УДК 004

Д.Правдюк, магістр гр. КІ-22МЗ
Центральноукраїнський національний технічний університет

ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ АВТОМАТИЗОВАНОГО РЕІНЖИРІНГУ

У статті розроблено програмне забезпечення, яке призначено для системи автоматизованого реінжирінгу. Метою розробки є дослідження та програмна реалізація системи автоматизованого реінжирінгу. Об'єктом дослідження є процес автоматизованого реінжирінгу. Предметом дослідження є методи автоматизованого реінжирінгу. Методи дослідження базуються на методах інженерії програмного забезпечення, методах математичної статистики, методах розробки програмного забезпечення. Результат роботи – програмна реалізація системи автоматизованого реінжирінгу. В процесі роботи над програмною моделлю виконано аналіз існуючих апаратних та програмних засобів. В повній мірі описані всі компоненти розробленого програмного забезпечення.

Постановка проблеми. Система призначена для реінжинірингу коду застарілого програмного забезпечення, який був реалізований на мові Pascal, на сучасні мови програмування, у тому числі й на Delphi 10. Реінжиніринг програмного забезпечення (англ. software reengineering) – повторна реалізація успадкованої системи з метою підвищення зручності її використання, супроводження, розширення функціоналу чи впровадження оновлених технологій. У сфері технологій, що постійно розвивається, програмне забезпечення панує безперечно. Але що станеться, коли ваше колись передове програмне забезпечення почне втрачати свій блиск? Навіть якщо ви розробили найкраще програмне забезпечення свого часу, завжди є місце для вдосконалення. Ласкаво просимо до захоплюючої подорожі реінжинірингу програмного забезпечення – процесу, який дозволяє воскресити, вдосконалити та революціонізувати існуюче програмне забезпечення.

Реінжиніринг програмного забезпечення – це не просто розкіш; це необхідність у сучасному динамічному діловому середовищі. Оскільки технології розвиваються, очікування користувачів зростають, а вимоги ринку змінюються, випередження стає найважливішим. Застосовуючи реінжиніринг програмного забезпечення, ви можете підвищити продуктивність, масштабованість і ефективність, гарантуючи, що ваше програмне забезпечення буде конкурентоспроможним і відповідатиме мінливим галузевим стандартам.

Аналіз останніх досліджень і публікацій. При аналізі останніх досліджень і публікацій [1-20] було виявлено певні прогалини у забезпеченні системи автоматизованого реінжирінгу**.**

Мета й завдання дослідження. Метою роботи є дослідження та програмна реалізація системи автоматизованого реінжирінгу.

Для досягнення поставленої мети визначена програма дослідження, що складається з наступних завдань:

- Огляд існуючих систем автоматизованого реінжирінгу.
- Дослідження системи автоматизованого реінжирінгу.
- Програмна реалізація системи автоматизованого реінжирінгу.

Об'єктом дослідження є процес автоматизованого реінжирінгу.

Предметом дослідження є методи автоматизованого реінжирінгу.

Методи дослідження базуються на методах інженерії програмного забезпечення, методах математичної статистики, методах розробки програмного забезпечення.

Виклад основного матеріалу. Розпізнавання ознак, які вказують на необхідність реінжинірингу програмного забезпечення, має вирішальне значення для того, щоб ваше

програмне забезпечення залишалося ефективним, надійним і відповідало цілям вашої організації.

Реінжиніринг застарілого програмного забезпечення або проблеми сумісності з існуючими системами

Однією з явних ознак того, що програмна система може потребувати реінжинірингу, є те, що вона застаріла або стала несумісною з сучасними технологіями. Застаріле програмне забезпечення, створене на застарілих платформах або з використанням застарілих мов програмування, може важко інтегруватися з новими системами або мати незручний інтерфейс користувача. У таких випадках стає необхідною реінжиніринг програмного забезпечення для оновлення системи та забезпечення сумісності з новітніми технологіями.

Обмеження продуктивності та масштабованості

Якщо програмна система відчуває вузькі місця продуктивності, часті збої або повільний час відгуку, це вказує на необхідність реінжинірингу. У міру того, як бізнес зростає, а вимоги користувачів зростають, програмні системи повинні бути здатні обробляти більші робочі навантаження та ефективно масштабуватися. Якщо існуюча система не відповідає вимогам до продуктивності або не має можливості масштабування, реінжиніринг може оптимізувати архітектуру програмного забезпечення, покращити алгоритми та покращити керування ресурсами для забезпечення кращої продуктивності та масштабованості.

