COMPUTER SYSTEMS AND NETWORKS

Vol. 5, No. 1, 2023

DOI: https://doi.org/10.23939/csn2023.01.182

УДК 004.9

АЛГОРИТМІЧНІ ТА ПРОГРАМНІ ЗАСОБИ ДЛЯ АВТОМАТИЧНОГО ВСТАНОВЛЕННЯ ТА ЗАПУСКУ УТИЛІТ В СЕРЕДОВИЩІ WINDOWS

О. Р. Яцків, Ю. С. Клушин

Національний університет "Львівська політехніка",
кафедра електронних обчислювальних машин
E-mails: oleh.yatskiv.mkisp.2022@lpnu.ua, yurii.s.klushyn@lpnu.ua

У сучасному світі автоматизація процесів інсталяції та управління програмним забезпеченням у середовищі Windows є ключовим елементом для забезпечення зручності та ефективності користувачів. У межах цього напряму була розроблена спеціалізована програма, мета якої – істотне спрощення процесів інсталяції та управління утилітами. Програма оснащена інтуїтивним користувацьким інтерфейсом, який сприяє безпроблемній інтеграції з операційною системою та надає користувачам легкий доступ до необхідних інструментів.

Архітектуру програми побудовано з використанням модульних моделей та гнучких провайдерів даних, які забезпечують динамічне встановлення та оновлення утиліт. Основні моделі визначають структуру даних і логіку взаємодії із користувацьким інтерфейсом та іншими компонентами системи, тоді як провайдери даних адаптовані для зчитування, оновлення та розподілу інформації з різноманітних джерел, ураховуючи локальні файли, сервери та віддалені системи.

Упровадження програми передбачало детальний аналіз відомих методів управління утилітами, із урахуванням виявлених недоліків та обмежень. З огляду на це було сформульовано вимоги до функціоналу програми, що забезпечили б підвищення продуктивності та зменшення можливих помилок під час встановлення та конфігурації утиліт.

Ретельне тестування та користувацькі опитування допомогли оцінити програму з позицій простоти використання, функціональності та загальної задоволеності. Висновки дослідження виявили високий рівень ефективності програми, підтвердивши її спроможність досягати встановлених цілей і позитивно впливати на досвід користувачів з утилітами.

Ключеві слова: Windows; сервіс; програмне забезпечення; оптимізація роботи; продуктивність.

Вступ

Операційна система Windows – одна з найпопулярніших і найпоширеніших операційних систем у світі з більш ніж мільярдом користувачів. Не дивно, що для Windows існує незліченна кількість програм та утиліт. Встановлення та керування програмними утилітами в операційній системі Windows може бути складним і трудомістким завданням, особливо якщо їх велика кількість. З цією проблемою щодня стикаються як організації корпоративного рівня з великою кількістю комп'ютерів і пристроїв для управління, так і користувачі, не обізнані з технічними особливостями операційної системи Windows.

Для вирішення цієї проблеми розроблено спеціалізовані програмні платформи, які полегшують користувачам встановлення та запуск утиліт у середовищі Windows. Ці платформи мають простий у використанні інтерфейс, який допомагає користувачам здійснити інсталяцію та часто пропонує додаткові функції, такі як автоматичне оновлення, обслуговування та усунення несправностей.

Спеціалізована програмна платформа для встановлення утиліт у Windows може істотно спростити встановлення, оновлення та керування програмними утилітами. Ці платформи зазвичай на-

дають централізоване місце для управління програмним забезпеченням, що дає змогу адміністраторам легко встановлювати й налаштовувати програмне забезпечення на різних пристроях, а також відстежувати та контролювати його використання.

Багато організацій віддають перевагу створенню єдиного стандарту для програмного забезпечення, що використовується на робочих станціях працівників. Це дає змогу забезпечити не тільки легке управління, але і високий рівень безпеки, оскільки адміністратори можуть швидко виявляти й виправляти потенційні проблеми.

Зі зростанням популярності хмарних технологій можливість синхронізації налаштувань та програм між різними пристроями стала особливо актуальною. Спеціалізовані програмні платформи допомагають вирішувати цю проблему, забезпечуючи користувачам можливість працювати ефективно, де б вони не перебували.

Загалом, спеціалізована програмна платформа для управління утилітами на базі Windows може істотно підвищити ефективність і безпеку IT-інфраструктури організації, даючи змогу адміністраторам зосередитися на інших важливих завданнях і обов'язках.

Аналіз наявних рішень

Для полегшення процесу встановлення програмного забезпечення та керування утилітами розроблено різні платформи. У цьому аналізі ми зосередимося на найпопулярніших менеджерах пакетів для операційної системи Windows.

Windows Package Manager 1 – це інструмент командного рядка, призначений для керування встановленням, оновленням та обслуговуванням програмного забезпечення у Windows 10 і пізніших версіях операційної системи. Інструмент використовує файл маніфесту на основі YAML для надання вичерпного опису програмних пакетів, ураховуючи залежності, місця встановлення та інші відповідні метадані. Однією із ключових переваг цього інструменту є простота використання, швидкий процес встановлення та автоматизоване керування залежностями. Для тих, хто надає перевагу графічному інтерфейсу, WingetUI доступний для окремого завантаження та встановлення, забезпечуючи зручніший досвід керування програмними пакунками. Однак варто зазначити, що він має обмежені репозиторії програмного забезпечення.

Windows Store 2, платформа цифрової дистрибуції, розроблена корпорацією Майкрософт, є ще однією платформою для встановлення та керування програмним забезпеченням у Windows 10 і пізніших версіях операційних систем. Вона дає змогу користувачам завантажувати та встановлювати програми, ігри та інше програмне забезпечення з такими перевагами, як безпека, автоматичне оновлення та синхронізація пристроїв. Однак у неї обмежений вибір програм і можливостей налаштування. Крім того, вона забороняє програми, що потребують адміністративного доступу.

Інший менеджер пакунків для Windows, Chocolatey 3, забезпечує простий підхід до інсталяції та керування програмним забезпеченням у Windows 7 і пізніших версіях операційних систем. Він пропонує такі переваги, як простота використання, швидке встановлення та автоматизоване керування залежностями. Для покращення користувацького досвіду ChocolateyGUI доступний для окремого завантаження та встановлення, пропонуючи графічний інтерфейс для навігації та керування програмними пакетами за допомогою Chocolatey. Однак у нього обмежена кількість сховищ програмного забезпечення.

І, нарешті, Ninite 4 – вебсервіс, який дає змогу користувачам операційної системи Windows одночасно завантажувати та інсталювати кілька програм. Він пропонує такі переваги, як простота використання, автоматичне оновлення та можливості налаштування. Однак у нього обмежені доступність програмного забезпечення та можливості налаштування, а також бракує управління пакунками та підтримки користувачів.

Порівняння алгоритмів шифрування

Алгоритми шифрування мають вирішальне значення для забезпечення безпеки даних користувача в спеціалізованих програмних платформах для встановлення утиліт у середовищі Windows. Існують різні алгоритми шифрування, у кожного з яких свої переваги та недоліки.

Симетричні алгоритми, такі як AES, Blowfish та Kalyna 5, використовують той самий ключ для шифрування та дешифрування. AES – один з найпоширеніших алгоритмів шифрування, а Blowfish – швидкий та ефективний. Kalyna – це новий симетричний алгоритм шифрування, розроблений спеціально для сучасних криптографічних вимог, що робить його надійним і безпечним вибором для шифрування даних у різних системах і додатках.

Асиметричні алгоритми, такі як RSA і ECC, використовують два різні ключі для шифрування і розшифрування. RSA широко використовують для цифрових підписів і протоколів обміну ключами, тоді як ECC призначений для забезпечення такого самого рівня безпеки, як і RSA, але з меншим розміром ключа.

Для утиліти Kalyna є кращим вибором. Завдяки надійному шифруванню, ефективній роботі та підтриманню ключів різної довжини вона є надійним і безпечним варіантом для забезпечення конфіденційності та цілісності даних користувача.

Технічні компоненти

Технічні компоненти цього програмного проєкту вибрано на основі їхніх унікальних сильних сторін і того, наскільки добре вони відповідали вимогам проєкту. Dynamic-Link Library (DLL) 6 вибрано для того, щоб дати змогу декільком додаткам використовувати ті самі код і ресурси, що зменшує дублювання функціоналу і підвищує ремонтопридатність. JSON 7 вибрано як формат обміну даними, оскільки він є легким, простим для аналізу та широко підтримується сучасними мовами програмування, що робить його ефективним форматом для обміну даними між додатками. C++ 8 та C# 9 вибрано з огляду на їх швидкість, ефективність та надійність, причому C++ ідеально підходить для критично важливих до продуктивності компонентів завдяки низькорівневому контролю над системними ресурсами, а C# – для розроблення користувацьких інтерфейсів, бізнес-логіки та інших компонентів, які не потребують низькорівневого контролю.

WPF 10 використано як фреймворк користувацького інтерфейсу, що забезпечує гнучкість і багатий набір функцій для створення інтерактивних користувацьких інтерфейсів за допомогою XAML, який відокремлює логіку інтерфейсу від логіки додатка. Такий підхід забезпечує кращу підтримку, тестування та розширюваність інтерфейсу. Нарешті, .NET 11 вибрано як платформу для розроблення завдяки потужному набору інструментів та бібліотек, які спрощують розроблення та сприяють крос-платформній сумісності. .NET дає змогу розробникам писати код кількома мовами програмування, зокрема C++ та C#, а також надає потужний набір бібліотек класів для виконання поширених завдань, таких як файлове введення/виведення, робота з мережею та криптографія.

