Keywords: *foreign language professional competence, professional training, multimedia technologies, authentic language environment.*

DOI: https://doi.org/10.31392/NZ-udu-159.2025.08
УДК 378 147:004. 92

Малежик М. П., Малежик П. М., Майданюк І. В.

АНАЛІЗ ПІДХОДІВ ДЛЯ ПІДГОТОВКИ ФАХІВЦІВ З ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У статті розглянуто актуальну проблему формування підходів до підготовки фахівців з інженерії програмного забезпечення. Ця проблема ставить виклики перед вищою освітою і потребує аналізу та перегляду педагогічних підходів та технологій з метою підвищення ефективності підготовки ІТ-фахівців з цього напряму.

Показано, що підготовка з інженерії програмного забезпечення має формувати насамперед компетентності, які є характерними для всіх інженерів з програмного забезпечення в їх професійній діяльності.

Метою дослідження було вивчити та проаналізувати підходи до підготовки фахівців з інженерії програмного забезпечення та подальшим виробленням базових рекомендацій для розробників та викладачів курсів, що вивчаються фахівцями даного напряму. Сформульовано рекомендації, що містять важливі висновки з аналізу багатьох освітньо-професійних програм (ОПП) та можуть бути корисними для використання при розробці ОПП та викладачів основних курсів даного напряму підготовки.

Слід зазначити, що на дослідження і практику в галузі інженерії програмного забезпечення впливають як її базові положення в інформатиці, так і її становлення в якості самостійної інженерної дисципліни. Оскільки переважна частина досліджень у галузі програмної інженерії провадяться на факультетах чи кафедрах інформатики, то відповідно навчальні програми з інженерії програмного забезпечення також, розробляються як на факультетах, так і на навчальних кафедрах. Отже, ця дисципліна може розглядатися як інженерна галузь, що має тісніші зв'язки з інформатикою, ніж інші інженерні галузі.

Таким чином, дослідження еволюції та аналіз різних підходів до підготовки фахівців з інженерії програмного забезпечення надало можливість сформулювати базові рекомендації для розробників та викладачів курсів, які вивчаються при підготовці фахівців з даного напряму.

Ключові слова: *інженерія програмного забезпечення, компетентнісний підхід, освітньо-професійна програма, підготовка ІТ-фахівців, рекомендації розробникам, навчальний план, синергетичний ефект, проектно-орієнтовані курси.*

Вирішення проблеми підготовки фахівців з інженерії програмного забезпечення є актуальним та перспективним напрямом, що ставить серйозні виклики перед закладом вищої освіти та вимагає ретельного вибору педагогічних технологій та дослідження впливу від їх використання.

Інженерія програмного забезпечення є по суті інженерною спеціальністю. Підготовка з даної спеціальності має включати характеристики, які є бажаними для всіх програмних інженерів в їх професійній діяльності. До них віднесемо

такі:

– приймати ряд рішень, ретельно оцінюючи альтернативи та вибирати для рішення такий підхід, що є оптимальним в даній задачі із врахуванням існуючого контенту;

– в міру можливостей, працювати з використанням вимірюваних кількісних характеристик, вдосконалювати чи уточнювати існуючі методи вимірювань та за необхідності будувати наближені розв'язки на основі досвіду та емпіричних даних;

– надавати особливого значення використанню дисциплінованого процесу при здійсненні проекту і розуміти важливість питань ефективної організації командної роботи;

– відповідати за виконання широкого спектру задач, починаючи з досліджень, розроблення, проектування, виробництва, тестування, впровадження, експлуатації та управління, продажами, консультуванням та навчанням;

– у процесі виконання своїх обов'язків широко використовувати інструментальні засоби, оскільки, вибір та використання відповідних засобів є також важливим питанням;

– об'єднуватися в професійні спільноти, сприяти розвитку своєї галузі шляхом розробки та впровадження рекомендацій, стандартів, атестаційних принципів, дієвих підходів.