Високі витрати на обслуговування та часті збої системи

Застарілі системи часто вимагають значного обслуговування та спричиняють високі витрати через застарілу інфраструктуру, складну кодову базу та відсутність підтримки з боку постачальників. Якщо система програмного забезпечення потребує частих виправлень, виправлень та оновлень, а пов’язані з цим витрати на технічне обслуговування стають непідйомними, це може бути ознакою необхідності реінжинірингу. За допомогою реінжинірингу програмного забезпечення компанії можуть оптимізувати кодову базу, усунути технічну заборгованість і зменшити навантаження на поточне обслуговування, що призведе до економії коштів і підвищення стабільності системи.

Подорож процесу реінжинірингу програмного забезпечення

Процес реінжинірингу програмного забезпечення дійсно можна умовно розділити на три основні фази:

- зворотний інжиніринг;
- реструктуризація;
- прямий інжиніринг.

Ці етапи працюють разом для вдосконалення та модернізації існуючих програмних систем. Розглянемо кожну фазу більш детально.

Фаза 1: зворотне проектування

Реверсивне проектування передбачає вивчення та розуміння поточної архітектури програмного забезпечення. Він має на меті отримати уявлення про архітектуру, дизайн і функціональність системи. Ось деякі ключові аспекти зворотного проектування:

– Перевірка коду: розробники вивчають вихідний код, документацію та інші доступні ресурси, щоб зрозуміти, як працює програмне забезпечення. Вони аналізують структуру коду, алгоритми та шаблони, що використовуються в системі.

– Перегляд документації: Існуюча документація, наприклад специфікації системи, посібники користувача та проектні документи, переглядається для збору інформації про передбачувану функціональність системи та конструктивні рішення.

– Аналіз системи: розробники аналізують поведінку системи, потік даних і взаємодію із зовнішніми компонентами. Вони можуть використовувати інструменти для візуалізації структури системи, визначення залежностей і відстеження потоку виконання.

Основною метою зворотного проектування є отримання повного розуміння існуючої програмної системи. Ці знання допомагають визначити сфери для вдосконалення та закладають основу для наступних етапів.

Етап 2: Реструктуризація

Фаза реструктуризації зосереджена на перепроектуванні та реорганізації системи програмного забезпечення для підвищення її продуктивності, зручності обслуговування та масштабованості. Ось основні аспекти етапу реструктуризації:

– Рефакторинг коду: розробники змінюють існуючу кодову базу, щоб покращити її структуру, читабельність і зручність обслуговування. Це може передбачати видалення повторюваного коду, вилучення багаторазових компонентів і застосування шаблонів проектування для підвищення модульності системи.

– Покращення архітектури: архітектуру системи можна переробити, щоб усунути недоліки архітектури, покращити масштабованість або включити нові технології. Це включає переоцінку взаємодії компонентів, впровадження рівнів або модулів і оптимізацію продуктивності системи.

– Видалення застарілого коду: застарілий або невикористаний код визначається та видаляється, зменшуючи складність і покращуючи продуктивність системи. Це розвантажує кодову базу та полегшує її розуміння та підтримку.

Фаза реструктуризації має на меті оптимізувати структуру програмної системи, зробити її більш ефективною, гнучкою та зручною для обслуговування. Він усуває виявлені недоліки та готує систему до майбутніх вдосконалень.

Фаза 3: передова інженерія

Фаза передового проектування передбачає використання знань, отриманих під час зворотного проектування, і вдосконалень, зроблених під час фази реструктуризації, для розробки нової та вдосконаленої версії програмного забезпечення. Ось ключові аспекти передового проектування:

– Реалізація оновлених компонентів: розробники впроваджують оновлені або оновлені компоненти на основі інформації, отриманої в результаті зворотного проектування та реструктуризації. Це може включати переписування коду, впровадження нових бібліотек або фреймворків та інтеграцію сучасних технологій.

– Введення нових функцій: оновлене програмне забезпечення може включати нові функції або функції, щоб відповідати оновленим вимогам бізнесу. Це передбачає проектування та розробку додаткових модулів або модулів із розширеними можливостями.

– Тестування та забезпечення якості: проводиться ретельне тестування , щоб переконатися, що оновлене програмне забезпечення відповідає визначеним вимогам і функціонує за призначенням. Це включає в себе різні методи тестування, такі як модульне тестування, інтеграційне тестування та тестування системи, щоб перевірити стабільність, надійність і продуктивність системи.

