому зв'язку в класовій діаграмі;

I_c — кількість зв'язків в класовій діаграмі.

Загальними властивостями для предикатів є:

- наявність назви для кожної сутності діаграми;
- відсутність циклів в діаграмі за зв'язками типів узагальнення, агрегації або композиції;
- транзитивність зв'язку типу узагальнення за передачею атрибутів від суб'єкта до об'єкта.

ВИСНОВКИ

В даному томі відображено результати розроблення і використання методичних і інструментальних засад створення ІТ-засобів підтримки університетського центра підприємництва, а також відповідного веб-порталу. Ці розробки базуються на:

- аналізі досвіду партнерів за проектом CABRIOLET, а саме університетів і компаній Великобританії (Університет Ньюкаслу), Іспанії (компанія Inercia Digital), Італії (компанія Critiware), Португалії (Університет Коімбри) та Швеції (Університет КТН, Стокгольм) в процесі безпосереднього спілкування та різноманітних семінарів і зустрічей в цих країнах і в Україні;
- аналізі публікацій та досвіду провідних університетів і компаній інших країн.

При розробленні відповідних принципів, моделей і засобів використано джерела, у яких описується та аналізується такий досвід, а також досвід безпосередніх партнерів по проекту [140-165].

Веб-портал систематизує інформацію про майже 40 проектів кооперації університетів та індустрії, які вкнувалися за різними моделями. У четвертому томі описано тренінги, які забезпечують відповідну підготовку фахівців для використання запропонованих методичних та інструментальних засобів.

ЛІТЕРАТУРА

1. Leonardo da Vinci programme // Wikipedia. – https://en.wikipedia.org/wiki/Leonardo_da_Vinci_programme (Last 84access: 09.06.2015).
2. The Bologna Process // The Bologna Process – building a European Higher Education Area. ouncil of Europe. – http://www.coe.int/t/dg4/highereducation/EHEA2010/Default_en.asp (Last access: 05.02.2015).
3. The Copenhagen Declaration // Declaration of the European Ministers of Vocational Education and Training, and the European Commission, on enhanced European cooperation in vocational education and training. – Copenhagen, 2002. – http://ec.europa.eu/education/policy/vocational-policy/doc/copenhagen-declaration_en.pdf (Last access: 09.06.2015).
4. Lisbon Strategy for Growth and Jobs . – Brussels, 2006. – http://www.central2013.eu/fileadmin/user_upload/Downloads/ Tools_ Resources/Lisbon_MEMO-06-23_EN_1_.pdf (Last access: 09.06.2015).
5. Bologna Process // The European Higher Education Area. – http://www.ehea.info/ (Last access: 09.06.2015).
6. The role of the universities in the Europe of knowledge // Commission of the European Communities. – http://arhiva.tempus.ac.rs/uploads/documents/EC%20role%20of%20universities%20in%20the %20Europe%20of%20knowledge.pdf (Last access: 05.02.2003).
7. Digital Agenda Scoreboard 2013 // European Commission. – https://ec.europa.eu/digital-single-market/sites/digital-agenda/files/DAE%20SCOREBOARD%202013%20-%20SWD%202013%20217%20FINAL.pdf (Last access: 12.06.2013).
8. Here is how we will improve digital skills and create more jobs in Europe // Digital single market. – https://ec.europa.eu/digital-single-market/en/blog/here-how-we-will-improve-digital-skills-and-create-more-jobs-europe-0 (Last access: 08.03.2015).
9. Aldermanjan L. Unemployed in Europe Stymied by Lack of Technology Skills / L. Aldermanjan // The New York Times. – 2014. – 3 January. – P. B1.
10. Startup Ecosystem Whitepaper / Startup Commons. – http://www.startupcommons.org/download-documents.html (Last access: 18.03.2015).
11. Baptista R. Proximity to Knowledge Sources and the Location of Knowledge Based Start-ups / R. Baptista, J. Mendonça // IN+, Instituto Superior Técnico. – Lisbon: Technical University of Lisbon, 2015. – P.1–28.
12. Bouwman H. Business models dynamics for start–ups and innovating e-businesses / H. Bouwman, M. de Reuver, I. MacInnes // Int. J. Electronic Business. – 2007. – Vol. 7, N 3. – P. 269–286.

13. Porter M. E Clusters and the new economics of competition. Innovation and Entrepreneurship / Porter M.E. – Boston: Harvard Business Review, 2009. – P. 77– 90.
14. Носовский А. В. О подготовке специалистов в области снятия с эксплуатации атомных электрических станций / А. В. Носовский, М. В. Савельев // Ядерная и радиационная безопасность. – 2015. – № 4. – С. 58–63.
15. Chesbrough H. The role of the business model in capturing value from innovation: evidence from Xerox corporation's technology spin-off companies / H. Chesbrough, R. S. Rosenbloom // 'T', Industrial and Corporate Change. – 2002. – Vol. 11, N 3. – P. 529–555.
16. Morris M. The entrepeneur's business model: toward a unified perspective / M. Morris, M. Schindehutte, J. Allen. // Journal of Business Research. – 2005. – Vol. 58. – P. 726–735.
17. Tool-Based Support of University-Industry Cooperation in IT-Engineering / V. V. Lytvynov, V. S. Kharchenko, S. V. Lytvyn, M. V. Saveliev, E. V. Trunova, I. S. Skiter. – Chernihiv: Chernihiv National University of Technology, 2015. – 107 p.
18. Paulk M. C. Capability Maturity Model for Software, Version 1.1 / M. C. Paulk, B. Curtis, M. B. Chrissis, C. V. Weber. – Pittsburgh: Software Engineering Institute Carnegie Mellon University, 1993. – 82 p.
19. Bouwman H. Mobile Service Innovation and Business Models / H. Bouwman, T. Haaker, H. de Vos. – Berlin: Springer-Verlag Berlin Heidelberg, 2008. – 327 p.
20. CMMI for Development. – Pittsburgh: Carnegie Mellon University. – 2006. – 573 p.
21. Фатрелл Р. Т. Управление программными проектами: достижение оптимального качества при минимуме затрат / Фатрелл Р. Т., Шафер Д. Ф., Шафер Л.Н.; пер. с англ. – М.: Издательский дом «Вильямс», 2003. – 1136 с.
22. PMBOK Guide and Standards. – Pennsylvania: Project Management Institute, 2013. – 589 p.
23. Литвинов В. В. Архітектура знаннєорієнтованої автоматизованої системи навчання / В. В. Литвинов, І. С. Посадська, М. В. Савельєв // Технічні науки та технології. – 2016. – № 3 (5). – С. 122–130.
24. Модель компетенции как инструмент для оценки состояния IT-компаний в бизнес-центре университета / В. В. Литвинов, М. В. Савельев, И. С. Скитер, Е. В. Трунова // Математичні машини і системи. – 2015. – № 2. – С. 49 – 60.
25. Трунова О. В. Когнітивна модель компетенції ІТ-фахівця / О. В. Трунова, А. Г. Гребінник // Десята міжнар. наук.-практ. конф. «Математичне та імітаційне моделювання систем. МОДС'2015»: тези доповідей, (Чернігів-Жукин, 22–26 червня 2015.). – Чернігів: Чернігівський державний технологічний університет, 2015. – С. 20.
26. Advanced Spin-off model of UIC / O. Starov, V. Kharchenko, V. Sklyar [et al.] // University-Industry Interaction Conference, (Amsterdam, May 2013). – Amsterdam, 2013. – P. 90–97.