Проблема критичного впливу якості і вартості програмного забезпечення на суспільство сьогодення робить особливо важливим питання побудови навчальних планів з інженерії програмного забезпечення, на відміну від інших інженерних програм. Оскільки, орієнтація на особистісні та професійні інтереси студентів є одним із пріоритетних завдань вищої освіти України, питання вибору і формування змісту курсів для підготовки майбутніх фахівців з інженерії програмного забезпечення є на часі.

Одним із попередніх документів, що мав об'єднуюче значення для освіти з комп'ютерних наук була модель навчального плану ACM (Association for Computing Machinery), що явилося результатом роботи значної кількості фахівців і груп, які сприяли становленню інформатики та інформаційних технологій, завдяки чому десятки тисяч випускників університетів стали професійними розробниками програмного забезпечення [1-6]. У роботі [7] проведено дослідження щодо стану освітніх ресурсів для вивчення окремих питань інженерії програмного забезпечення, а також цілісних онлайн програм для підготовки фахівців даної галузі. Значна увага, приділена методичним аспектам навчання базових технічних дисциплін майбутніх IT-фахівців у контексті міждисциплінарного підходу в роботі [8]. Модель системи формування технічної компетентності майбутніх фахівців з інформаційних технологій в умовах проектного навчання наводиться в дослідженні [9].

На сьогодні, в Україні діючим є стандарт вищої освіти за спеціальністю 121 «Інженерія програмного забезпечення» для першого (бакалаврського) рівня освіти, затверджений Міністерством освіти і науки України 29 жовтня

2018 року [10].

Метою даного дослідження було дослідити різні підходи до підготовки фахівців з інженерії програмного забезпечення та виробити базові рекомендації для розробників та викладачів курсів, що вивчаються при підготовці фахівців з даного напряму.

Якщо питання «що викладати з інженерії програмного забезпечення» є вивченим і стандартизованим [10], то питання «як викладати» потребує серйозного дослідження. Власний педагогічний досвід і аналіз науково-методичної літератури дали нам змогу виокремити основні рекомендації для розробників навчальних планів із інженерії програмного забезпечення та викладачів окремих курсів:

1. Розробники навчальних планів і викладачі мають володіти достатніми знаннями та досвідом у відповідних галузях і розуміти характерні властивості програмної інженерії. До цього віднесемо наявність знань з програмної інженерії в більшості її галузей, достатній досвід практичної роботи, визнання власних знань у сфері програмної інженерії, наявність публікацій науково-методичних праць, постійне ознайомлення з новими предметними областями, наявність мотивації для підтримки високої обізнаності про останні розробки з даної дисципліни.

2. Розробники навчальних планів і викладачі мають мислити в термінах кінцевого результату. Результати навчання повинні враховуватися і при викладанні окремих курсів. Спрямованість на кінцевий результат допоможе викладачу розміщувати в навчальних планах тільки релевантні матеріали та забезпечить його викладання належним чином з відповідним рівнем деталізації.

3. Розробники навчальних планів мають підтримувати баланс між повнотою матеріалу та гнучкістю побудови курсів, надаючи викладачам простір для інновацій. В цілому не слід допускати зниження рівня вивчення поглибленого технічного чи математичного матеріалу, оскільки вивчення даного матеріалу на робочому місці буде затрудженим.

4. Більшість ключових концепцій, принципів, і проблем програмної інженерії необхідно викладати як наскрізні теми протягом всього навчального плану для формування у студентів інженерного складу мислення. Деякі теми програмної інженерії доцільно розподіляти по різним курсам навчального плану. Ранні курси послуговують вступом до теми, а наступні курси закріплюють та розширяють отримані студентами знання. Тут необхідно, також, передбачити курси, що забезпечують поглиблене вивчення даної теми.