Постановка проблеми

Сучасні менеджери пакунків для операційних систем Windows мають різноманітні проблеми, які необхідно вирішити для покращення роботи користувачів. Наявні платформи, такі як Windows Package Manager (Winget), Chocolatey та Ninite, стикаються з обмеженнями, які впливають на їхню загальну ефективність та зручність використання.

Однією із головних проблем цих менеджерів пакунків є обмеженість репозиторіїв програмного забезпечення, які вони пропонують. Користувачі, яким потрібен широкий спектр програм, можуть вважати ці обмеження істотною проблемою. Щоб задовольнити ширшу аудиторію, важливо розширити доступні репозиторії програмного забезпечення на цих платформах.

Іншою проблемою є доступність цих менеджерів пакунків, особливо для тих, хто не дуже добре знайомий з інтерфейсами командного рядка. Хоча було розроблено окремі графічні інтерфейси, такі як WingetUI для Windows Package Manager та ChocolateyGUI для Chocolatey, більш інтегрований та зручний для користувача підхід міг би покращити загальний досвід.

Платформа Windows Store, пропонуючи такі переваги, як безпека, автоматичне оновлення та синхронізація пристроїв, страждає від обмеженого вибору програм і варіантів налаштування. Крім того, вона обмежує програми, які потребують адміністративного доступу, що потенційно заважає користувачам встановлювати необхідні програми.

Ninite, з іншого боку, дає змогу користувачам завантажувати та встановлювати кілька програм одночасно, але не має функцій керування пакунками та підтримки користувачів. Це обмеження може відштовхнути користувачів, які потребують цих функцій, від використання платформи.

Зважаючи на ці проблеми, потрібно розробити комплексну платформу для встановлення та управління програмним забезпеченням, яка б долала обмеження наявних рішень. Всебічно розвинена платформа пропонуватиме великі сховища програмного забезпечення, інтуїтивно зрозумілі графічні інтерфейси, широкий спектр програм, надійне управління пакунками та підтримку користувачів, що у підсумку забезпечить безперешкодніший досвід для користувачів операційної системи Windows.

Мета статті

У цій статті розглянуто роботу однієї з таких спеціалізованих програм, яка створена на основі запропонованого алгоритму і призначена для спрощення встановлення та запуску утиліт у середовищі Windows. Продемонстровано можливість інтегрувати її в наявні IT-системи, щоб спростити управління програмним забезпеченням і підвищити загальну продуктивність.

Розглянуто особливості та переваги цієї програми, а також те, чим вона відрізняється від інших подібних програм. Крім того, обговоримо різні типи утиліт, які можна встановити і запустити за допомогою цієї платформи, і як вони можуть допомогти оптимізувати роботу користувача в середовищі Windows.

Основна мета дослідження – проєктування та розроблення алгоритму комплексної програми для управління утилітами в середовищі Windows, щоб істотно покращити користувацький досвід, вирішивши різні аспекти та проблеми, пов’язані з процесом інсталяції. Програма пропонуватиме додаткові функції, такі як можливість запускати програми без інсталяції (портативний режим) і легко створювати користувацькі пакети для особистого використання за допомогою інтуїтивно зрозумілого інтерфейсу. Було визначено кілька цілей для оцінювання ефективності платформи та розуміння її переваг, що у підсумку допоможе усунути основні недоліки наявних програм.

Одна з цілей – сприяння простоті встановлення та експлуатації, що передбачає таке розроблення програми, щоб зменшити складність процесу встановлення. Для цього будуть розроблені автоматизовані процедури, які зменшують кількість кліків користувача і надають користувачам чіткі інструкції. Програма також буде інтегрована з операційною системою Windows, забезпечуючи безперебійну сумісність і підтримку застарілих і майбутніх утиліт, тим самим мінімізуючи потенційні конфлікти і покращуючи загальний користувацький досвід.

Ще однією метою є підвищення ефективності процесу інсталяції, що передбачає мінімізацію необхідності ручного втручання, введення вбудованих механізмів обробки помилок і відновлення, які зменшують кількість збоїв під час інсталяції. Крім того, будуть використані передові алгоритми для виявлення та вирішення проблем встановлення у режимі реального часу, що зменшить потребу у втручанні користувача та забезпечить безперебійне та ефективне встановлення.

Наступна мета – спрощення використання як для технічних, так і для нетехнічних користувачів. Цього можна досягти завдяки інтуїтивно зрозумілому користувацькому інтерфейсу, контекстозалежній допомозі та підказкам. Платформа також міститиме функції, які допомагають користувачам приймати обґрунтовані рішення, такі як рекомендації щодо утиліт та перевірка сумісності, що сприятиме зменшенню конфліктів програмного забезпечення. Щоб ще більше покращити користувацький досвід, платформа підтримуватиме кілька мов і варіантів локалізації, що забезпечить доступність для глобальної бази користувачів.

Нарешті, мета забезпечення безпеки та конфіденційності передбачає надання пріоритету безпеці та конфіденційності користувачів завдяки впровадженню суворих заходів безпеки та дотриманню найкращих галузевих практик. Будуть передбачені такі функції, як безпечні канали завантаження, перевірка цілісності утиліт за допомогою цифрових підписів і шифрування конфіденційних

даних користувачів. Розроблений алгоритм повинен забезпечити дотримання відповідних правил захисту даних, таких як GDPR, щоб захистити конфіденційність користувачів.

Отже, розроблення алгоритму програми для автоматичного встановлення та запуску утиліт в середовищі Windows спрямоване на досягнення цілей, що полягають у спрощенні встановлення та експлуатації, підвищенні ефективності процесу встановлення та зручності використання як для технічних, так і для нетехнічних користувачів, а також забезпеченні безпеки та конфіденційності. Успішне досягнення цих цілей, підкріплене конкретними кількісними показниками, сприятиме загальному успіху платформи у вирішенні проблем, з якими стикаються користувачі під час інсталяції утиліт, і переосмисленню очікувань користувачів щодо ефективності, зручності та безпеки.

Структурна модель системи

Структурна модель системи (рис. 1) для спеціалізованої програми автоматичного встановлення та запуску утиліт в середовищі Windows складається з декількох компонентів, які працюють разом для забезпечення необхідної функціональності. До таких компонентів належать модуль DLL, який реалізує логіку запуску та встановлення програм, служба, яка працює у фоновому режимі та обробляє запити від інтерфейсу користувача, та інтерфейс користувача, який дає змогу користувачеві вибирати, яку програму запускати.

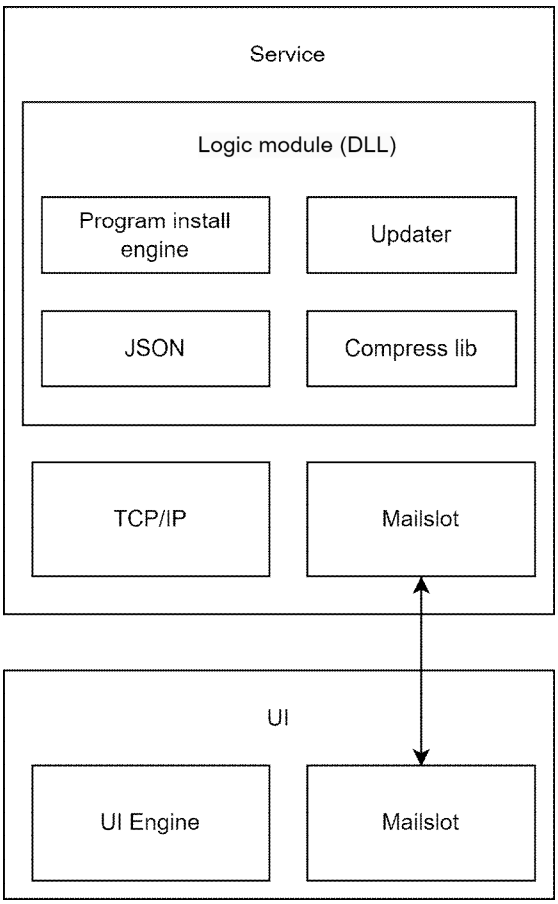

Рис. 1. Структурна модель системи

Модуль DLL є ядром системи, оскільки містить весь необхідний функціонал для запуску та встановлення програм. Цей модуль відповідає за управління встановленням і запуском утиліт, а також за обробку будь-яких помилок або винятків, що виникають під час цього процесу. Модуль

розроблено так, щоб його можна було легко оновлювати новими функціями або виправляти помилки, гарантуючи, що система завжди буде актуальною і працюватиме безперебійно.

Program install engine – інший важливий компонент системи, який відповідає за автоматизацію процесу встановлення програм. Він працює в тісній взаємодії з модулем DLL, щоб отримати інструкції щодо встановлення, такі як шляхи до встановлення, необхідні системні вимоги, параметри встановлення та інші налаштування. Program install engine також здатен обробляти різні формати інсталяційних пакетів, що робить його універсальним рішенням для встановлення різних типів програм.

Крім основного процесу встановлення, Program install engine також передбачає функції для опрацювання виняткових ситуацій, таких як нестача місця на диску, конфлікти версій або відсутність необхідних залежностей. Це гарантує, що програми будуть встановлені правильно й ефективно, навіть якщо виникають проблеми.

Основна мета Program install engine полягає в тому, щоб зробити процес встановлення максимально автоматизованим, забезпечивши надійність та стабільність. Це дає користувачам змогу легко і швидко встановлювати програми, не заглиблюючись у технічні деталі процесу встановлення.

Updater – служба, яка відповідає за автоматичне оновлення всієї системи, зокрема модуля DLL, сервісу та UI. Вона регулярно перевіряє наявність нових версій компонентів системи на віддаленому сервері та завантажує їх за необхідності. Однією з ключових особливостей Updater є здатність оновлювати систему без необхідності перезапуску, що забезпечує безперебійну роботу користувача. Також Updater має низку засобів безпеки, щоб гарантувати, що оновлення завантажуються лише з авторизованих джерел і не містять шкідливого коду.