27. Савельєв М. В. Імітаційна модель процесу розробки ІТ-продукту силами університетської команди / М. В. Савельєв, А. В. Дерлеменко // Матеріали Міжнар. наук.-практ. Інтернет-конф. «Інформаційні технології: теорія, інновації, практика», (Полтава, 15 – 18 грудня 2015 р.). – Полтава: ПолтТНУ імені Ю. Кондратюка, 2015. – С. 105–107.

28. Хворостенко Р. Ю. Веб-портал університетського бізнес-центру підтримки новостворених ІТ-компаній / Р. Ю. Хворостенко // Междунар. науч.-практ. конф. «Інформаційна безпека та комп'ютерні технології», (Кировоград, 24–25 марта 2016 г.) – Кировоград, 2016. – С. 140–141.

29. Хворостенко Р. Ю. Веб-портал університетського бізнес-центру підтримки новостворених ІТ-компаній /Хворостенко Р. Ю. – Славутич:НТУУ«КПИ», 2016. – 75 с.

30. Beck K. Extreme Programming Explained: Embrace Change / Beck K. – Boston: Addison-Wesley, 1999. – 224 p.

31. Мартин Р. К. Быстрая разработка программ: принципы, примеры, практика / Мартин Р. К., Ньюкирк Д. В., Косс Р. С.; пер. с англ. – М.: Вильямс, 2004. – 744 с.

32. Alliance A. User Stories / A. Alliance. – https://www.agilealliance.org/glossary/user-stories/

33. Varun Grover, William J. Kettinger (2000). Process Think: Winning Perspectives for Business Change in the Information Age. p.168.

34. What is UML? / Unified Modeling Language. – http://www.uml.org/ (last access: 12.02.2016).

35. Менеджмент: учеб. пособ. – [2-е изд.]. – СПб.: Питер, 2014. – 352 с.

36. The Scrum Guide – http://www.scrumguides.org/scrum-guide.html (Last access: 23.06.2015).

37. Reenskaug T. M. H. MVC. XEROX PARC 1978-79 / T. M. H. Reenskaug. – http://heim.ifi.uio.no/~trygver/themes/mvc/mvc-index.html (last access: 11.07.2015).

38. Фаулер М. Шаблоны корпоративных приложений / М. Фраулер, Д. Райс, М. Фоммел [и др.]; пер. с англ. – М.: Вильямс, 2012. – 544 с.

39. Elman J. Lightweight Django / J. Elman, M. Lavin. – Sebastopol: O'Reilly Media, 2014. – 246 p.

40. Django. The web framework for perfectionists with deadlines . – https://www.djangoproject.com/ (last access: 10.03.2016).

41. ngixn [Електронийресурс]. – https://nginx.org/ru/

42. gunicorn [Електронийресурс]. – http://gunicorn.org/

43. PostgreSQL [Електронийресурс]. – https://www.postgresql.org/

44. Добров Г. М. Экспертные оценки в научно-техническом прогнозировании / Г. М. Добров, Ю. В. Ершов, Е. И. Левин, Л. П. Смирнов. – Киев: Наукова думка, 1974. – 160 с.

45. Бешелев С. Д. Экспертные оценки / С. Д. Бешелев, Ф. Г. Гурвич. – М.: Наука, 1973. – 161 с.
46. Dalkey N. An Experimental Application of the Delphi Method to the use of experts / N. Dalkey, O. Helmer // Management Science. – 1963. – N 9 (3). – P. 458–467.
47. Thurstone L. L. A law of comparative judgement / L. L. Thurstone // Psychological Review. – 1927. – N 34. – P. 278–286.
48. Саати Т. Принятие решений. Метод анализа иерархий / Саати Т.; пер. с англ. – М.: Радио и связь, 1993. – 278 с.
49. Кондратенко Ю. П. Моделі кооперації університетів та ІТ-компаній: системи прийняття рішень на нечіткій логіці / Ю. П. Кондратенко, Т.В. Кондратенко, Є.В. Сіденко, В.С. Харченко / Під ред. д.т.н., проф. Ю. П. Кондратенка. – Харків: Лисенко І. Б., 2015. – 132 с.
50. Gosenheimer C. Project Prioritization Guide. A structured approach to working on what matters most / C. Gosenheimer – Madison: University of Wisconsin, 2012. – 8 p.
51. Smith A. An introduction to Project Management [Електронийресурс] / A. Smith. – https://asmithatmancat.files.wordpress.com/2007/03/project-man-part-1.ppt (Last access: 02.12.2014).
52. Botchkarev A. Complexity in the context of information systems project management [Електронийресурс] / A. Botchkarev, P. Finnigan. – https://epress.lib.uts.edu.au/journals/index.php/opm/article/view/4272/4823 (Last access: 10.12.2014).
53. Copertari L. Selecting projects in a portfolio using risk and ranking / Copertari L. – Mexico: Computer Engineering Department, Autonomous University of Zacatecas, 2011. – 19 p.
54. A model of project complexity: distinguishing dimensions of complexity from severity: Proc. of the 9th International Research Network of Project Management Conference, (Berlin, 11–13 October, 2009). – Berlin, 2009. – 31 p.
55. Saaty T. L. Decision Making for Leaders: The Analytic Hierarchy Process for Decisions in a Complex World / Saaty T. L. – Pittsburgh, Pennsylvania: RWS Publications, 1990. – 292 p.
56. Саати Т. Принятие решений. Метод анализа иерархий / Саати Т.; пер. с англ. – М.: Радио и связь, 1993. – 278 с.
57. Миронова Н.А. Интеграция модификаций метода анализа иерархии для систем поддержки принятия групповых решений / Н.А. Миронова // Радіоелектроніка, інформатика, управління. – 2011. – № 2. – С. 47–54.
58. Aerts K. Critical role and screening practices of European business incubators / K. Aerts, P. Matthyssens, K. Vandenbempt // Technovation. – 2007. – N 27. – P. 14.
59. Ryzhonkov V. Typical Objectives and Mission statements of Business Incubation Program [Електронийресурс] / V. Ryzhonkov. – https://