5. Вивчення деяких тем потребує певного рівня підготовки, тому вони мають викладатися ближче до завершення навчального плану. Отже, необхідно передбачити викладання матеріалів, які сприяють формуванню зрілості у студентів на більш ранніх стадіях. Рекомендується прочитати вступні курси якомога раніше в процесі навчання, проте в той же час, відкласти викладання основної частини матеріалу якомога пізніше. На початковому етапі

навчання студентів необхідно, також знайомити з «складними» задачами програмної інженерії. Прикладами таких задач можуть бути задачі, пов'язані зі змінними вимогами, задачі розуміння і зміни існуючих великих систем, і робота в великій групі та ін. Це дає змогу розвинути у студентів розуміння значущості таких областей знань, як процес, якість, еволюція та менеджмент до початку вивчення цих областей.

6. Студенти мають вивчити якусь певну предметну область, яка не відноситься до програмної інженерії. Оскільки переважна частина діяльності в галузі програмної інженерії, пов'язана з пошуком рішень для задач в конкретних предметних областях, що не відносяться до програмної інженерії, то в навчальному плані необхідно передбачити вивчення студентами однієї або кількох «зовнішніх» предметних областей. Вивчення даного матеріалу не тільки надасть студентам знання, які вони зможуть застосувати до вирішення задач програмної інженерії, але й навчить їх термінології і процесам, притаманним даній предметній області.

7. Студенти мають отримати певні навички, що виходять за межі програмної інженерії. Такі навички набуваються переважно на практиці. Це*:* критичне мислення – набуття необхідних знань, навичок аналізу та методів ефективної оцінки; оцінка і відстоювання отриманих знань – студенти повинні розуміти обмеженість сучасних знань з програмної інженерії і напрямки, в яких ці знання мають розвиватися; усвідомлення власних обмежень – студент має знати, що професіонали часто вдаються до консультування в інших професіоналів, а також усвідомити переваги командної роботи; ефективна комунікація – студенти повинні навчитися ефективно обмінюватися інформацією різними способами, письмово, при проведенні презентацій, демонстрацій власних і чужих програмних розробок, а також під час різних обговорень; етична і професійна поведінка – студенти мають навчитися контролювати відповідність своєї діяльності нормам етики, конфіденційності і безпеки.

Студенти можуть засвоїти дані навички більш ефективно при їх постійному використанні в групових проектах, письмових роботах і студентських презентаціях.

8. Студентам необхідно прищепити прагнення до навчання і саморозвитку. Оскільки більша частина того, що вивчається – зміниться протягом майбутньої професійної діяльності, то вкрай важливо, щоб студенти розвивали в собі навички постійного розширення власних знань.

9. Програмна інженерія повинна викладатися, як дисципліна, яка орієнтується на вирішення проблем. Усвідомлення даного факту допоможе студентам раціонально підходити до процесу навчання, полегшить розуміння матеріалу і допоможе оцінити відповідність матеріалу цілям курсу. Помилковим є фокусування винятково на технічних проблемах, що в подальшому призводить до створення непридатних до використання систем. Навички вирішення проблем найкращим чином розвиваються при аналізі прикладів і розв'язку навчальних задач під час практичних занять.

Демонстрація розв'язку на екрані викладачем являється необхідною, але недостатньою умовою розвитку необхідних навичок, тому студентам потрібно давати значну кількість практичних задач для самостійного розв'язання.

10. Найбільшу увагу у навчанні необхідно звертати на основоположні і незмінні принципи програмної інженерії, а не на інформацію про новітні чи спеціалізовані засоби та інструменти. Застосовуючи цю рекомендацію до навчання мов програмування необхідно відзначити, що границя між постійними і змінними знаннями є такою, що визначається досить складно і тому, може змінюватися. Без сумнівів, що програмні інженери мають детально вивчити декілька мов програмування, а також інших типів (наприклад мови специфікацій). Тобто при вивченні цих мов, студенти повинні засвоїти значно більше, ніж поверхневий синтаксис, щоб наступне ознайомлення з новими мовами потребувало мінімальних зусиль. Для технологій (як апаратних, так і програмних), дана рекомендація означає, що немає необхідності детально запам'ятовувати програмний інтерфейс додатків (API), систему команд конкретної комп'ютерної системи або користувацький інтерфейс просто, щоб знати їх на пам'ять. Замість цього студенти повинні розвинути в собі навички пошуку необхідної інформації в довідковій літературі та системних керівництвах, сконцентровуючись при цьому на важливіших задачах.