Фаза передового проектування завершується новою версією програмного забезпечення, яка включає в себе знання, отримані в результаті зворотного проектування, і вдосконалення, зроблені під час реструктуризації. Тепер оновлене програмне забезпечення готове до розгортання та використання.

Дотримуючись цих трьох етапів зворотного проектування, реструктуризації та передового проектування, процес реінжинірингу програмного забезпечення дозволяє підприємствам вдихнути нове життя у свої існуючі системи. Це дає їм змогу оптимізувати продуктивність, підвищити придатність до обслуговування та узгодити сучасні вимоги, гарантуючи, що система залишається надійною та адаптованою до мінливих потреб бізнесу.

Розробка структурної схеми

Реінжиніринг програмного забезпечення – це процес оновлення програмного забезпечення. Цей процес включає розробку додаткових функцій програмного забезпечення та додавання функцій для кращого та ефективнішого програмного забезпечення. Що стосується визначення, цей процес також передбачає, що програмний продукт матиме покращену ремонтопридатність.

Таким чином, реінжиніринг є кроком до постійного вдосконалення програмного забезпечення для кращого розвитку та клієнтського досвіду. Крім того, це спосіб зробити існуючі продукти покращеними.

Необхідність реінжинірингу програмного забезпечення стає невід'ємною частиною покращення якості ваших продуктів. Цей процес додає більше цінності вашому бізнесу, оскільки він не лише покращує ваші послуги, але й сприяє додатковому доходу.

Реінжиніринг програмного забезпечення є економічно ефективним методом розробки програмного забезпечення. чому Цей процес дозволяє виявити непотрібні елементи, реалізовані у вашому поточному програмному забезпеченні, і видалити їх із системи.

Роблячи це, ви мінімізуєте понесені витрати (часові, фінансові, прямі, непрямі тощо). Якщо клієнту потрібен продукт, який у вас уже є, але йому потрібні додаткові функції, вам, можливо, доведеться лише переробити наявний, щоб максимізувати ефективність розробки.

Що стосується економії, цей процес також полегшує обслуговування ваших продуктів. Можливість повторно перевірити програмне забезпечення може допомогти виявити кілька помилок у поточній реалізації.

Це дозволяє групі розробників приймати рішення про вдосконалення процедур розробки. Таким чином, тривалість обслуговування життєвого циклу розробки програмного забезпечення стає набагато легшою для виконання та підтримки.

Процес реінжинірингу програмного забезпечення в основному проходить три основні фази. Це (1) реверс-інжиніринг, (2) реструктуризація та (3) форвард-інжиніринг.

1. Зворотне проектування

Простий пошук у Google скаже нам цезворотне проектуванняє «відтворення продукту іншого виробника після детального вивчення його конструкції або складу». Однак цей процес не обмежується лише застосуванням цього процесу до продукту іншого виробника, а й до вашого власного.

Це досягається шляхом ретельного аналізу та перевірки специфікацій системи та розуміння існуючих процесів. Систематично, реверсування життєвого циклу розробки програмного забезпечення для реалізації програмного забезпечення найкраще підходить для цієї процедури, оскільки вона зазвичай розгадує кожен рівень від вищого рівня до нижчого рівня представлення системи.

2. Реструктуризація

Після завершення зворотного проектування та визначення відповідних специфікацій виконується реструктуризація. Реструктуризація пов'язана з перевпорядкуванням або реконструкцією вихідного коду та вирішенням питання про збереження чи зміну умов програмування.

Однак це не повинно вплинути на наявні функції програмного забезпечення. Натомість цей процес покращує їхню надійність і придатність до обслуговування.

Іншою частиною цієї процедури є видалення або реконструкція частин вихідного коду, які часто викликають помилки в програмному забезпеченні (також може бути налагодження).

Окрім цього, усунення застарілих або старіших версій певних частин системи (таких як програмна реалізація та апаратні компоненти) має підтримувати оновлення системи.

3. Передова техніка

Потік закінчується спередова техніка. Це процес інтеграції останніх специфікацій на основі результатів оцінки зворотного проектування та реструктуризації.

Стосовно процесу в цілому, це визначається відносно зворотного проектування, коли є спроба побудувати назад, від закодованого набору до моделі, або порушити процес інтеграції програмного забезпечення.