worldbusinessincubation.wordpress.com/2013/04/05/typical-objectives-and-mission-statements-of-business-incubation-program/ (Last access: 05.04.2013).
60. Савельев М. В. Классификация моделей жизненного цикла разработки ИТ-продукта / М.В. Савельев // Одинадцята міжнар. наук.-практ. конф. «Математичне та імітаційне моделювання систем. МОДС'2016»: тези доповідей, (Жукин, 27 червня – 1 липня 2016 р.). – Жукин: Чернігівський державний технологічний університет, 2016. – С. 407–409.
61. Royce W. W. Managing the Development of Large Software Systems: Concepts and Techniques / W. W. Royce. – California: Proc. Wescon, 1970. – P. 328–338.
62. ГОСТ 34. Информационная технология. Сб. стандартов. Технические требования. – Введ. 1990-01-01. – М.: Изд-во стандартов, 1990. – 6 с.
63. ГОСТ 34.601-90. Информационная технология. Комплекс стандартов на автоматизированные системы. Автоматизированные системы. Стадии создания. Технические требования. – Введ. 1992-01-01. – М.: Изд-во стандартов, 1992. – 11 с.
64. ISO/IEC 12207:2008. Systems and software engineering – Software life cycle processes. – Geneva: IEEE Computer Society, 2008. – 138 p.
65. Fundamentals of the V-Modell // V-Modell® XT. – 2006. – ftp://ftp.heise.de/pub/ix/projektmanagement/vmodell/V-Modell-XT-Gesamt-Englisch-V1.3.pdf (Last access: 30.11.2014).
66. Boehm B. A. Spiral Model of Software Development and Enhancement / B. A. Boehm // IEEE Computer. – 1988. – N 21 (5). – P. 61–72.
67. Martin J. Rapid Application Development / Martin J. – Macmillan, 1991. – P. 81–90.
68. Microsoft Solutions Framework Process Model v.3.1. Microsoft Solutions Framework White paper. Microsoft Corp. – Redmond: Microsoft Corporation, 2002. – 47 p.
69. Jacobson I. The Unified Software Development Process / I. Jacobson, G. Booch. – Boston: Addison-Wesley Object Technology, 1999. – 512 p.
70. Савельев М. В. Выбор модели жизненного цикла проекта в области информационных технологий / М. В. Савельев // X міжнар. конф. «Управління проектами у розвитку суспільства»: тези доповідей, (Київ, 17 – 18 травня 2013 р.). – Київ, 2013. – С. 217–220.
71. Савельєв М. В. Проектне управління в АСУ: класика VS Agile. / М. В. Савельєв // Форум лідерів АСУ «Нова країна – Новий ландшафт». – Киев. – 2015 . – http://www.slideshare.net/APPAU_Ukraine/lead-asu-2015saveliev (Дата звернення 12.04.2015).
72. Humphrey W. S. Managing the software process / Humphrey W. S. – Pittsburgh: Addison-Wesley Publishing Company, 1999. – 494 p.
73. Hamming R. W. Error detecting and error correcting codes / Hamming R. W. // Bell System Technical Journal – 1950. – N 29 (2). – P. 147–160.

74. Malcolm D. G. Application of a Technique for Research and Development Program Evaluation / D. G. Malcolm, J. H. Roseboom // Operations Research. – 1959. – Vol. 7, N 5. – P. 646–669.

75. Архипенков С. Лекции по управлению программными проектами: конспект лекций / Архипенков С. – М.: Самиздат, 2003. – 222 с.

76. Томашевський В.М. Моделювання систем / Томашевський В. М. – К.: Видавнича група BHV, 2005. – 352 с.

77. Software Engineering Body of Knowledge, SWEBOK V3 / IEEE computer society. – http://www.computer.org/web/swebok/v3 (Last access: 12.03.2015).

78. Saveliev M. V. The simulation model of IT-product (service) development by a «start-up» company growing inside an academic institution / M. V. Saveliev, V. V. Lytvynov // Математичні машини і системи. – 2015. – № 4. – С. 92–99.

79. Савельев М. В. О сложности одной задачи комбинаторной оптимизации / М. В. Савельев // Математичні машини і системи. – 2016. – № 4. – С. 106–110.

80. Tadao M. Petri Nets: Properties, Analysis and Applications / M. Tadao // Proc. of the IEEE. – 1986. – Vol. 77, N 4. – P. 541–558.

81. Стеценко, ІннаВячеславівна. Петрі-об'єктне моделювання систем : автореф. дис. .д-ратехн. наук : 05.13.06 / Стеценко Інна Вячеславівна ; Нац. акад. наук України, М-воосвіти і науки, молоді та спорту України, Міжнар. наук.-навч. центр інформ. технологій та систем. - К., 2012. - 36 с.

82. Stetsenko I.V. Petri-Object Simulation: Software Package and Complexity / I.Stetsenko, V.Dorosh, A.Dyfuchyn // Proceedings of the 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2015). – Warsaw (Poland), 2015. – P. 381-385.

83. Stetsenko I.V. State Equations of Stochastic Timed Petri Nets with Informational Relations // Cybernetics and systems analysis Volume 48, Number 5(2012), 784-797.

84. Башкин В. А. Некоторые методы ресурсного анализа сетей Петри / Башкин В. А. – Ярославль: Ярославский государственный университет им. П. Г. Демидова, 2014. – 244 с.

85. Wilensky U. NetLogo itself // NetLogo. Center for Connected Learning and Computer-Based Modeling, Northwestern University. Evanston, IL. – http://ccl.northwestern.edu/netlogo/ (last access: 04.05.2015).

86. CMMI® or Agile: Why Not Embrace Both! / H. Glazer, J. Dalton, D. Anderson, M. Konrad, S. Shrum // Technical note CMU/SEI-2008-TN-003. – Baltimore: Software Engineering Institute, 2008. – 48 p.

87. CMMI for Services. – Pittsburgh: Carnegie Mellon University. – 2010. – 520 p.

88. CMMI Institute . – http://cmmiinstitute.com/ (Last access: 23.06.2015).

89. Appraisal Requirements for CMMI Version 1.3. – Pittsburgh: Carnegie Mellon University. – 2011. – 33 p.
90. Standard CMMI Appraisal Method for Process Improvement (SCAMPI) A, Version 1.3: Method Definition Document . – Pittsburgh: Carnegie Mellon University. – 2011. – 276 p.
91. Grzegorzewski P. Metrics and orders in space of fuzzy numbers / P. Grzegorzewski // Fuzzy Sets and Systems. – 1998. – Vol. 97, Issue 1. – P. 83–94.
92. Кондратенко Ю. П. Особливості синтезу і моделювання ієрархічно-організованих СППР на основі нечіткої логіки / Ю. П. Кондратенко, Є. В. Сіденко // Вестник «Херсонского национального технического университета». – 2011. – № 2 (41). – С. 150–158.
93. Zadeh L. A. Fuzzy sets / L. A. Zadeh // An International journal in information Science and Engineering. – 1965. – Vol. 8, N 3. – P. 338–353.
94. Штовба С. Д. Введение в теорию нечетких множеств и нечеткую логику - http://matlab.exponenta.ru/fuzzylogic/book1.php (Дата обращения: 07.11.2014).
95. Dubois D. Operations on fuzzy numbers / D. Dubois, H. Prade // Int. J. Syst. Sci. 9. – 1978. – N 4. – P. 613–626.
96. Saveliev M. The model of IT-startup that grows in university ecosystem and approach to assess its maturity / M. Saveliev, V. Lytvynov // Information Models and Analyses. – 2016. – Vol. 5, N 3. – P. 236 – 245.
97. Джордж, Ф. Основы кибернетики: Пер. с англ. / Под ред. А. Л. Горелика. — М.: Радио и связь, 1984. — 272 с.
98. Кибернетика и проблемы обучения: Сб. переводов / Ред. и предисл. А. И. Берга. — М.: Прогресс, 1970. — 389 с.
99. Применение ЭВМ в учебном процессе // Сб. докладов научно-технич. семинара под ред. А. И. Берга. — М.: Сов. радио, 1969. — 248 с.
100. Программированное обучение и обучающие машины // Труды научно-технич. семинара. — Киев: Вып. 2, 1967.
101. Бирюков, В. В. Программированное обучение автокоду «Инженер» с использованием многопультовой системы / В кн.: Теория и применение математических машин / Под ред. А. М. Оранского, Н. Н. Поснова. — Мн.: Изд-во БГУ, 1972. — С. 213–216.
102. Uttal, W. R. On conversational interaction. / W. R. Uttal // “Programmed Learning and Computer Based Instruction”. — New York, Wiley, 1962.
103. Основы компьютерной грамотности / Е. И. Машбиц, Л. П. Бабенко, Л. В. Верник и др.; под ред. А. А. Стогния и др. — К. : Вища шк., 1988. — 215 с.