11. Навчання повинно здійснювати з використанням відповідних сучасних засобів, навіть якщо дані засоби не є ціллю навчання. Для студентів має бути звичним процес вибору різноманітних засобів, із такою звичкою, вони повинні розпочинати свою професійну діяльність. Таку звичку важко здобути на робочому місці, коли необхідність якнайшвидшого надання результатів часто ускладнюють вивчення нових засобів. Необхідно ретельно добирати найбільш придатні для навчання засоби. Надто складний, ненадійний, дорого вартісний та такий, що важко опановується в необхідний термін і в даній аудиторії, на даних ресурсах і надає дуже мало переваг від вивчення, засіб, буде непридатним як для навчання, так і для використання в професійній діяльності. Багато засобів програмної інженерії виявилися невдалими, оскільки вони не відповідали вказаним критеріям.

12. Матеріал, що викладається в рамках навчального плану з програмної інженерії повинен бути пов'язаний з серйозними дослідженнями і мати строго математичне або інше теоретичне обґрунтування або ж являтися загально прийнятою професійною практикою. Необхідно мати підтвердження того, що знання, які викладаються – відповідають дійсності і можуть бути застосовані. Таким підтвердженням може бути перевірена математична теорія або ж загально прийняті практичні принципи, які широко використовуються. В ситуаціях, коли матеріал, який викладається, ґрунтується на, переважно, загальноприйнятих принципах, але не підлягав раніше науковій перевірці, необхідно чітко обумовлювати, що матеріал не є беззаперечним.

13. Навчальний план обґрунтовується на реальному практичному досвіді. Навчальна програма повинна бути побудована таким, щоб, як мінімум, включати наступне: навчальні приклади; проектно-орієнтовані курси;

дипломний проект; практичні завдання, досвід роботи студентів. Тут, викладачі повинні пам'ятати, що виробничий досвід, який можуть отримати студенти під час навчання – обмежений. Студенти зможуть оцінити складність роботи і наслідки недосконалої роботи тільки отримавши реальні результати своєї діяльності в різноманітних проектах протягом професійної діяльності. Викладачі можуть надати лише початкову допомогу студентам в осягненні реалій життя і повинні усвідомлювати, що навчання студентів розумінню справжнього змісту отриманих знань є складною задачею.

14. Необхідно якомога частіше порушувати питання, пов'язані з моральними, юридичними і економічними аспектами майбутньої діяльності студентів, а також розкривати поняття того, що значить бути професіоналом у вибраній ними галузі. Однією з основним задач кожної професії є дотримання етичних і професійних норм її представниками. При обговоренні цих принципів протягом всього навчання вони глибше вкорінюються в свідомості студентів. Одним із можливих методів досягнення цього є знайомство студентів з відповідними стандартами і рекомендаціями.

15. Щоб забезпечити розуміння студентами певних важливих ідей, необхідно мотивувати їй цікавими конкретними і переконливими прикладами. Приклади повинні бути достатнього об'єму і складності, щоб показати, що застосування на практиці принципів, які вивчаються, несуть у собі очевидні переваги, а небажання використовувати вказані принципи – призводить до негативних результатів. Сфери, вивчення яких особливо потребують мотивації: основи математики; процес і якість; людський фактор і зручність використання програмного забезпечення.