Окрім модуля DLL, важливим компонентом системи встановлення програм є також архівна бібліотека. Ця бібліотека містить стислі інсталятори програм, такі як виконувані файли, скрипти та інші необхідні файли, щоб зробити процес встановлення ефективнішим і швидшим. Бібліотека архівів також містить алгоритми стиснення та розпакування файлів інсталяторів, завдяки чому інсталятори програм займають мінімум місця на диску.

Крім того, бібліотека архівів відповідає за керування версіями інсталяторів програм, гарантуючи, що для встановлення доступні лише найновіші та найстабільніші версії. Ця система контролю версій має такі функції, як відкат або точки відновлення, що дає користувачам змогу повернутися до попередньої версії, якщо це необхідно. Бібліотека архівів також передбачає процес перевірки, який гарантує, що стиснуті інсталятори програм є дійсними і не були підроблені, забезпечуючи додатковий рівень безпеки для системи. Цей процес перевірки охоплює перевірку контрольної суми або цифрових підписів для забезпечення цілісності та автентичності файлів інсталяторів.

Сервісний компонент системи відповідає за оброблення всіх запитів від інтерфейсу користувача. Цей компонент працюватиме у фоновому режимі та прослуховуватиме вхідні запити від інтерфейсу користувача. Отримавши запит, сервіс обробить його і або запустить, або встановить запитувану програму. Сервіс також оброблятиме будь-які помилки або винятки, що виникають під час процесу, і надаватиме користувацькому інтерфейсу зворотний зв’язок про статус запиту.

Компонент інтерфейсу користувача системи надасть користувачеві простий у використанні інтерфейс для вибору та запуску програми. Інтерфейс відображатиме список доступних програм, а також будь-яку відповідну інформацію про кожну програму, наприклад, номер версії та дату випуску. Потім користувач може вибрати програму, яку хоче запустити, а інтерфейс надішле запит до сервісу на запуск або встановлення програми.

На додаток до стандартного користувацького інтерфейсу, система також надаватиме віддалений доступ для адміністраторів. Це дасть змогу адміністраторам надсилати запити до сервісу віддалено, без необхідності прямого доступу до користувацького інтерфейсу. Це може бути корисно для управління системою в ситуаціях, коли прямий доступ неможливий або недоцільний.

Загалом структурна модель системи для спеціалізованої програми автоматичного встановлення та запуску утиліт у середовищі Windows спроєктована так, щоб вона була стійкою, надійною та простою у використанні. Виокремлення логіки запуску та встановлення програм в окремий модуль DLL та надання спеціального сервісу для обробки запитів забезпечує швидкий та ефективний запуск та встановлення програм, а також гарантує безпечну та надійну обробку будь-яких помилок чи виняткових ситуацій.

Зв'язок між компонентами системи

У спеціалізованій програмі автоматичного встановлення та запуску утиліт у середовищі Windows сервіс та користувацький інтерфейс взаємодіють один з одним через Mailslot. Це дає змогу сервісу і користувацькому інтерфейсу спілкуватися і обмінюватися інформацією один з одним, навіть якщо це окремі процеси, запущені на одній машині.

Зв'язок між сервісом та інтерфейсом користувача через Mailslot передбачає використання механізму, схожого на поштову скриньку. Коли користувацький інтерфейс запускається, він створює з'єднання з сервісом через Mailslot. Це з'єднання дозволяє користувацькому інтерфейсу надсилати запити до сервісу та отримувати відповіді від сервісу.

Для запуску запиту користувацький інтерфейс пише повідомлення до Mailslot із зазначенням дії, яку потрібно виконати, наприклад, встановити або запустити програму. Це повідомлення містить всю необхідну інформацію, яка потрібна сервісу для виконання запитуваної дії. Повідомлення надсилається через Mailslot і сервіс його приймає.

Отримавши повідомлення, сервіс обробляє запит і виконує запитувану дію. Якщо дія успішна, сервіс надсилає повідомлення-відповідь назад до поштової скриньки, в якому зазначає, що дію виконано успішно. Якщо дія не вдалася, сервіс надсилає повідомлення-відповідь до поштової скриньки із зазначенням причини невдачі.

Потім користувацький інтерфейс читає повідомлення-відповідь від Mailslot і відповідно оновлює користувацький інтерфейс. Наприклад, якщо користувач запросив інсталяцію програми й інсталяція відбулася успішно, користувацький інтерфейс може відобразити повідомлення про те, що програму успішно встановлено. Якщо інсталяція не вдалася, інтерфейс користувача може відобразити повідомлення про помилку із зазначенням причини невдачі.

Протягом усього процесу Mailslot використовується для зв'язку між службою та користувацьким інтерфейсом. Mailslot забезпечує швидкий та ефективний зв'язок між цими двома процесами, що важливо для забезпечення швидкого реагування та належної роботи системи.

Загалом, зв'язок між сервісом та інтерфейсом користувача через Mailslot є критично важливим компонентом спеціалізованої програми автоматичного встановлення та запуску утиліт у середовищі Windows. Використовуючи Mailslot, сервіс і користувацький інтерфейс спілкуються та обмінюються інформацією один з одним, забезпечуючи швидкий та ефективний запуск і встановлення програм, а також гарантуючи, що будь-які помилки або винятки обробляються безпечним і надійним способом.

Клієнт-серверна модель

Архітектуру клієнт-сервер

Рис. 2) реалізовано для забезпечення зв'язку між адміністратором та сервісами клієнтів на їхніх комп'ютерах. Адміністратор виконує роль сервера, тоді як клієнтські сервіси – ролі клієнтів, а зв'язок оснований на протоколі TCP/IP.

Зв'язок між клієнтом і сервером відбувається за схемою обміну повідомленнями "запит-відповідь" і повинен відповідати загальному протоколу зв'язку, який формально визначає правила, мову і діалогові шаблони, які потрібно використовувати. У цьому випадку протокол TCP застосову-

ється для підтримки з'єднання до завершення обміну повідомленнями. Протокол TCP також допомагає розподіляти програми на пакети, які мережа може доставити, передавати пакети і приймати пакети з мережі, а також керувати потоком і повторним передаванням втрачених або спотворених пакетів.

Клієнтські запити організовуються і розставляються за пріоритетами в системі планування, щоб допомогти серверу впоратися з отриманням запитів від багатьох різних клієнтів за короткий проміжок часу. Такий підхід дає змогу будь-якому універсальному комп'ютеру розширити свої можливості, використовуючи спільні ресурси інших хостів. Архітектура клієнт-сервер також гарантує, що дані, які передаються за допомогою клієнт-серверних протоколів, не залежать від платформи, що є вагомою перевагою.

Загалом, клієнт-серверна модель, що використовується в спеціалізованій програмі автоматичного встановлення та запуску утиліт у середовищі Windows, надає багато переваг, таких як можливість захисту даних і управління авторизацією та автентифікацією користувачів, можливість додавання ресурсів до мережі без великих перерв, а також ефективний доступ до даних без необхідності розташування клієнтів і сервера поблизу. Ця архітектура також полегшує оновлення, заміну та переміщення вузлів у системі.

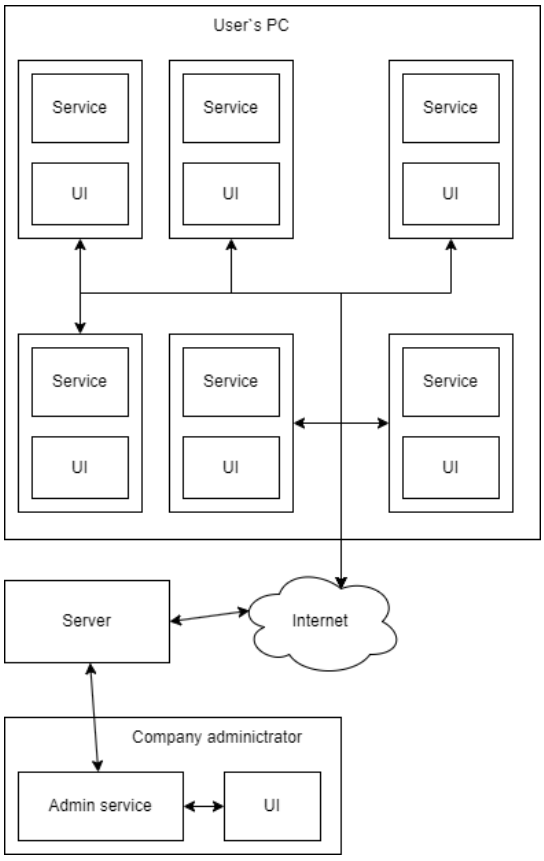

Рис. 2. Клієнт-серверна модель

Безпека даних користувача

Передавання даних між клієнтом і сервером – поширена практика в багатьох галузях. Однак вона створює значний ризик для безпеки користувачів, оскільки дані, що передаються, можуть перехопити та скомпрометувати зловмисники. Для забезпечення конфіденційності та цілісності даних користувачів під час передавання необхідні алгоритми шифрування, такі як Kalyna.

Використовуючи алгоритм Kalyna для шифрування даних під час передавання між клієнтом і сервером, організації можуть гарантувати, що конфіденційні дані користувачів захищені від пере-

хоплення і компрометації. Алгоритм забезпечує надійне шифрування, що ускладнює зловмисникам розшифрування інформації без ключа. В результаті це призводить до 100-відсоткового шифрування даних користувача, що ще більше підвищує безпеку процесу передавання даних.