Немає конкретної моделі SDLC, якої слід дотримуватися при реінжинірингу програмного забезпечення. Модель завжди залежатиме від того, що найкраще підходить до навколишнього середовища та реалізації вашого продукту.

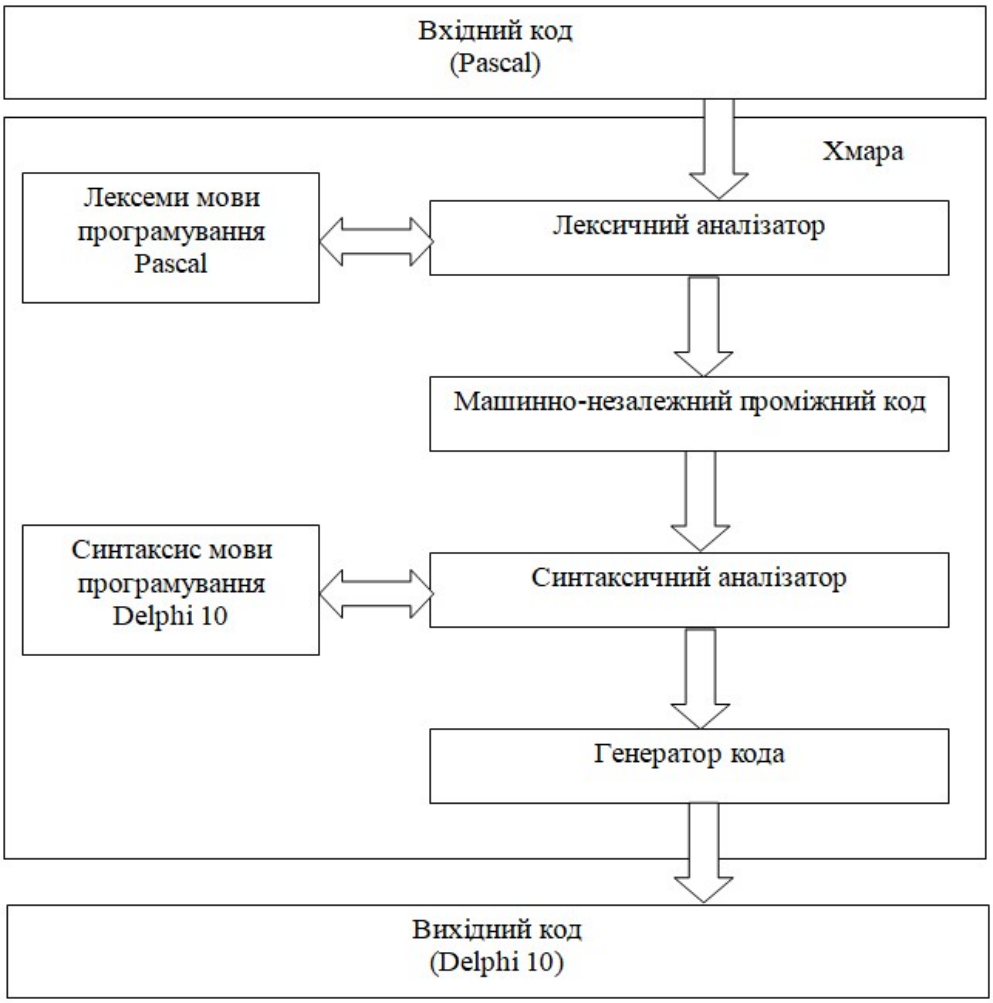

Рисунок 1 – Структурна схема системи

Однак, як і розробка програмного забезпечення, це систематична розробка, яка включає процеси всередині процесів і вимагає ретельної перевірки для бездоганних результатів.

Зараз багато IT-компаній осягнули концепцію реінжинірингу програмного забезпечення, оскільки вони займаються розробкою програмного забезпечення. З боку клієнта, багато малих і середніх підприємств вдаються до офшорного процесу в інші країни з набагато нижчими витратами та оплатою.

Офшоринг – це розширення діяльності вашої компанії (або частини вашої компанії) в іншій країні. Це зроблено з наміром заощадити витрати, мінімізувати навчання, отримати більше прибутку та отримати доступ до глобального ринку IT-індустрії з меншим наглядом, але повним контролем.

Відомим фактом є інформація про те. що зараз є велика кількість вихідного коду на старих мовах програмування, а саме таких як Pascal, Fortran, Modula.