16. На сьогодні навчання програмної інженерії вже виходить за межі лекційного формату, тому необхідно звернути увагу на важливість навчання різним підходам до викладання та отримання знань. Найбільш поширений підхід до викладання програмної інженерії – лекційний, що доповнюється лабораторними роботами, індивідуальною роботою з викладачем та ін. При цьому використання альтернативних підходів може сприяти більш ефективному навчанню студентів. Деякі підходи можуть бути застосовані, як додаток або, навіть, заміна значної частини навчання в лекційному форматі. Це такі альтернативні підходи: проблемно-орієнтоване навчання – цей підхід доводить свою результативність і в інших професійних областях і зараз застосовується певними закладами для навчання програмної інженерії; навчання «точно в термін» – навчання фундаментальним основам дисципліни безпосередньо перед навчанням можливостей застосування отриманих знань або навчання математичних методам за день до навчання їх застосування в контексті програмної інженерії. Цей підхід важко реалізувати, оскільки в цьому випадку виникають проблеми координації навчальних курсів; навчання на труднощах – студентам дають складне для вирішення завдання. Потім навчають методам, що дозволяють у майбутньому легко розв'язувати задачі подібного типу; розробка матеріалів для самонавчання, що опрацьовуються студентами самостійно – сюди входять онлайн-навчання і

навчання з використанням різних комп'ютерних та мобільних апаратних засобів.

17. Ефективність навчання і забезпечення синергетичного ефекту можуть бути досягненні при проектуванні навчального плану таким чином, щоб студенти отримали декілька типів знань одночасно. Навчальний план, як правило, містить у собі велику кількість матеріалів. І навпаки, на вивчення багатьох тем виділяється невелика кількість годин. Проте, якщо правильно побудувати навчальний план, велика кількість тем може вивчатись одночасно. Наприклад, приклади застосування методу сумісного навчання: моделювання, мови і нотації – глибина розуміння особливостей таких мов, як UML досягається шляхом її використання при вивченні інших концепцій. Те ж саме стосується формальних методів і програмування. Необхідно окремо виділити деякий час для вивчення основ мов і підходів моделювання, але подальше розширення і поглиблення знань студентами може досягатися при вивченні багатьох інших тем; процес, якість і менеджмент – студентів можна навчати прямуванню певних процесів при роботі над задачами і проектами, основною ціллю яких є навчання іншим концепціям, при цьому бажано ознайомити студентів з процесом, щоб вони розуміли, чому їм пропонують використати той, а не інший процес. При цьому також бажано простежити розв'язок даної задачі чи реалізацію проекту, обговорюючи одночасно корисність застосування даного процесу. З великою долею ймовірності подібний підхід допоможе вам досягти значної глибини розуміння матеріалу студентами, при цьому час, який подібне навчання відбере у основного матеріалу буде незначним; математика – можна знайти масу можливостей поглибленого вивчення логіки або інших напрямків дискретної математики; одночасне вивчення декількох тем – допомагає студентам встановити взаємозв'язки між дисциплінами і може посилити їх інтерес до вивчення матеріалу.

18. Навчальний план і курси, що входять до нього повинні регулярно переглядатися і оновлюватися. Навчальні заклади і викладачі повинні регулярно переглядати свої курси і програми та змінювати їх при необхідності. Ця рекомендація відноситься до навчального плану чи курсів, що самостійно розробляються ЗВО або викладачами.

Висновки. На дослідження і практику в галузі інженерії програмного забезпечення впливають як її базові положення в інформатиці, так і її становлення в якості самостійної інженерної дисципліни. Переважна частина досліджень в галузі програмної інженерії провадяться на факультетах чи кафедрах інформатики або комп'ютерної техніки. Відповідно, навчальні програми з інженерії програмного забезпечення розробляються як на факультетах, так і на навчальних кафедрах. Отже, ця дисципліна може розглядатися як інженерна галузь, що має тісніші зв'язки з інформатикою, ніж інші інженерні галузі.

Таким чином, дослідження еволюції та аналіз різних підходів до підготовки фахівців з інженерії програмного забезпечення надало можливість

сформулювати базові рекомендації для розробників та викладачів курсів, які вивчаються при підготовці фахівців з цього напряму.