З погляду безпеки та конфіденційності спеціалізована програма автоматичного встановлення та запуску утиліт у середовищі Windows надає пріоритет безпеці та конфіденційності користувачів, впроваджуючи суворі заходи безпеки та дотримуючись найкращих галузевих практик. Передбачено такі функції, як захищені канали завантаження, перевірка цілісності утиліт за допомогою цифрових підписів та шифрування усіх конфіденційних даних користувача. Платформу розроблено так, щоб забезпечити дотримання відповідних правил захисту даних, таких як GDPR, щоб захистити конфіденційність користувачів.

Крім того, програмна платформа може використовувати алгоритм Kalyna в поєднанні з іншими заходами безпеки, такими як цифровий підпис, автентифікація та контроль доступу, щоб забезпечити комплексне рішення безпеки. Такий багаторівневий підхід до безпеки гарантує, що конфіденційна інформація користувачів залишається захищеною під час передавання та зберігання даних, що істотно знижує ризик несанкціонованого доступу та витоку даних. Завдяки інтеграції цих функцій безпеки платформа істотно підвищила рівень безпеки та конфіденційності користувачів, пропонуючи безпечний та надійний досвід встановлення утиліт.

Алгоритм роботи системи

Алгоритм інтерфейсу (рис. 3) розпочинається із запуску сервісу. Спочатку ініціалізується процес, під час якого система перевіряє, чи є з'єднання з сервісом. У разі неполадок користувач бачить повідомлення про помилку.

Після успішного запуску система отримує список профілів для оброблення. Кожен профіль обробляється послідовно. Для цього використовується змінна ітератора, яка ініціалізується значенням нуль і збільшується після опрацювання кожного профілю. Система завантажує дані профілю, а потім отримує інформацію про групи цього профілю.

Далі для кожної групи в профілі система завантажує дані про цю групу та пов'язані з нею програми. Цей процес також контролює ітератор, який збільшується після обробки кожної групи.

Завершивши опрацювання всіх груп і профілів, система переходить у режим очікування дії від користувача. Коли користувач виконує дію, система формує і надсилає запит для її опрацювання. Потім система очікує на відповідь. Коли відповідь отримана, система обробляє її, і, залежно від результату, користувачеві може бути показано повідомлення або виконано інші дії.

Весь цей процес повторюється, поки користувач або система не завершить роботу.

Робота сервісу спеціалізованої програми автоматичного встановлення та запуску утиліт у середовищі Windows критично важлива для загальної продуктивності та надійності системи. Сервіс відповідає за отримання запитів від користувацького інтерфейсу та своєчасне й ефективне опрацювання цих запитів. Це забезпечило істотне підвищення ефективності, оскільки платформа успішно мінімізувала ручне втручання, зменшила кількість збоїв під час встановлення та оптимізувала швидкість встановлення. Нижче наведено детальний опис роботи сервісу (рис. 4).

Сервіс запускається та ініціалізує всі необхідні ресурси для початку роботи: налаштування Mailslot для зв'язку з інтерфейсом користувача, завантаження DLL-модуля, що реалізує логіку запуску та встановлення програм, а також ініціалізує усі інші необхідні ресурси.

Сервіс очікує на повідомлення, яке надходить до поштової скриньки. Повідомлення містить запит користувача, наприклад, встановити або запустити програму. Сервіс очікує, що повідомлення міститиме всю необхідну інформацію для виконання запитуваної дії, наприклад, шлях до виконуваного файлу програми та всі необхідні аргументи командного рядка.

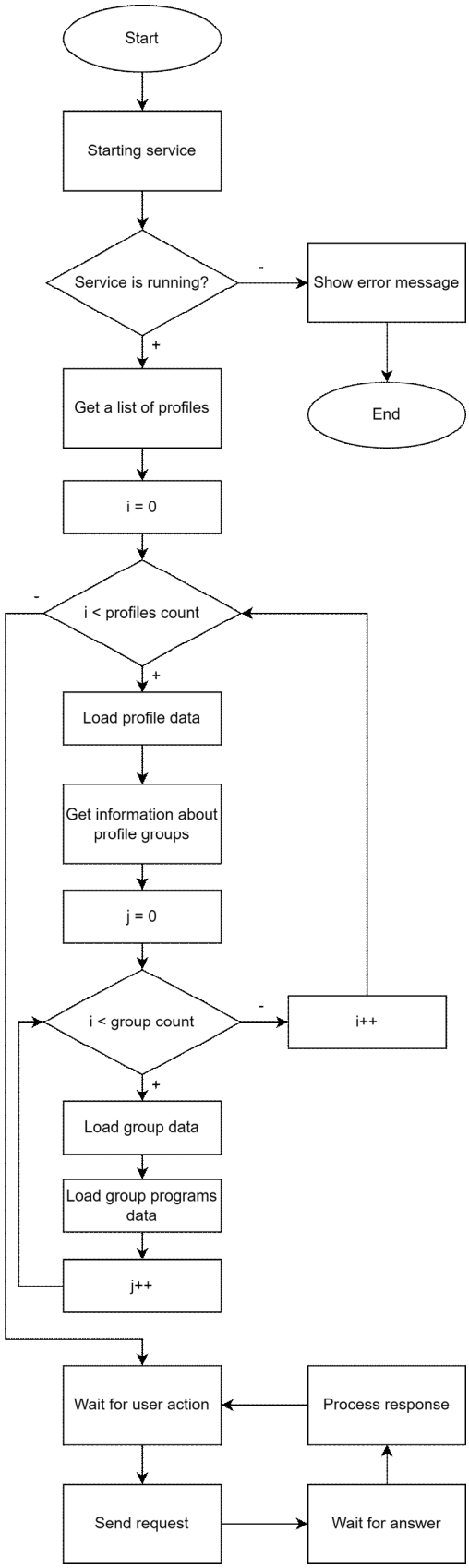

Рис. 3. Алгоритм роботи інтерфейсу

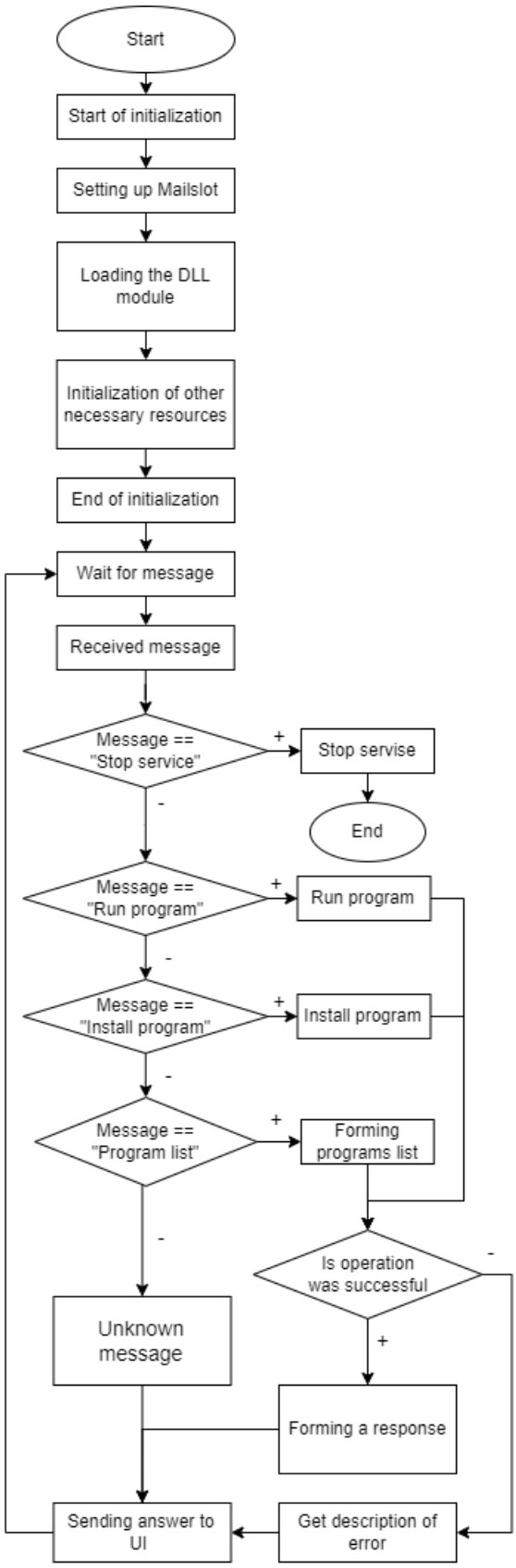

Рис. 4. Алгоритм роботи сервісу

Після отримання повідомлення служба обробляє запит, викликаючи відповідну функцію в модулі DLL. Ця функція встановлює або запускає запитувану програму і повертає код статусу, який вказує, чи була операція успішною, чи ні.

Сервіс очікує, що ця функція буде спроєктована так, щоб обробляти всі можливі помилки та винятки, які можуть виникнути під час встановлення або запуску програми. Для виявлення та вирішення проблем з інсталяцією у режимі реального часу використано передові алгоритми, що забезпечують безперебійну та ефективну інсталяцію.

Якщо запитувана дія успішна, служба надсилає користувачеві інтерфейсу повідомлення-відповідь про те, що дію було успішно завершено. Повідомлення-відповідь містить додаткову інформацію, наприклад, PID запущеного процесу. Служба очікує, що інтерфейс користувача отримає і обробить це повідомлення-відповідь вчасно, оновивши інтерфейс користувача відповідно.

Якщо запитувана дія не виконується, служба надсилає інтерфейсу користувача повідомлення-відповідь із зазначенням причини невдачі. Повідомлення-відповідь містить код помилки або детальніше повідомлення про помилку. Сервіс очікує, що інтерфейс користувача швидко отримає і обробить це повідомлення-відповідь, показавши користувачеві відповідне повідомлення про помилку.