104. Licklider, J. Preliminary experiments in computer-aided teaching / J. Licklider // “Programmed Learning and Computer Based Instruction”. — New York, Wiley, 1962. — P. 217–239.
105. Buiten, R., Lane, H. S. Experimental system gives language student instant error feedback / R. Buiten, H. S. Lane // Digital Equipment Corporation Computer Application Note, 1965.
106. Домрачев, В. Г. О классификации образовательных информационных технологий / В. Г. Домрачев, И. В. Ретинская. // Информационные технологии. — 1996. — № 2. — С. 10–13.
107. Воронцов, А. Модели обучения автоматизированных обучающих систем [Электронный ресурс]. - http://wiki.itorum.ru/ 2011/05/modeli-obucheniya-avtomatizirovannyx-buchayushhix-sistem/ — 19.10.2015.
108. Аттель, У. Обучающая вычислительная машина: моделирование в истинном масштабе времени обучающего диалога / В сб. «Кибернетика и проблемы обучения» / Ред. и предисл. А. И. Берга. — М.: Прогресс, 1970. — С. 206–228.
109. Кривошеев, А. О. Проблемы оценки качества программных средств учебного назначения. // Сб. докладов 1-го научно-практического семинара «Оценка качества программных средств учебного назначения». — М.: Гуманитарий, 1995. — С. 5–12.
110. Обучающие машины, системы и комплексы: Справочник / Под ред. А. Я. Савельева. — К.: Вища шк., 1986. — 303 с.
111. Савельев, А. Я. Автоматизированные обучающие системы на базе ЭВМ. Вып.1. / А. Я. Савельев — М.: Знание, 1977. — 36 с.
112. Петрушин, В. А. Экспертно-обучающие системы / В. А. Петрушин. — Киев: Наукова думка, 1991. — 196 с.
113. Соловов, А. В. Проектирование компьютерных систем учебного назначения: Учебное пособие / А. В. Соловов. — Самара: СГАУ, 1995. — 137 с.
114. Кривицкий, Б. Х. О систематизации учебных компьютерных средств. - http://ifets.ieee.org/russian/depository/v3_i3/html/3.html — 19.10.2015.
115. Савінов, О. М. Перспективні напрямки автоматизації навчання / О. М. Савінов. — Збірник наукових праць ЖВІ НАУ. — № 3. — 2010. — С.165–170.
116. Цибульский, Г. М. Модели обучения автоматизированных обучающих систем / Г. М. Цибульский, Е. И. Герасимова, В. В. Ерошин // Системотехника: Сетевой электронный научный журнал. — 2004. — № 2. — С. 53–162.
117. Шустов, С. Б. Теорія ресурсів і ресурсні кризи: минуле, сьогодення і майбутнє: аналітичний огляд, навчальний посібник. - Нижній Новгород, 2009. — С. 144–163.
118. Кликов, Ю. І. Банки даних для прийняття рішень / Ю. І. Кликов, Л. Н. Горьков. — М .: Сов. радіо, 1980. — 208 с.

119. Жук, К. Д. Исследование структур и моделирование логико-динамических систем: моногр. / К. Д. Жук, А. А. Тимченко, Т. И. Доленко. — Киев: Наукова думка. — 1975. — 199 с.
120. Скрэгг, Г. Семантические сети как модели памяти. // Новое в зарубежной лингвистике. — М.: Радуга, 1983. — Вып. 12. — С. 228–271.
121. Минский, М. Фреймы для представления знаний / М. Минский. — М.: Мир, 1979. — 152 с.
122. Субботін, С. О. Подання й обробка знань у системах штучного інтелекту та підтримки прийняття рішень: Навчальний посібник / С. О. Субботін. — Запоріжжя: ЗНТУ, 2008. — 341 с.
123. Искусственный интеллект. Кн. 2: Модели и методы: Справочник / Под ред. Э. В. Попова. — М.: Радио и связь, 1990. — 303 с.
124. Глушков, В. М. Абстрактная теория автоматов / В. М. Глушков // Успехи математических наук. — 1961. — № 6 (101). — С. 3–62.
125. Королюк, В. С. Процессы марковского восстановления в задачах надежности систем / В. С. Королюк, А. Ф. Турбин. — Киев : Наук, думка, 1982. — 236 с.
126. Бусленко, Н. П. Лекции по теории сложных систем / Н. П. Бусленко, В. В. Калашников, И. Н. Коваленко. — Москва: Сов.радио, 1973. — 440 с.
127. Питерсон Дж. Теория сетей Петри и моделирование систем / Дж. Питерсон. — Москва: Мир, 1984. — 264 с.
128. Любченко, В. В. Модели знаний для предметных областей учебных курсов / В. В. Любченко // Искусственный интеллект. — 2008. — № 4. — С. 458–462.
129. Batini Carlo. Conceptual database design: an entity-relationship approach / Carlo Batini, Stefano Ceri, Shamkant B. Navathe. — The Benjamin/Cummings Publishing Company, Inc. 1992. — 490 p.
130. Douglass, Bruce Powel. Real-Time UML. Second Edition. Developing Efficient Objects for Embedded Systems / Bruce Powel Douglass. — Wesley, 1999. — 238 p.
131. Соммервилл И. Инженерия программного обеспечения / И. Соммервилл. — Москва: Вильямс, 2002. — 624 с.
132. Литвинов Виталий. Проектирование формализованного представления предметной области исходя из ее текстово-графического описания / Виталий Литвинов, Ирина Посадская // Технічні науки та технології: науковий журнал / Черніг.нац.технол.ун-т. — Чернігів, 2016. — № 4(6). — С. 107–119.
133. Lytvynov Vitalii. Knowledge representation in the automated learning systems [Text] /Vitalii Lytvynov, Iryna Posadska // International Journal «Information Technologies & Knowledge» Volume 9, Number 1 — 2015. — P. 34–43.