Використана література:

1. Henry M. Walker and G. Michael Schneider. A revised model curriculum for a fiberal arts degree in computer science. Communications of the ACM, 39(12):8595, Desember 1996.
2. Сейдаметова З. С. Підготовка магістрів в ІТ-галузі *Науковий часопис НПУ імені М. П. Драгоманова. Серія № 2. «Комп'ютерно-орієнтовані системи навчання» :* [зб. наук. праць]. Київ : НПУ ім. М. П. Драгоманова, 2012. № 12 (19). С. 48–53.
3. Hoganson K. Non-traditional graduate CS program integrated with distance technojogy. *43rd ACM Southeast Conference*, Mach 18-20, 2005, v. 1. P. 324-328.
4. White L. J., Coffey J. The design and implementation of an innovative online program for a master of science degree in Computer Science – Software Engineering specialization. *CSEET '11 Proceedings of the 2011 24th IEEE-CS Conference on SE Education and Nraining, IEEE-CS.* Washington, DC, USA, 2011. P. 257-265.
5. Wendt Kevin. Audience and Content Areas of Online Software Engineering Education and Training: A Systematic Review Proceedings of the 52nd Hawaii International Conference on System Science, 2019 [Electronic Resource]. URL : https://scholarspace.manoa.hawaii.edu/items/d385d8e9-781e-48fb-b94d-271c7ff50d50
6. Giovana Giardini Borges, Rogéria Cristiane Gratão de Souza Skills development for software engineers: Systematic literature review, Volume 168, April 2024]. URL : https://www.sciencedirect.com /journal/information-and-software-technology/vol/168/suppl/C
7. Fundamental Approaches to Software Engineering 27th International Conference, FASE 2024 Held as Part of the European Joint Conferences on Theory and ractice of Software, ETAPS 2024 Luxembourg City, Luxembourg, April 6–11, 2024. URL : https://link.springer.com/book/10.1007/978-3-031-57259-3
8. Малежик П. М. Методичні аспекти навчання базових технічних дисциплін майбутніх ІТ-фахівців в контексті міждисциплінарного підходу. *Наукові записки. Серія: Педагогічні науки.* Кропивницький : РВВ ЦДПУ ім. В. Винниченка, 2019. Вип. 177. Ч. 1. С. 227–231.
9. Малежик П. М., Малежик М. П. Модель системи формування технічної компетентності майбутніх фахівців з інформаційних технологій в умовах проектного навчання. *Актуальні проблеми неперервної освіти в інформаційному суспільстві :* тези Міжнародної науково-практичної конференції з Інтернет підтримкою присвячених 185-річчю Національного педагогічного університету імені М. П. Драгоманова, 29-30 травня 2020 року. Київ, 2020. С. 347-350.
10. Стандарт вищої освіти за спеціальністю 121 «Інженерія програмного забезпечення» для першого (бакалаврського) рівня освіти. URL : https://mon.gov.ua/static-objects/mon/sites/1/vishcha-osvita /zatverdzeni%20standarty/12/21/121-inzhener.programn.zabezp.bakalavr-1.pdf

References:

1. Henry M. Walker and G. Michael Schneider (1996). A revised model curriculum for a fiberal arts degree in computer science. Communications of the ACM, 39(12):8595, Desember [in Englian].
2. Seydametova Z. S. (2012). Pidgotovka magistriv v IT-galuzi [Master's degree training in the IT field]. *Naukovyi chasopys NPU imeni M.P. Dragomanova. Seria № 2. «Komp'uterno-orientovani systemy navchanya» :* [zb. nauk. prac']. K. : NPU im. M. P. Dragovanova. № 12 (19). S. 48–53 [In Ukrainian].
3. Hoganson K. (2005). Non-traditional graduate CS program integrated with distance technojogy. *43rd ACM Southeast Conference*, Mach 18-20, v. 1. P. 324-328 [in Englian].
4. White L. J., Coffey J. (2011). The design and implementation of an innovative online program for a master of science degree in Computer Science – Software Engineering specialization. *CSEET '11 Proceedings of the 2011 24th IEEE-CS Conference on SE Education and Nraining, IEEE-CS.* Washington, DC, USA,. P. 257-265 [in Englian].
5. Wendt Kevin. Audience and Content Areas of Online Software Engineering Education and Training: A Systematic Review Proceedings of the 52nd Hawaii International Conference on System Science, 2019 [Electronic Resource]. URL : https://scholarspace.manoa.hawaii.edu/items/d385d8e9-781e-48fb-b94d-271c7ff50d50 [in Englian].