Після того, як запит оброблено і відповідь надіслано, сервіс очікує на наступне повідомлення, яке надійде до поштової скриньки. Цей процес триває доти, доки сервіс не буде вимкнено або доки він не зіткнеться з невиправною помилкою.

Отже, робота сервісу спеціалізованої програми автоматичного встановлення та запуску утиліт у середовищі Windows полягає в очікуванні надходження повідомлень до поштової скриньки, обробленні цих повідомлень із викликанням відповідної функції в DLL-модулі та надсиланні відповідного повідомлення назад до користувацького інтерфейсу. Сервіс очікує, що всі повідомлення міститимуть всю необхідну інформацію, а користувацький інтерфейс вчасно оброблятиме повідомлення-відповіді. Дотримуючись цих очікувань, сервіс може забезпечити надійну та ефективну платформу для встановлення та запуску утиліт у середовищі Windows, що забезпечить істотне підвищення ефективності та успішності використання ресурсів.

Виділення логіки в сервіс

Відокремлення логіки від сервісу в сфері встановлення та управління програмним забезпеченням дає кілька очевидних переваг, які в сукупності сприяють покращенню роботи користувачів та адміністраторів. Такий підхід сприяє ефективному виконанню програм, ненав’язливому оновленню, регулярній перевірці оновлень та віддаленому адмініструванню, спрощуючи загальний процес для користувачів та адміністраторів.

Однією з помітних переваг такого підходу є можливість запускати програми з підвищеними правами адміністратора, коли це необхідно. Завдяки виділенню логіки в окремий сервіс програмні додатки, які потребують вищих привілеїв, можна легко встановлювати та керувати ними. Отже, користувачі можуть виконувати важливі завдання, не стикаючись з обмеженнями доступу, що покращує загальний досвід роботи.

На додаток, відокремлення логіки в сервіс дає змогу автоматично оновлювати репозиторій програм у фоновому режимі. Ця функція гарантує, що користувачі мають доступ до найновіших версій та оновлень програмного забезпечення без ручного втручання. Завдяки автоматизації цього процесу користувачі можуть отримати вигоду від постійно оновлюваного репозиторію без жодних перешкод для виконання поточних завдань.

Крім того, завдяки логічному поділу на сервіси система може періодично перевіряти наявність оновлень програмного забезпечення, повідомляючи користувачів про появу нових версій. Ця функціональність дає користувачам змогу залишатися в курсі останніх оновлень і підтримувати найактуальніші версії свого програмного забезпечення, що в кінцевому підсумку підвищує продуктивність, безпеку і стабільність програмного забезпечення.

Крім того, значною перевагою відокремлення логіки в сервіс є можливість віддалено отримувати команди від адміністратора компанії. Ця можливість дає адміністраторам змогу керувати інсталяціями, оновленнями та конфігураціями програмного забезпечення на різних пристроях в організації. Централізуючи контроль і спрощуючи віддалене керування, адміністратори можуть гарантувати, що всі пристрої працюють із найновішим програмним забезпеченням і дотримуються організаційних політик, що в підсумку підвищує ефективність і безпеку в усій організації.

Управління пакетами

Програма автоматичного встановлення та запуску утиліт пропонує функцію керування пакетами, призначену для користувачів, які потребують більше, ніж можуть надати готові пакети. Ця функція дозволяє користувачам створювати, організовувати та встановлювати свої програмні пакети, задовольняючи їхні конкретні потреби та вподобання, одночасно покращуючи загальний користувацький досвід.

Створення пакунків

Користувачі, які не можуть знайти відповідних готових пакунків, можуть створювати свої пакунки на цій платформі. Для створення пакунка вони можуть вибрати одне з різних джерел встановлення, зокрема завантаження архіву інсталятора або виконуваного файлу (EXE) безпосередньо з вебсайту розробника, доступ до нього з репозиторію GitHub або розміщення його на локальному сервері. Така гнучкість забезпечує оптимальний користувацький досвід, пристосований до індивідуальних вимог.

Для тих, хто зацікавлений у портативних версіях програмного забезпечення, користувачі повинні вказати місце розташування архіву портативної версії та виконуваного файлу в своєму пакунку. Ця функція дає змогу запускати програми без встановлення їх на хост-машині, забезпечуючи додаткову гнучкість і простоту використання.

Профілі

Функція профілів платформи спрощує організацію та встановлення декількох пакетів, які створив користувач. Використовуючи профілі, користувачі можуть встановлювати цілий набір персоналізованих програм одним кліком, заощаджуючи час і зусилля.

Однією із ключових переваг використання функції профілю в новій платформі є можливість ефективно керувати процесом інсталяції та спростити його. Завдяки можливості завантажувати наступну програму під час встановлення поточної платформа ефективно використовує час очікування для підготовки наступних пакетів.

Цей паралельний процес завантаження та встановлення істотно прискорює загальний процес інсталяції, зменшуючи час, витрачений на очікування, поки кожна програма буде послідовно завантажена та встановлена. Автоматизуючи та оптимізуючи процес, платформа підвищує продуктивність користувачів і забезпечує плавніше та ефективніше встановлення декількох програм одночасно.

Щоб створити профіль, користувачі можуть виконати такі кроки:

1. Створіть новий профіль: призначте унікальне ім’я та короткий опис, що описує мету профілю.

2. Додати пакунки: користувачі можуть додати свої пакунки до профілю, вибравши бажане джерело встановлення і надавши необхідну інформацію для кожного пакунка, зокрема, повинен пакунок використовувати портативний чи стандартний спосіб встановлення.

3. Впорядкувати пакунки: Користувачі можуть впорядкувати свої пакунки у профілі відповідно до своїх уподобань, наприклад, класифікувати їх за функціями, пріоритетами або іншими власними критеріями.

4. Встановлення в один клік: Заповнивши профіль, користувачі можуть встановити всі пов’язані з ним пакунки одним кліком. Платформа керує всім процесом встановлення, від керування залежностями до налаштування параметрів, забезпечуючи належне встановлення кожного пакунка.

Функція керування пакетами у спеціалізованій програмі автоматичного встановлення та запуску утиліт у середовищі Windows покликана забезпечити комплексне та ефективне рішення для користувачів з унікальними вимогами. Завдяки підтримці різних джерел інсталяції, портативних версій програмного забезпечення, управлінню профілями та можливості створювати персоналізовані пакунки, ця платформа дає змогу користувачам легко керувати та інсталювати свої програмні утиліти, задовольняючи широкий спектр потреб.

Ліцензування програмного забезпечення

Використовуючи спеціалізовану програму автоматичного встановлення та запуску утиліт в середовищі Windows, необхідно розуміти і дотримуватися ліцензійних угод, пов'язаних із програмами, що встановлюються. Завантажуючи та встановлюючи будь-які програми за допомогою цієї утиліти, користувачі визнають і погоджуються з тим, що вони прочитали і прийняли ліцензійну угоду відповідного постачальника програм.

- Відповідальність за ліцензування

Користувач відповідає за дотримання ліцензійних умов кожного програмного пакета, який він завантажує та встановлює через платформу. Це передбачає, але не обмежується, дотриманням обмежень на використання, розповсюдження та модифікацію. Користувачі також повинні знати про будь-які обмеження, накладені безкоштовними або пробними версіями, а також про необхідність придбання ліцензії для комерційного використання або для розблокування додаткових функцій.

- Доступ до ліцензійних угод

Більшість постачальників програмного забезпечення вводять ліцензійну угоду в процес інсталяції або роблять її легкодоступною на своєму офіційному вебсайті. Користувачі повинні переглянути ці угоди перед встановленням будь-якого програмного забезпечення, щоб переконатися, що вони розуміють і погоджуються з умовами та положеннями, які виклав постачальник.

- Дотримання правових норм

Недотримання ліцензійних умов може призвести до юридичних наслідків, таких як штрафи, судові позови або покарання. Використовуючи спеціалізовану програму автоматичного встановлення та запуску утиліт, користувачі неявно погоджуються дотримуватися ліцензійних угод кожного встановленого програмного пакета і брати на себе будь-яку юридичну відповідальність за їх недотримання.

Розробка алгоритму програми

Спеціалізована програма автоматичного встановлення та запуску утиліт у середовищі Windows складається із трьох основних компонентів: модуля DLL, сервісу та інтерфейсу користувача (UI). Крім того, платформа використовуватиме формат JSON для обміну повідомленнями між компонентами. Кожен компонент відіграє важливу роль у загальній функціональності платформи, і вони взаємодіють один з одним різними способами. Опишемо детальніше кожен компонент.

1. Модуль DLL є ядром програмної платформи. Він реалізує всю логіку запуску та встановлення програм, а також відповідає за опрацювання запитів від сервісу. Модуль DLL написаний на мові C/C++ та скомпільований у файл бібліотеки з динамічним підключенням (.dll).

Модуль використовує Win32 API та формат JSON для взаємодії. Він експортує набір функцій, які може викликати сервіс, такі як RunProgram та InstallProgram. Ці функції отримують вхідні параметри у форматі JSON, які передбачають шлях до виконуваного файлу програми або посилання на інсталятор, будь-які необхідні аргументи командного рядка та тип дії, яку потрібно виконати (запустити або встановити). Модуль DLL повертає вихідні значення у форматі JSON, які вказують на успішність виконання запитуваної дії.

2. Сервіс є фоновим процесом, який працює у середовищі Windows. Він відповідає за отримання запитів від інтерфейсу користувача або віддаленого адміністратора, перенаправлення цих запитів до модуля DLL і повернення результатів запитувачу. Сервіс написаний на C/C++ і скомпільований у виконуваний файл (.exe).