Мови програмування застаріли та їх перестали використовувати та залишилась велика кількість вихідного коду на цих мовах. Цей код не має розвинутого інтерфейсу чи багатофункціональність але в цих кодах є реалізація математичних алгоритмів, високопродуктивних та новаторських підходів у реалізації математичних розв’язків рівнянь.

Незважаючи на те, що ці програми складалися досить давно, ідеї реалізації математичних теорій ніколи не старіють, особливо багато алгоритмічно-математичного вихідного коду використалося у вищих навчальних закладах.

Всі ці наробітки й рішення були заблоковані при старінні мови програмування.

Завдяки цьому розробка напрямку перекладу коду із застарілої мови програмування в більш новий завжди актуальна, код котрий надалі буде більш детально розглядатися та перероблятися під новий лад. Можливо також таке, що код після перетворення не буде у повній мірі робити але сама структура алгоритму залишиться.

У розробленій магістерській програмі використається переклад частин вихідного коду з мови програмування Pascal у Delphi 10.

На рисунку 1 зображена структурна схема системи де розглянута будова транслятора. При розробці схеми транслятора був проведений аналіз з теорії трансляторів [1-10].

З рисунку добре видно що транслятор розбито на декілька блоків а саме – блок лексичного аналізатора який взаємодіє з набором лексем мови програмування Pascal, проміжного машино-незалежного коду, синтаксичного аналізатора з синтаксисом мови Delphi 10 та генератора коду.

Детальний розгляд роботи цих компонентів розглянуто на функціональній схемі. На вхід транслятора поступає код Pascal, перед тим як запустити його у лексичний аналізатор (почати його оброблювати) проходить перевірка коду, що це дійсно є код Pascal. Ця дія виконується простою підстановкою шаблонного коду початку програми Pascal.

Висновки. У статті наведені теоретичне узагальнення й рішення наукового завдання дослідження методів автоматизованого реінжирінгу. Рішення даного завдання полягало у вирішенні наступних задач: Був проведений огляд існуючих систем автоматизованого реінжирінгу; Досліджена система автоматизованого реінжирінгу; На основі отриманих результатів досліджень створена програмна реалізація системи автоматизованого реінжирінгу. Розроблені під час виконання випускної кваліфікаційної роботи за другим (магістерським) рівнем вищої освіти алгоритми дозволяють успішно вирішувати завдання автоматизованого реінжирінгу. Проведено аналіз предметної галузі в ході якого були виявлені об'єкти, взаємодія яких носить істотний характер для функціональної діяльності предметної галузі, і їхні основні характеристики; побудована алгоритм і вибраний середовище розробки.

Список літератури

1. Smirnov, O., Kuznetsov, A., Potii, O., Poluyanenko, N., Stelnyk, I., Mialkovsky, D. «Combining and filtering functions in the framework of nonlinear-feedback shift register». International Journal of Computing; 2020, Volume 19, Issue 2 – Research Institute for Intelligent Computer Systems – 2020. – P. 247-256.
2. Smirnov O., Kuznetsov A., Kiian A., Kuznetsova T. «Non-binary constant weight coding technique». CEUR Workshop Proceedings. Volume 2740, 2020, Pages 102-114.
3. Smirnov O., Kuznetsov A., Kiian A., Cherep A., Kanabekova M., Chepurko I. «Testing of code-based pseudorandom number generators for post-quantum application». 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), Ukraine, Kyiv, May 14-18. 2020. P. 172-177.
4. Smirnov, O., Shekhanin, K., Kuznetsov, A., Krasnobayev, V. «Detecting Hidden Information in FAT». International Journal of Computer Network and Information Security (IJCNIS). Vol. 12, No. 3, 2020. PP.33-43.
5. Smirnov, O., Drieieva, H., Drieiev, O., Simakhin, V., Bondar, S., Odarchenko, R. «Managing multifractal properties of the binary sequence generated with the Markov chains», CEUR Workshop Proceedings Volume 2608, 2020, Pages 633-645.
6. Smirnov O. Kuznetsov A., Zaicenko Yu., Pastukhov M., Oleshko O., Kuznetsova K., «Formation of Discrete Signals with Special Correlation Properties». International Conference on Information and Telecommunication Technologies and Radio Electronics, UkrMiCo 2019; Odessa; Ukraine; 9-13 September 2019. P.22-28.
7. Smirnov, O., Kuznetsov, A., Kolovanova, I., Kuznetsova, T., «Noise immunity of the algebraic geometric codes». International Journal of Computing; 2019, Volume 18, Issue 4 – Research Institute for Intelligent Computer Systems – 2019. – P. 393-407.
8. Smirnov, O., Kuznetsov, A., Reshetniak, O., Ivko, N., Katkova, T., Kuznetsova, T., «Generators of Pseudorandom Sequence with Multilevel Function of Correlation». 2019 IEEE International Scientific-Practical Conference Problems of Infocommunications, Science and Technology (PIC S&T), Kyiv, Ukraine, 8 – 11 October 2019 . P.517-522.
9. Smirnov, O., Krasnobayev, V., Yanko, A., Kuznetsova, T. «Methods of nulling numbers in the system of residual classes». CEUR Workshop Proceedings, Vol 2588, P. 90-106, 2019.
10. Kuznetsova, T., «Code-Based Schemes for Post-Quantum Digital Signatures», 10th IEEE International

Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019; Metz; France; 18-21 September 2019. P. 707-712.

11. Smirnov, O., Kuznetsov, A., Stefanovych, O., Gorbenko, Y., Krasnobaev, V., Kuznetsova K. «Information Hiding Using 3D-Printing Technology», 10th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2019; Metz; France; 18-21 September 2019. P.701-706.
12. Smirnov, O., Kuznetsov, A., Kovalchuk, D., Averchev, A., Pastukhov, M., Kuznetsova, K., «Formation of Pseudorandom Sequences with Special Correlation Properties», 2019 3rd International Conference on Advanced Information and Communications Technologies, AICT -2019/ Lviv, Ukraine, 2-6 July, 2019, P. 395-399.
13. Вінтенко Б.Ю., Смірнов О.А., Коваленко А.С., Смірнов С.А., Буравченко К.О. «Дослідження вимог міжнародних стандартів IEC60880 та IEC62138 з розробки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Системи управління, навігації та зв'язку, 2023, вип. 3(73), С. 155-166.
14. Вінтенко, Б., Миронець, І., Смірнов, О., Кравчук, О., Козірова, Н., Савеленко, Г., Коваленко, А. «Дослідження вимог та аналіз кібербезпеки програмного забезпечення інформаційно-керуючих систем АЕС, важливих для безпеки». Кібербезпека: освіта, наука, техніка. 2024. №3(23), С. 111-131.
15. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А., Коваленко А.С. «Дослідження нормативних документів та галузевих стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». Системи управління, навігації та зв'язку, 2023, вип. 2(72), С. 170-178.
16. Аль-Мудхафар Акіл Абдулхуссейн М., Смірнова Т.В., Буравченко К.О., Смірнов О.А. «Метод оцінки та підвищення користувальницького досвіду абонентів в програмно-конфігурованих мережах на основі використання машинного навчання». Сучасні інформаційні системи, 2023, том 7, № 2, С. 49-56.
17. Вінтенко Б.Ю., Смірнов О.А., Коваленко О.В., Смірнов С.А. «Дослідження нормативної документації та стандартів розробки програмного забезпечення комп'ютерних систем управління АЕС, важливих для безпеки». VI міжнародна науково-практична конференція "Інформаційна безпека та комп'ютерні технології", м. Кропивницький. 20-21 квітня 2023 р. – Кропивницький: ЦНТУ. – 2023. – С. 35-36.
18. Смірнов, О.А., Усік П.С., Полігенько О.О., Одарченко Р.С., Терещенко Л.Ю. «Інформаційна технологія та програмне забезпечення для підвищення ефективності планування підсистеми базових станцій стільникового зв'язку». Проблеми телекомунікацій. № 1(26). С. 83-96. 2020.
19. Смірнов О.А., Усік П.С., Миронець І.В., Буравченко К.О., Якименко Н.М. «Метод підвищення ефективності розподіленої обробки даних у комп'ютерних системах операторів стільникового зв'язку» Вісник Черкаського державного технологічного університету. Технічні науки. №4. С. 103-110. 2020.
20. О.А.Смірнов, Т.В.Смірнова, Л.І. Поліщук, К.О. Буравченко, А.О.Макевнін, «Дослідження хмарних технологій як сервісів», Кібербезпека: освіта, наука, техніка. № 3(7). С. 43-62. 2020.
21. Смірнов О.А., Дрєєва Г.М., Дрєєв О.М., Смірнова Т.В. «Фрактальний аналіз генератора самоподібного трафіку на основі ланцюга Маркова». Центральноукраїнський науковий вісник. Технічні науки. № 2(33). с. 161-172, 2019.