134. Посадская Ирина. Архитектура базы знаний автоматизированной системы обучения / Ирина Посадская // Технічні науки та технології: науковий журнал / Черніг.нац.технол.ун-т. — Чернігів. — 2017. — № 1 (7). — С. 154–161.
135. Литвинов В.В. Об'єктно-орієнтоване моделювання при проектуванні вбудованих систем і систем реального часу / В. В. Литвинов, С. В. Голуб, К. М. Григор'єв, В. Ю. Жигульська. — Черкаси: Черкаський національний університет ім. Б. Хмельницького, 2011. — 379 с.
136. Голенко Д. И. Статистические методы сетевого планирования и управления. - М.: Наука, 1968. — 196 с.
137. Кобозева, И. М. Лингвистическая семантика - М.: Эдиториал УРСС, 2000. - 352 с.
138. Вавіленкова А.І. Логіко-лінгвістичні моделі речень як засіб порівняння текстових документів за змістом. // Математичні машини і системи. — 2012. — № 1. — С. 166–173.
139. Вавіленкова, А. І. Теоретичні основи аналізу електронних текстів / А. І. Вавіленкова, Д. В. Ланде, О. Є. Литвиненко. — К.: НАУ, 2014. — 250 с.
140. KTH spin-off boosts industry cooperation https://www.kth.se/en/ees/nyheterochpress/annualreport/yearbook-2015/kth-spin-off-boosts-industry-cooperation-1.665452
141. Bridge builders between academia and society https://www.kth.se/en/ees/nyheterochpress/annualreport/yearbook-2015/bridge-builders-between-academia-and-society-1.643199
142. Collaboration with companies https://www.kth.se/en/ece/2.48522/samverkanskampanj-1.437497
143. New perspectives on internationalization and competitiveness integrating economics, innovation and higher education, Ullberg, Eskil (Ed.) , Springer International Publishing Switzerland 2015, 185 p.
144. New innovation centre to promote collaboration between academia and industry, https://www.akademiskahus.se/en/news/news-room/2013/12/new-innovation-centre-to-promote-collaboration-between-academia-and-industry/
145. Anders Broström, Firms' rationales for interaction with research universities and the principles for public co-funding, The Journal of Technology Transfer, June 2012, Volume 37, Issue 3, pp 313–329, https://link.springer.com/article/10.1007/s10961-010-9177-4
146. Folke Valfrid Snickars, Ulf Karlsson, Research infrastructure, networks of science and regional development – the case of Oskarshamn, REGION, The journal of ERSA, Volume 4, Number 3, 2017, 119–13 http://openjournals.wu.ac.at/ojs/index.php/region/article/view/143/205

147. Linda Assbring, Cali Nuur, What's in it for industry? A case study on collaborative doctoral education in Sweden, Industry and Higher Education, Volume: 31 issue: 3, page(s): 184-194, http://journals.sagepub.com/doi/pdf/10.1177/0950422217705245
148. Partnerships for Innovation and Socio-Economic Impact: The Entrepreneurial University. Report of proceedings, European Commission, Royal Institute of Technology, 63 p. http://ec.europa.eu/dgs/education_culture/repository/education/tools/docs/ub-forum-stockholm_en.pdf
149. Prospectus and future tasks of universities. Digitalization – Internationalization – Differentiation, Austrian Council for research and technology development, Austrian Council for Research and Technology, 2017, 413 p.
150. Marie Magnell, Anette Kolmos, Employability and work-related learning activities in higher education: how strategies differ across academic environments, Tertiary Education and Management Journal, Volume 23, 2017 - Issue 2, European Higher Education Society and Taylor & Francis pp. 103-114.
151. J.P.C. Marques, J.M.G. Carac, H. Diz, How can university-industry-government interactions change the innovation scenario in Portugal? - The case of the University of Coimbra, The International Journal of Technological Innovation, Entrepreneurship and Technology Management, Elsevier Ltd 26 (2006) 534–542, http://www.sciencedirect.com/science/article/pii/S0166497205000751
152. Website of Instituto de Pedro Nunes, https://www.ipn.pt/
153. Addison, J. T., Teixeira, P., Evers, K., Bellmann, L. Collective Bargaining and Innovation in Germany. Industrial Relations, 2017, 56(1), 73–121.
154. Simões, M., Andrade, J. S., & Duarte, A. Differences in human capital and openness to trade as barriers to growth and convergence in the EU. In N. da C. Cabral, J. R. Gonçalves, & N. C. Rodrigues (Eds.), The Euro and the Crisis: Perspectives for the Eurozone as a Monetary and Budgetary Union (pp. 73–94). Springer.
155. UIIN Good Practice Series 2013 – Fostering University-Industry Relationships, Entrepreneurial Universities and Collaborative Innovation Arno Meerman & Thorsten Kliewe (eds.), University Industry Innovation Network, 131 p.
156. Website of European Commission, section "University Business Cooperation" http://ec.europa.eu/education/policy/higher-education/university-business-cooperation_en
157. The State of European University-Business Cooperation, Todd Davey, Thomas Baaken, Victoria Galan Muros, Arno Meerman, Europe Science-to-Business Marketing Research Centre, Münster University of Applied Sciences, Germany, 140 p. http://ec.europa.eu/dgs/education_culture/repository/education/tools/docs/uni-business-cooperation_en.pdf

158. 5th University-Business Forum on Strategic Partnerships for Innovation and Growth: From Dialogue to Partnerships, Final report, Rebecca Allinson, Flora Giarracca, Zsuzsa Jávorka, Xavier Potau, European Commission, Brussels, 4-5 June 2013, 38 p. http://ec.europa.eu/dgs/education_culture/repository/education/tools/docs/ubforum-5_en.pdf
159. 6th University-Business Forum, Forum report, Rebecca Allinson, Zsuzsa Jávorka, Adam Krčál, Xavier Potau, Brussels, European Commission, 5-6 March 2015, 77 p. http://ec.europa.eu/dgs/education_culture/repository/education/tools/docs/university-business-forum-brussels_en.pdf
160. 7th University-Business Forum University-Business Cooperation - For Innovation And Modernisation, Forum report, Christine Bertram, Janna Puukka, Michael Blakemore, Angeli Jeyarajah, 6 - 7 April 2017 The Square, Meeting Centre, Mont des Arts-Kunstberg, Brussels, 105 p. https://ec.europa.eu/education/sites/education/files/university-business-forum-2017-report_en.pdf
161. 30 best case studies of good practice in the area of UBC within Europe, Todd Davey, Michael Deery, Clive Winters, Peter van der Sijde, Tomasz Kusio, Silvia Rodríguez Sedano, Europe Science-to-Business Marketing Research Centre, Münster University of Applied Sciences, Germany, 184 p. https://www.ub-cooperation.eu/pdf/casestudyreport.pdf
162. Grubicka, J.; Matuska, E. 2015. Sustainable entrepreneurship in conditions of UN (Safety) and technological convergence, Entrepreneurship and Sustainability Issues 2(4): 188–197. http://jssidoi.org/jesi/uploads/articles/8/Grubicka_Sustainable_entrepreneurship_in_conditions_of_UN_Safety_and_technological_convergence.pdf
163. Cristina De Castro, Barbara Mavì Masini, Management of Group Evolution Through Cooperative Work in E/M-learning Systems, Consumer Electronics Times Apr. 2014, Vol. 3 Iss. 2, The world academic publishing Co., Limited, Hong Kong, pp. 220-232
164. R. Inklaar, M. O'Mahony, and M. Timmer, "ICT and Europe's Productivity Performance: Industry-Level Growth Account Comparisons with the United States", Review of Income and Wealth, vol. 51, pp. 505–536, Dec. 2005.
165. Emerging Modes of Cooperation between Private Enterprises and Organizations, The International report of EMCOSU project, Mateja Melnik, Tomas Pusnik, Samo Pavlin, University of Ljubljana, 116 p. http://www.emcosu.eu/static/uploaded/files/outcomes/Att5.8.3_EMCOSU_International_report_final.pdf