Сервіс використовує Win32 API, TCP/IP-сокети та формат JSON для взаємодії з інтерфейсом користувача та модулем DLL. Він прослуховує вказаний порт для вхідних з'єднань TCP/IP-сокетів і створює окремий потік для кожного вхідного з'єднання. Кожен потік обробляє один запит і взаємодіє з модулем DLL за допомогою викликів функцій у форматі JSON.

Коли сервіс одержує запит від інтерфейсу користувача, він видобуває параметри з JSON-повідомлення і передає їх відповідній функції в модулі DLL. Потім сервіс чекає на результат від модуля DLL, який також має формат JSON. Отримавши результат, сервіс упаковує його в JSON-повідомлення-відповідь і надсилає назад в інтерфейс користувача;

3. Інтерфейс користувача (UI) складається з декількох елементів, розташованих у певній послідовності (рис. 5). Основним елементом інтерфейсу користувача є вікно, яке містить панель у верхній частині з кількома меню, налаштуваннями, параметрами перегляду та іншими елементами керування. Ці елементи керування призначені для того, щоб дати змогу користувачеві взаємодіяти з програмним додатком різними способами, такими як зміна зовнішнього вигляду інтерфейсу, зміна налаштувань або вибір різних представлень. Це забезпечило істотне зменшення складності встановлення, що спростило роботу користувача.

Для простоти використання платформа надає інтуїтивно зрозумілий користувацький інтерфейс, що пришвидшує навчання, пропонує контекстозалежну допомогу та підказки. Платформа також містить рекомендації щодо використання утиліт і перевірку сумісності, що зменшує кількість конфліктів програмного забезпечення. Підтримка декількох мов і варіантів локалізації забезпечує доступність для глобальної бази користувачів.

Під панеллю міститься панель вибору профілю, яка надає користувачеві список доступних профілів на вибір. Кожен профіль являє собою набір налаштувань, параметрів і встановлених програм, які відповідають потребам користувача. Вибравши профіль, користувач може отримати доступ до певного набору програм і налаштувань, які відповідають його вимогам.

Під панеллю вибору профілю розміщений список програм, відсортованих за групами. Групи програм зазвичай організовані за категоріями, такими як продуктивність, розваги або утиліти, щоб полегшити користувачеві пошук потрібних програм. Програми в кожній групі можна вибирати окремо або як групу, що дає користувачеві змогу встановити одну програму, цілу групу або весь профіль одним клацанням миші. Ця функція дозволила істотно зменшити кількість кліків користувача та зменшити тривалість інсталяції.

Коли користувач вибирає програму для запуску або встановлення, інтерфейс пакує необхідні параметри в JSON-запит і надсилає його сервісу через сокетне з'єднання. Це повідомлення містить всю необхідну інформацію для обробки запиту сервісом, наприклад, назву програми, версію та будь-які додаткові налаштування або опції.

Потім інтерфейс очікує на повідомлення-відповідь у форматі JSON, яке вказує на те, чи виконана запитувана дія успішно, чи ні. Якщо запит був успішним, повідомлення-відповідь міститиме повідомлення-підтвердження, а інтерфейс відобразить користувачеві повідомлення із зазначенням результату. Якщо запит не був успішним, повідомлення у відповідь міститиме повідомлення про помилку, а користувацький інтерфейс покаже користувачеві відповідне повідомлення про помилку.

Безшовна інтеграція з операційною системою Windows забезпечує сумісність і підтримку застарілих і майбутніх утиліт, мінімізуючи потенційні конфлікти і покращуючи загальний користувацький досвід. Ці покращення, у поєднанні з підвищенням зручності використання, істотно покращили процес встановлення утиліти, зробивши його ефективнішим, зручнішим та доступнішим для широкого кола користувачів.

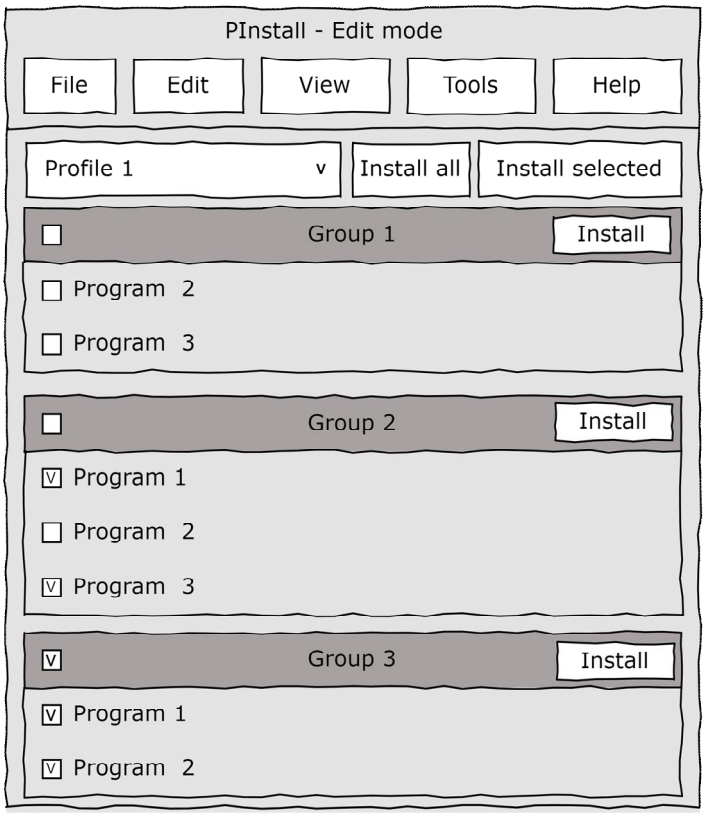

Рис. 5. Макет інтерфейсу

Внутрішня архітектура UI

В управлінні даними ефективність та організація інформації є ключовими факторами успіху будь-якої системи. Один із підходів, який допомагає досягти цієї мети, полягає в застосуванні моделі, що відділяє дані від їх представлення. Такий метод дає змогу змінювати та адаптувати візуальне представлення без потреби втручатися в основну структуру даних. Це забезпечує гнучкість і масштабованість, які необхідні для сучасних динамічних додатків.

Модель даних

Модель [12] виконує важливу функцію контролю за доступом до інформації. Вона управляє всіма основними операціями з даними, такими як вставка, оновлення, видалення та читання. Крім того, вона може виконувати додаткові завдання, зокрема сортування та фільтрацію інформації, забезпечуючи тим самим не тільки зберігання, але й організацію даних.

Однією з найважливіших характеристик такої моделі є її здатність надсилати повідомлення про зміни, що відбуваються у даних. Це означає, що кожен раз, коли інформація змінюється, інтерфейс користувача може миттєво оновлюватись для відображення цих змін, забезпечуючи актуальність та консистентність представлення.

Незалежність від конкретного джерела даних є ще однією значною перевагою. Це означає, що модель може бути інтегрована з будь-яким типом сховища даних – реляційні бази даних, документо-орієнтовані бази, файлові системи чи навіть віддалені сервіси. Ця гнучкість робить її універсальним інструментом у руках розробників.

Крім того, модель сприяє легшій інтеграції з компонентами інтерфейсу користувача та делегатами, які відповідають за відображення та редагування даних. Це дає змогу створювати інтуїтивно зрозумілі та візуально привабливі інтерфейси, які можна легко адаптувати під різні вимоги користувачів.

Важливо також зазначити, що модель дає змогу вбудовувати валідацію даних безпосередньо у їх оброблення. Це підвищує надійність системи, забезпечуючи, що тільки коректні та очікувані дані будуть прийняті або змінені.

Такий інтегрований підхід забезпечує не лише зручність і ефективність управління даними, але й поліпшує якість загального користувацького досвіду, знижуючи складність та збільшуючи продуктивність системи загалом.

Архітектура

Архітектура, наведена на рис. 6, є яскравим прикладом модульної структури управління даними. Вона розглядає дані через три основні сегменти: профілі, групи та програми. Кожен сегмент структурований і має власні унікальні характеристики, які дають змогу ефективно організовувати дані й управляти ними відповідно до потреб користувача.