6. Giovana Giardini Borges, Rogéria Cristiane Gratão de Souza Skills development for software engineers: Systematic literature review, Volume 168, April 2024]. URL : https://www.sciencedirect.com/journal /information-and-software-technology/vol/168/suppl/C [in Englian].
7. Fundamental Approaches to Software Engineering 27th International Conference, FASE 2024 Held as Part of the European Joint Conferences on Theory and ractice of Software, ETAPS 2024 Luxembourg City, Luxembourg, April 6–11, 2024. URL : https://link.springer.com/book/10.1007/978-3-031-57259-3 [in Englian].
8. Malezhyk P. M. (2019). Metodychni aspekty navchanya bazovyh tekhnichnyh dyscyplin maybutnih IT-fahivciv v konteksti mizhdyscyplinarnogo pidhodu [Methodological aspects of teaching basic technical disciplines to future IT specialists in the context of an interdisciplinary approach]. *Naukovi zazypsku. Seria: Pedagogichni nauky.* Kropyvnyc'kyi : RVV CDPU im. V. Vynnychenka. Vyp. 177. CH. 1. C. 227–231 [in Ukrainian].
9. Malezhyk P. M., Malezhyk M. P. (2020). Model' systemy formuvanya technichnoji kompetentnosti maybutnih fahivciv z informaciynyh tekhnologiy v umovah proektnogo navchanya [Model of the system for forming technical competence of future information technology specialists in project-based learning]. *Aktual'ni problemy neperervnoji osvitu v informaciynomu suspil'stvi :* tezy Mizhnarodnoi naukovo-praktychnoi konferenciji z internet pidtrymkou prusvjacheniy 185-richu Nacional'nogo pedagogichnogo universytetu imeni M. P. Dragomanova, 29-30 travnya 2020 roky. Kyiv. S. 347-350 [in Ukrainian].
10. Standart vyshoji osvity za special'nistju 121 «Inzheneriya programnogo zabezpechenya» dya pershogo (bakalavrs'kogo) rivnya osvity [Standard of higher education in specialty 121 «Software Engineering» for the first (bachelor's) level of education]. URL : https://mon.gov.ua/static-objects/mon/sites/1/vishcha-osvita/zatverdzeni%20standarty/12/21/121-inzhener.programn.zabezp.bakalavr-1.pdf [in Ukrainian].

M. Malezhyk, P. Malezhyk, I. Maidanyuk. Analysis of approaches to training software engineering specialists.

The article considers the current problem of developing approaches to training software engineering specialists. This problem poses challenges for higher education and requires the analysis and revision of pedagogical approaches and technologies in order to increase the effectiveness of training IT specialists in this area. It is shown that training in software engineering should form, first of all, competencies that are characteristic of all software engineers in their professional activities. The purpose of the study was to study and analyze approaches to training software engineering specialists and to further develop basic recommendations for developers and teachers of courses studied by specialists in this area. Recommendations were formulated that contain important conclusions from the analysis of many educational and professional programs (EPP) and can be useful for use in the development of EPPs and teachers of basic courses in this area of training. It should be noted that research and practice in the field of software engineering are influenced by both its basic provisions in computer science and its emergence as an independent engineering discipline. Since the majority of research in the field of software engineering is conducted at faculties or departments of computer science, software engineering curricula are also developed at faculties and departments. Therefore, this discipline can be considered as an engineering field that has closer ties to computer science than other engineering fields. Thus, the study of the evolution and analysis of different approaches to training software engineering specialists has made it possible to formulate basic recommendations for developers and teachers of courses that are studied in the training of specialists in this area.

Keywords: *software engineering, competency-based approach, educational and professional program, training of IT specialists, recommendations for developers, curriculum, synergistic effect, project-oriented courses.*