ДОДАТОК А. КЕРІВНИЦТВО КОРИСТУВАЧА ПОРТАЛУ «UNIVERSITY-INDUSTRY COOPERATION»

А.1. Цільове призначення порталу

Веб портал призначений для систематизації, опису та аналізу практик (кейсів) університетсько-індустріальної кооперації, які отримано за підтримки європейської програми TEMPUS, а саме в рамках проекту CABRIOLET «Model-Oriented ApproaCh And Intelligent Knowledge-Based System for Evolvable Academia-IndustRy CooperatIOn in ELEctronic and CompuTer Engineering».
Керівництво користувача охоплює питання з використання та адміністрування порталу «University-Industry Cooperation» з метою отримання даних та динамічного управління інформаційним наповненням порталу.
Наповненням порталу можуть займатися тільки користувачі з правами адміністратора.

А.2. Структура порталу

Портал «Університети-бізнес кооперація» має ієрархічну структуру веб-сторінок, які об'єднані як за змістом, так і за навігацією. Основними сторінками порталу, які відповідають структурі головного меню, є «Головна», «Кейси», «Інструменти», «Про нас», «Гайд», «Увійти» (рис.1).

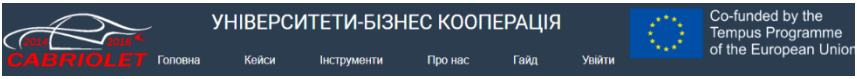


Рис.1. Головне меню порталу «Університети-бізнес кооперація»

Головна сторінка порталу

Містить головне меню системи, посилання на офіційні сайти проекту CABRIOLET та програму Tempus, а також основну та контактну інформацію про розробників порталу.

Сторінка «Кейси»

Забезпечує перегляд як загального списку усіх внесених до системи варіантів кооперації, так і вибірку зі списку за певними характеристиками (рис.2). Користувач може відстежувати скільки кейсів знаходиться на сторінці загалом або за результатами

фільтрування. Список кейсів містить по кожному кейсу назву кейсу, коротку анотацію та посилання «Детальніше» яке відкриває сторінку з обраним для перегляду кейсом.

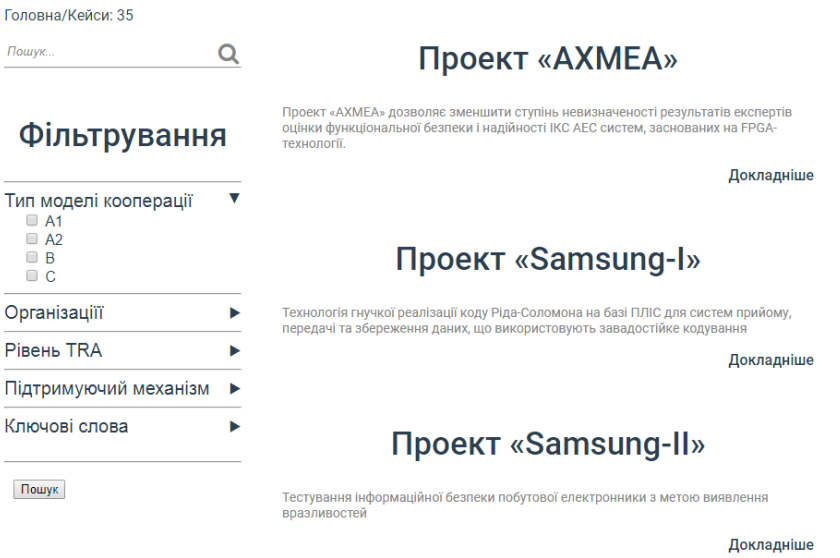


Рис.2. Фрагмент сторінки «Кейси»

У верхньому лівому куті даної сторінки знаходиться поле «Пошук…», використовуючи яке можна знайти кейс за назвою або її частиною, якщо користувач не впевнений в повній назві.

Нижче поля «Пошук…» розташовано блок «Фільтрування», який можна застосувати до списку кейсів. Фільтрація можлива за такими признаками:

- типом моделі кооперації;
- задіяними організаціями;
- досягнутим рівнем TRA;
- використаним підтримуючим механізмом;
- ключовими словами.

Для здійснення фільтрації користувач повинен зазначити певні значення класифікаторів в маркерних блоках, вибрати необхідну строку з випадаючого списку та натиснути кнопку «Пошук» унизу блоку «Фільтрування». За результатами цих дій буде виведено список кейсів, що задовольняють обраним критеріям, та вказана їх загальна кількість.

Переглянути повну інформацію за конкретним кейсом можна за умова натискання на назву або кнопку «Детальніше…» відповідного кейсу. Кожному кейсу відповідає окрема сторінка, яка має структуру, відповідну шаблону опису кейсів, наведеному у п. А.3.

Сторінка «Інструменти»

Надає користувачу можливість переглянути основні інструменти, використані в проекті, та здійснити перехід на відповідний обраному інструменту сайт для ознайомлення.

Сторінка «Про нас»

Інформує користувача про всіх учасників проекту.

Сторінка «Гайд»

Містить дану інструкцію користувача, яка описує можливості використання порталу.

Сторінка «Увійти»

Містить поля для вводу логіну та паролю користувача з правами адміністратора. Реєстрація даних користувачів відбувається адміністратором порталу по запиту.

А.3. Правила введення кейсів

При введенні кейсів слід керуватися шаблоном, наведеним у таблиці 1.