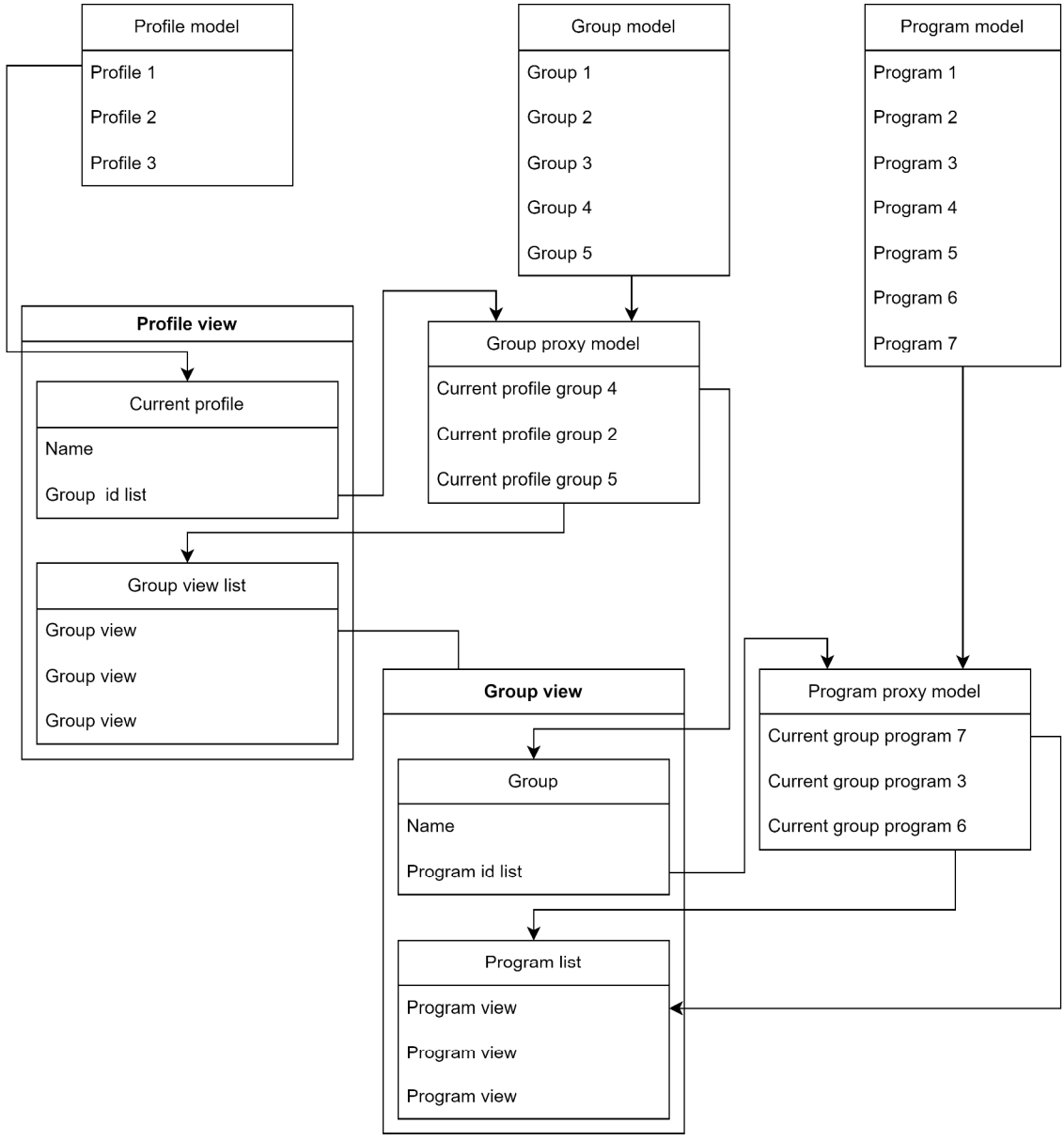

Рис. 6. Архітектура UI

У серці системи – профілі, що функціонують як базові контейнери для індивідуальних наборів даних. Кожен профіль має свій ідентифікатор і асоційований з списком груп, що дозволяє групувати пов'язану інформацію для кращої організації та доступності. Подання профілю дає змогу відображати деталі, важливі для користувача, зокрема назву та списки відображень пов'язаних груп.

Групи є другим важливим елементом архітектури, вони призначені для згрупованого управління окремими програмами. Кожна група має власне представлення, що містить назву, список ідентифікаторів програм, які вона об'єднує. Такий підхід забезпечує чітке визначення зв'язків між різними частинами системи.

Третім ключовим компонентом є програми, кожна з яких містить усю необхідну інформацію для відображення та запуску.

Профіль та групи мають свою проксімодель, яка є специфічним вибором елементів з головної моделі, що дає змогу сортувати елементи та управляти ними в межах цього об'єкта. Такий підхід спрощує управління індивідуальними компонентами без впливу на загальну структуру.

Проксімоделі [13] відіграють важливу роль у цій архітектурі, вони функціонують як фільтри або переглядачі, що вибірково відображають дані з основних моделей. Це дає змогу кастомізувати подання даних, надаючи можливість зосередитися на найважливіших аспектах без необхідності опрацювання всієї інформації. Проксімоделі можуть вільно переорганізовувати елементи відповідно до потреб користувача, пропонуючи гнучкість у сортуванні та пріоритезації.

Така архітектура підсилює важливість взаємозв'язку між різними рівнями даних. Вона стимулює створення інтуїтивно зрозумілих інтерфейсів, полегшує управління даними і підвищує загальну продуктивність системи. Структурованість і гнучкість цієї архітектури відіграють ключову роль у впровадженні інформаційних систем великого масштабу, забезпечуючи ефективність і адаптивність до змінних потреб користувачів.

Провайдери даних

Провайдери даних у системі функціонують як мости між основними моделями системи та їх джерелами даних. Ці елементи критично важливі для забезпечення гнучкості та масштабованості архітектури, даючи змогу системі інтегруватися з різними типами джерел даних і відповідати на різноманітні запити користувачів.

Класи провайдерів даних:

5. Файловий провайдер даних: Цей клас відповідає за зчитування даних із локальних або мережевих папок. Він може обробляти файли різних форматів, екстрактуючи інформацію, яка буде використана для заповнення моделей профілів, груп та програм. Цей провайдер може містити механізми для виявлення структури папок, розпізнавання типів файлів і вилучення даних.

6. Серверний провайдер даних: Серверний провайдер підключається до централізованих баз даних або інтерфейсів API, забезпечуючи доступ до даних, що зберігаються на серверах. Він відправляє запити до сервера й обробляє отриману відповідь, конвертуючи її в структури, необхідні для моделей системи.

7. Віддалений провайдер даних: Цей клас спеціалізується на доступі до даних, які зберігаються на віддалених комп'ютерах, наприклад, через протоколи, такі як FTP або SSH. Він забезпечує передавання файлів або потоків даних і може передбачати додаткові заходи для шифрування та безпеки під час передавання даних.

Кожен із цих провайдерів може бути реалізований так, щоб надавати інформацію виключно про профілі, групи або програми, або будь-яку їх комбінацію. Це забезпечується використанням спеціалізованих інтерфейсів або базових класів, від яких провайдери можуть успадковувати необхідну поведінку:

- Базовий провайдер: усі провайдери можуть успадкувати від базового класу, який визначає стандартний інтерфейс для отримання даних. Він може містити методи для ініціалізації з'єднань, відправлення запитів і опрацювання відповідей.

• Спеціалізований провайдер: Для кожного типу даних (профілі, групи, програми) можуть бути створені окремі класи, які реалізують специфічну логіку, необхідну для взаємодії з конкретним типом даних. Наприклад, провайдер профілів знає, як парсити дані про них, тоді як провайдер програм вміє взаємодіяти з даними про програмне забезпечення.

• Композитний провайдер: іноді корисно мати провайдер, який використовує декілька інших провайдерів для збирання даних з різних джерел і об'єднання їх в одину консистентну структуру.

Такі класи провайдерів даних є фундаментальними для забезпечення адаптивності та розширюваності системи. Вони дають змогу ізолювати логіку обробки даних від загальної бізнес-логіки системи, що сприяє підтримці та модернізації системи.

Комплексний порівняльний аналіз

Удосконалення нової платформи спрямовані на усунення недоліків наявних платформ, таких як Windows Package Manager, Chocolatey, Ninite та платформа Windows Store. Вона пропонує комплексний репозиторій програмного забезпечення, інтуїтивно зрозумілий користувацький інтерфейс, адаптивність, розширюваність, підтримку програм, що потребують адміністративного доступу, а також вдосконалене управління пакунками та підтримку користувачів. Підвищення складності інсталяції, кількості кліків користувача, часу інсталяції та використання системних ресурсів демонструють, що нова платформа значно перевершує наявні системи в різних аспектах.

Порівняння нової платформи з чотирма наявними системами здійснено із встановленням десяти програм на одному комп'ютері. Для підтримання однакового тестового середовища та забезпечення точного порівняння ефективності систем інсталяції програм перед встановленням будь-яких утиліт на віртуальній машині (ВМ) було створено знімок з віртуальної машини. Цей знімок слугує чистою відправною точкою для кожної системи інсталяції, що дає змогу об'єктивно оцінити їхню продуктивність.

Знімок містить такі елементи:

8. Конфігурація віртуальної машини: віртуальна машина налаштована на два ядра з тактовою частотою 2 ГГц, щоб мінімізувати вплив апаратних відмінностей на результати.

9. Базовий стан системи: Знімок фіксує стан системи до встановлення будь-яких інсталяційних утиліт або програм, гарантуючи, що кожна інсталяційна система починається з однакового чистого середовища.

Для кожної інсталяційної системи ВМ повертається до чистого знімка перед початком інсталяції. Такий підхід гарантує, що системні ресурси буде виділено для інсталяції кожної утиліти без втручання інших інсталяцій або будь-яких залишкових файлів від попередніх інсталяцій.

Після повернення до знімка відповідна утиліта для встановлення (Windows Package Manager, Chocolatey, Ninite або Windows Store) завантажується і встановлюється на віртуальну машину. Там, де це можливо, були встановлені додатково версії інтерфейсу, щоб полегшити взаємодію користувача з програмою і забезпечити оптимальний користувацький досвід. Потім десять програм встановлюються одна за одною за допомогою утиліти. Цей процес повторюється для кожної інсталяційної системи, що дозволяє безпосередньо порівняти процес інсталяції та продуктивність.

Вимірювали та порівнювали такі параметри, як складність інсталяції, кількість кліків користувача, час інсталяції та використання системних ресурсів. Для кожної характеристики використано такі методи вимірювання:

10. Складність встановлення (вимірюється у кроках): оцінено із вимірюванням кількості кроків, необхідних для встановлення, ураховуючи взаємодію з користувачем та точки прийняття рішень.

11. Кліки користувачів: підраховано загальну кількість кліків користувача, необхідних для завершення процесу встановлення для кожної системи.

12. Час встановлення: виміряно загальний час, необхідний для процесу інсталяції, від запуску до завершення інсталяції для всіх десяти програм.

13. Використання системних ресурсів: простежено використання процесора та пам'яті під час процесу інсталяції та розраховано середнє використання для кожної платформи.