Таблиця 1. Шаблон опису кейсів

Компонент шаблону (укр.)	Компонент шаблону (англ.)	Зміст компонента шаблону
ЗАГАЛЬНА ІНФОРМАЦІЯ	GENERAL INFORMATION	
Назва кейсу	Title of the case	Вкажіть назву, наприклад, Програма, Проект, Ініціатива….
Заголовок для «продажу»	Sales pitch	Вкажіть короткий опис вашого методу для залучення

		уваги читача
Організація (ї)	Organization (s)	Вкажіть назви залучених організацій
Країна / Країни	Country / countries	Вкажіть назви залучених країн
Дата	Date	Вкажіть дату написання документу
Автор(и)	Author(s)	Вкажіть ім’я/імена автора(ів)
Види взаємодії	Nature of interaction	Вкажіть один або декілька методів реалізації UBC, які використовувалися
Механізм підтримки	Supporting mechanism	Будь ласка, оберіть один або декілька механізмів, що використовувалися: - Стратегічний інструмент - Структурний інструмент або підхід - Операційна діяльність - Локальні умови
ПРОФІЛЬ КЕЙСУ	CASE STUDY PROFILE	
Резюме	Summary	Надайте короткий огляд дослідження
Обгрунтування	Background	Опишіть загальну ситуацію
Цілі	Objectives	Якими були основні об'єктивні або бажані результати від цієї ініціативи?
Відповідальність	Responsibility	Хто був відповідальним за реалізацію дій та заходів в даному випадку?
РЕАЛІЗАЦІЯ ТА ФІНАНСУВАННЯ	IMPLEMENTATION & FUNDING	
Стратегія та вжиті заходи	Strategy &activities undertaken	Опишіть кейс в термінах стратегії та дій, що виконані в підтримку стратегії
Моніторинг та оцінювання	Monitoring and evaluation	Які були вжиті заходи для моніторингу та оцінювання?

Заходи для сталого розвитку	Sustainability measures	Які здійснюються заходи / кроки для забезпечення сталого розвитку в перспективі?
Вартість	Costs	Що було/є основним джерелом витрат, включаючи персонал/обладнання/ресурси ? Поясніть, наскільки це можливо, структуру ціни в грошовому і/або часовому еквіваленті
Фінансування	Funding	Який забезпечувався вид фінансування і з якого джерела? Поясніть, наскільки це можливо, відсоткове співвідношення джерел фінансування і необхідних обсягів
РЕЗУЛЬТАТИ ТА ВПЛИВ	OUTCOMES & IMPACT	
Результати	Outcomes	Які конкретні отримані результати і/або виконані роботи?
Наслідки	Impacts	Як вплинула (переваги або недоліки) програма на зацікавлених осіб як прямих, так і непрямих? По можливості, прохання надати статистичні дані або приклади і окремо короткостроковий і довгостроковий вплив
Зацікавлені особи та одержувачі вигоди	Involved stakeholders and beneficiaries	Які зацікавлені сторони виграють і в чому їх вигода? В якій мірі ситуація інтегрована в місцеву інноваційну систему?

Нагороди / визнання	Awards / recognition	Чи мав кейс будь-яке визнання у вигляді нагород або визнання третіми особами?
ОТРИМАНІ УРОКИ	LESSONS LEARNED	
Основні виклики	Primary challenges	Які були основні виклики і перешкоди при реалізації кейсу та чи є вони наразі?
Фактори успіху	Success factors	Які ключові фактори сприяли успіху?
Розповсюджуваність	Transferability	Наскільки поширюваним і можливим для застосування для інших є кейс, як саме та кому він може бути корисним?
ДОДАТКОВА ІНФОРМАЦІЯ	FURTHER INFORMATION	
Публікації / статті	Publications / articles	Публічно доступні публікації та статті, які надають додаткові подробиці
Посилання	Links	Публічно доступні посилання на веб-сайти, які надають додаткові подробиці
Ключові слова	Keywords	Які основні ключові терміни?
Контактна інформація	Public contact details	Контактні дані для публічного доступу, де можна отримати додаткову інформацію (організація, департамент, адреса, телефон, адреса електронної пошти, веб-сайт)
ДОДАТКОВІ РЕЗУЛЬТАТИ, ОТРИМАНІ В ПРОЕКТІ TEMPUS CABRIOLET	FURTHER OUTCOMES AS PER PROJECT	
Модель UIC	UIC model in	Яка саме модель UIC

відповідності з КАБРИОЛЕТ-класифікатором	accordance with CABRIOLET-classifier	застосовувалась відповідно до наступної класифікації: - модель A1 - «кафедра як освітній і тренінговий центр»; - модель A2 - «кафедра як тренінгово-сертифікаційний центр»; - модель B - «кафедра як науково-дослідний та інноваційний центр»; - модель C - «кафедра як центр підприємництва»
Верифікація моделі UIC	UIC model verification	Які існують підтвердження того, що була реалізована саме обрана модель?
Зміна рівня готовності технології (TRL) за результатами проекту	Changing of Technology Readiness Level (TRL) as a result of the project	Як змінився TRL в результаті виконання проекту відповідно до 9-рівневої моделі Technology Readiness Assessment (TRA)?
Оцінка успішності проекту	Project success assessment	Оцініть успішність проекту за шкалою від 1 до 10 (або за шкалою ВИСОКА, СЕРЕДНЯ, НИЗЬКА)
Короткий опис UIC кейсу в табличному вигляді (обов'язковий додаток)	Brief UIC case description in a table view	Представте короткий опис UIC кейсу у вигляді даної таблиці
Інші додатки	Additional annexes	Будь-яка релевантна інформація, представлена у вигляді додатків

Примітки:

1. При заповненні поля Організація (Organization) для залучених організацій необхідно вказати їх класифікаційні ознаки відповідно до наступної таксономії:

- вуз (U) або науково-дослідна організація (A);

- розмір індустріальної компанії: мале і середнє підприємство (SME) - до 250 чоловік, велика локальна компанія (LC) - більше 250 чол. і має основниймайданчик в одному регіоні, глобальна компанія (GL) - має майданчики в різних регіонах;
- профіль індустріальної IT-компанії: розробка програмних продуктів (SW-P), аутсорсинг розробки програмних продуктів (SW-O), системна інтеграції (SYS), надання сервісів (IT-S).

2. При заповненні поля Оцінка успішності проекту (Project success assessment) доцільно враховувати такі напрямки:
- інноваційні та творчі результати (наскільки проект був цікавим і вагомим в науковому плані і перспективним в плані просування в бізнесі);
- технічні і технологічні результати (наскільки вагомим проект був в прикладному відношенні);
- організаційні результати (наскільки добре був проект організований і керований);
- фінансові результати (наскільки проект був ефективний з точки зору вкладень і отриманих результатів).

А4. Адміністрування порталу «Університети-бізнес кооперація»

Для будь-яких дій зі зміни наповнення порталу необхідно увійти до панелі адміністрування за допомогою пункту головного меню «Увійти» використовуючи унікальний логін та пароль користувача з правами адміністратора.

Створення кейса

На панелі адміністрування користувачу з правами адміністратора необхідно:
1. Ввести основну інформацію про проект: назву, авторів, країни, мови та дату (рис.3).
2. Заповнити усі поля, що повністю описують проект відповідно до шаблону, наведеному у п. 3.5.3 (рис.4).
3. Відмітити теги, за якими можна буде знаходити кейс після занесення у розділі «Фільтрування», у списку тегів, який можна побачити натиснувши напис «Оберіть категорію та теги» (рис.3).
4. Натиснути кнопку «Створити кейс», що знаходиться нижче заповнених полів (рис.4).
5. Кейс створено!

Рис.3. Створення кейсу: основна інформація, вибір тегів

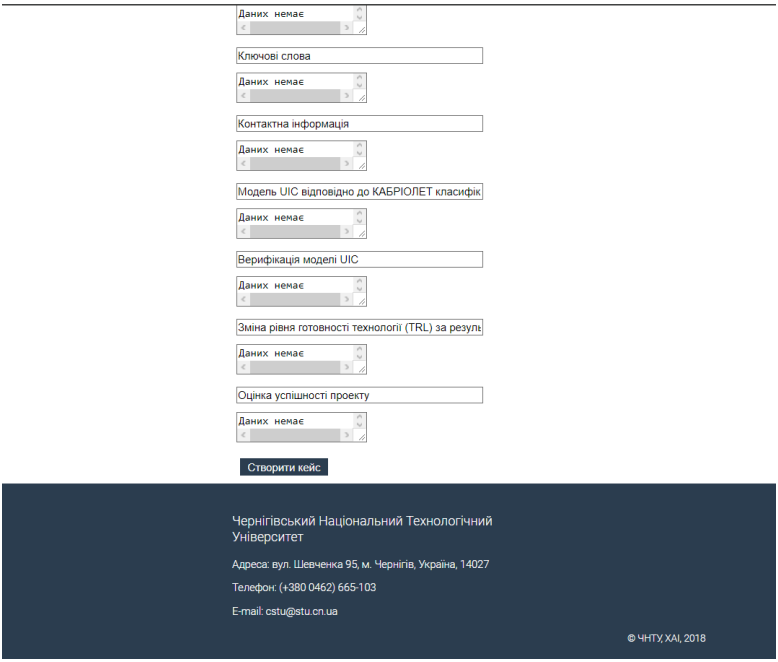


Рис.4. Створення кейсу: поля шаблону, кнопка «Створити кейс»

Видалення кейса

На панелі адміністрування користувачу з правами адміністратора необхідно (рис.5):
1. Обрати кейс зі списку кейсів натиснувши на кнопку «Список кейсів».
2. Ввести id - унікальний код проекту.
3. Натиснути кнопку «Видалити».
4. Кейс видалено!

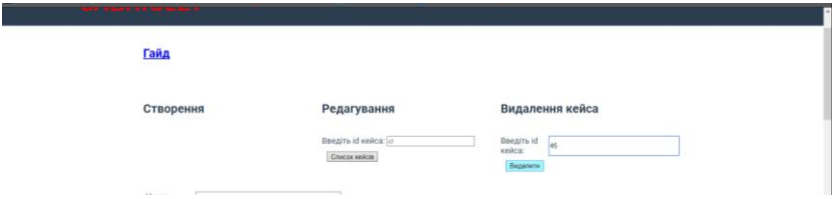


Рис.5. Видалення кейсу

Редагування кейса

На панелі адміністрування користувачу з правами адміністратора необхідно (рис.6):
1. Обрати кейс зі списку кейсів натиснувши на кнопку «Список кейсів».
2. Ввести id - унікальний код проекту.
3. Натиснути кнопку «Редагування».
4. Відредагувати потрібні поля.
5. Відмітити або додати відповідні теги зі списку тегів, який відкривається натисканням напису «Оберіть категорію та теги».
6. Натиснути кнопку «Редагувати кейс»
7. Кейс відредаговано!

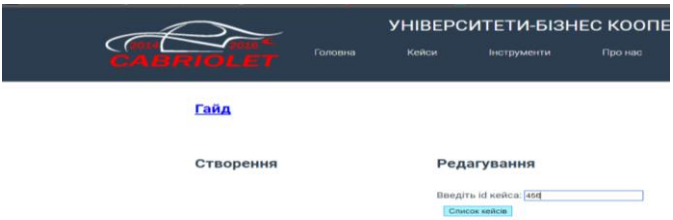


Рис.6. Редагування кейсу

Створення тегу

Теги призначені для пошуку кейсів після занесення у розділі «Фільтрування». Вони згруповані за категоріями, ідентифікатори яких перелічені в розділі «Додати тег до категорії» (рис.7). Для додавання тегу необхідно:
1. Ввести id категорії тегу.
2. Ввести значення нового тегу.
3. Натиснути кнопку «Додати тег».
4. Тег додано!

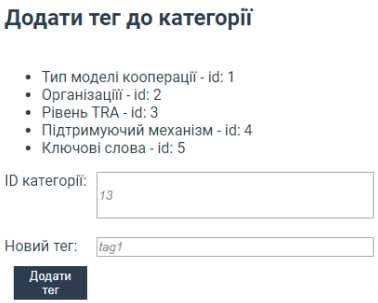


Рис.7. Додавання тегу

ABSTRACT

UDC 378:004

Lytvynov V. V., Kazymyr V. V., Posadskaya I. S., Saveliev M. V., Grebennik A. G., Illiashenko O. O., Kharchenko V. S. / Kharchenko V. S. (Editor). University industry cooperation. Web portal. Operational roadmap. Volume 3. – Ministry of Education and Science of Ukraine, Chernihiv National University of technology, National Aerospace university "KhAI", 2017. - 181 p.

The theoretical bases and principals of development and implementation of tools for support of university and IT industry companies cooperation as well as for creation and evolvement of startups are presented. The stages of development and features of the functioning of the communication web-portal are described. Recommendations on practical use in implementation of different models of cooperation are given. The book materials have been prepared according TEMPUS CABRIOLET project, "Model-Oriented Approach and Intelligent Knowledge-Based System for Evolvable Academia-Industri Cooperative in ELEctronic and CompuTer Engineering" (544497-TEMPUS-1-2013-1-UK-TEMPUS-JPHES).

For university students studying in the areas of information technology and starting practical activities, university specialists and IT industry, which will take part in organizing and increasing the efficiency of university-industrial cooperation, as well as for teachers who have classes on appropriate courses.

Bibliography – 165 titles, drawings – 22, tables – 35.

CONTENTS

ЗМІСТ

Литвинов Віталій Васильович
Казимир Володимир Вікторович
Харченко Вячеслав Сергійович
та інші

УНІВЕРСИТЕТСЬКО-ІНДУСТРІАЛЬНА КООПЕРАЦІЯ. ВЕБ-ПОРТАЛ. НАСТАНОВА З ВИКОРИСТАННЯ (українсько мовою)

Том 3

Під редакцією
Харченка В.С.

Комп'ютерна верстка
Ілляшенко О.О.

Зв. план, 2017
Підписаний до друку 7.12.2017 Формат 60x84 1/16.
Папір офс.. Офс. друк. Гарнітура Times New Roman
Обл.вид. 9,50. Умов. друк. арк. 8,84. Наклад 175 прим.
Замовлення 3/12/17

Національний аерокосмічний університет ім. М. Є. Жуковського
"Харківський авіаційний інститут"
61070, Харків-70, вул. Чкалова, 17 **http://www.khai.edu**

Випускаючий редактор: ФОП Голембовська О.О.

Видавець: ТОВ «Видавництво «Юстон»
01034, м. Київ, вул.. О. Гончара, 36-а,
тел.: +38 044 360 22 66 **www.yuston.com.ua**

Свідоцтво про внесення суб'єкта видавничої справи до державного реєстру видавців, виготовлювачів і розповсюджувачів видавничої продукції серіяДК № 497 від 09.09.2015 р.