Варто зазначити, що для консольних версій програм, зокрема для нової утиліти, є можливість написання скриптів. Однак ця можливість не буде розглядатися в порівнянні, оскільки більшість утиліт підтримують написання скриптів, що лише зменшить кількість кроків і кліків, пов'язаних з процесом встановлення. Хоча створення сценаріїв є перевагою для деяких користувачів, це не найзручніший варіант для пересічного кінцевого користувача, який може не володіти технічними знаннями, необхідними для створення та ефективного використання сценаріїв.

Спеціалізована програма автоматичного встановлення та запуску утиліт у середовищі Windows продемонструвала істотні покращення порівняно з наявними системами. Наведемо основні висновки.

14. Складність встановлення (вимірюється в кроках):

• Нова платформа:
Крок 1: Запустіть платформу.
Крок 2: Виберіть потрібний профіль.
Крок 3: Натисніть "Встановити все", щоб ініціювати встановлення усіх десяти програм.
Разом: три кроки.

• Windows Package Manager (UI):
Крок 1: Запустіть платформу.
Крок 2: Знайдіть кожну утиліту окремо.
Крок 3: Натисніть "Встановити" для кожної утиліти.
Крок 4: Повторіть кроки 2–3 для всіх десяти програм.
Разом: 21 крок (два кроки на програму × десять програм)

• Windows Store:
Крок 1: Запустіть платформу.
Крок 2: Увійдіть за допомогою облікового запису Microsoft (якщо потрібно).
Крок 3: Знайдіть кожну утиліту окремо.
Крок 4: Натисніть "Отримати" для кожної утиліти.
Крок 5: Повторіть кроки 3–4 для всіх десяти програм.
Разом: 22 кроки (два кроки для кожної програми × десять програм)

• Chocolatey (UI):
Крок 1: Запустіть платформу
Крок 2: Знайдіть кожну утиліту окремо
Крок 3: Натисніть "Встановити" для кожної утиліти.
Крок 4: Повторіть кроки 2–3 для всіх десяти програм.
Разом: 21 крок (два кроки на програму × десять програм)

• Ninite:
Крок 1: Відвідайте вебсайт Ninite.
Крок 2: Виберіть потрібні утиліти.
Крок 3: Завантажте спеціальний інсталятор.
Крок 4: Запустіть інсталятор.
Разом: чотири кроки

15. Кліки користувачів (вимірюється у загальній кількості кліків):

• New Platform: 3 кліки
• Windows Package Manager: 29 кліків
• Windows Store: 27 кліків
• Chocolatey: 27 кліків
• Ninite: 23 кліки

16. Час встановлення (вимірюється в хвилинах):

- New Platform: 14 хвилин
- Windows Package Manager: 20 хвилин
- Windows Store: 23 хвилини
- Chocolatey: 26 хвилин
- Ninite: 22 хвилини

17. Використання системних ресурсів (вимірюється середнім використанням CPU та пам'яті):

- New Platform: 14 % CPU, 19 % пам'ять
- Windows Package Manager: 21 % CPU, 34 % пам'ять
- Windows Store: 24 % CPU, 39 % пам'ять
- Chocolatey: 29 % CPU, 44 % пам'ять
- Ninite: 24 % CPU, 38 % пам'ять

Отже, нова спеціалізована програма автоматичного встановлення та запуску утиліт у середовищі Windows пропонує істотні покращення у різних аспектах порівняно з наявними системами, такими як Windows Package Manager, Windows Store, Chocolatey та Ninite. Покращено такі аспекти:

18. Складність встановлення (вимірюється у кроках):

Нова платформа зменшує кількість кроків, необхідних для встановлення десяти програм, на 87 % порівняно з Windows Package Manager, Windows Store та Chocolatey і на 26 % порівняно з Ninite.

19. Кліки користувачів (вимірюється у загальній кількості кліків):

Нова платформа зменшує кількість кліків, необхідних для встановлення десяти програм, на 92 % порівняно з Windows Package Manager, на 91% порівняно з Windows Store та Chocolatey і на 89 % порівняно з Ninite.

20. Час встановлення (вимірюється в хвилинах):

Нова платформа зменшує час встановлення на 31 % порівняно з Windows Package Manager, на 40 % порівняно з Windows Store, на 46 % порівняно з Chocolatey та на 37 % порівняно з Ninite.

21. Використання системних ресурсів (вимірюється середнім використанням процесора та пам'яті).

Нова платформа зменшує використання процесора на 34 % порівняно з Windows Package Manager, на 42 % порівняно з Windows Store, на 52 % порівняно з Chocolatey і на 40 % порівняно з Ninite. Використання пам'яті зменшено на 45 % порівняно з Windows Package Manager, на 52 % порівняно з Windows Store, на 58 % порівняно з Chocolatey і на 49 % порівняно з Ninite.

Ці кількісні покращення стосовно ефективності, простоті використання та користувацького досвіду супроводжуються додатковими якісними перевагами, такими як посилена безпека та конфіденційність, підтримка декількох мов, можливості локалізації, можливість запускати портативні версії, створювати власні пакунки та перевіряти сумісність. Зосередженість нової платформи на вирішенні обмежень і проблем наявних систем забезпечує створення плавнішого, ефективнішого та безпечнішого рішення для користувачів, що задовольняє різноманітні потреби та вподобання в середовищі Windows.

Оцінка покращення

Цей розділ містить детальний аналіз покращень, досягнутих завдяки програмі автоматичного встановлення та запуску утиліт у середовищі Windows, з акцентом на кількісному порівнянні цієї програми з наявними рішеннями, такими як Windows Package Manager, Chocolatey, Ninite та платформа Windows Store.

22. Простота встановлення та експлуатації. Раніше користувачам доводилося шукати програми вручну, роблячи кілька рухів мишею та кліків. Завдяки спеціалізованій програмній платформі користувачі тепер можуть знаходити та вибирати потрібні програми за значно меншу кількість кліків, що істотно зменшує кількість необхідних рухів миші та клацань мишею. Таке значне покращення пояснюється

передусім спрощеним процесом інсталяції платформи, автоматизованими процедурами та чіткими інструкціями, що надаються користувачам. Крім того, програма автоматичного встановлення та запуску утиліт дає змогу створювати власні профілі для певних наборів утиліт і розпаралелювати процес інсталяції, що ще більше спрощує і прискорює інсталяцію.

23. Ефективність процесу встановлення. У традиційних методах процес інсталяції, як правило, потребував декількох ручних кроків, що забирало чимало часу і зусиль. Програма автоматичного встановлення та запуску утиліт зменшила середній час інсталяції, що істотно підвищило її швидкість. Програма також пропонує паралельну інсталяцію, що дає змогу встановлювати кілька утиліт одночасно, що ще більше підвищує ефективність. Крім того, платформа успішно зменшила потребу в ручному втручанні під час інсталяції, сприяючи безперебійній та ефективнішій роботі користувачів.

24. Користувацький досвід і простота використання. Програма автоматичного встановлення та запуску утиліт пропонує інтуїтивно зрозумілий користувацький інтерфейс, що скорочує час навчання для користувачів. Завдяки контексточутливій допомозі та підказкам зменшилася кількість запитів користувачів, що свідчить про спрощення використання. Платформа також підтримує запуск портативної версії, що полегшує користувачам управління утилітами на різних пристроях без необхідності повної інсталяції.

25. Адаптивність та розширюваність. Програма автоматичного встановлення та запуску утиліт дає змогу користувачам без особливих зусиль вводити нові програми без модифікації основного коду, що було неможливо на таких платформах, як Windows Store. У типовому випадку додавання нової програми до платформи Windows Store потребувало численних кроків, тоді як спеціалізована програмна платформа потребує лише кількох кроків з використанням конфігураційного файлу JSON, що значно зменшує кількість необхідних кроків.

Висновки

Алгоритм програми для автоматичного встановлення та запуску утиліт у середовищі Windows успішно подолав обмеження та проблеми наявних систем, зосередившись на основних цілях: спрощення інсталяції та експлуатації, підвищення ефективності процесу інсталяції, покращення зручності використання, як для технічних, так і для нетехнічних користувачів, а також забезпечення безпеки та приватності. Це комплексне рішення пропонує зручніший досвід для користувачів операційної системи Windows, задовольняючи різноманітні потреби та вподобання.

Розроблений алгоритм програми дав змогу зменшити складність встановлення на 72 %, а кількість кліків користувачів – на 90 %. Бездоганна інтеграція з операційною системою Windows забезпечує сумісність і підтримку застарілих і майбутніх утиліт, мінімізуючи потенційні конфлікти і покращуючи загальний користувацький досвід. Крім того, платформа дає користувачам змогу створювати власні пакети та профілі, швидко встановлювати певні набори утиліт, пристосовані до їхніх потреб, та керувати ними.

З погляду ефективності програма автоматичного встановлення та запуску утиліт успішно мінімізувала ручне втручання, зменшила кількість збоїв під час встановлення та оптимізувала швидкість встановлення. Програма скоротила час встановлення на 32 % і зменшила використання системних ресурсів на 35 %. Вдосконалені алгоритми були використані для виявлення та вирішення проблем встановлення ув режимі реального часу, забезпечуючи безперебійне та ефективне встановлення. Крім того, програма підтримує паралельну інсталяцію, що дає змогу встановлювати кілька утиліт одночасно, ще більше підвищуючи ефективність.

Для зручності використання платформа має інтуїтивно зрозумілий користувацький інтерфейс, що істотно пришвидшує навчання, пропонує контекстозалежну допомогу та підказки, які зменшують кількість запитів користувачів, а також враховує різні користувацькі вподобання та варіанти налаштування. Програма також надає рекомендації щодо використання утиліт і перевірку сумісності, що істотно зменшує кількість конфліктів програмного забезпечення. Підтримка декількох мов і

варіантів локалізації забезпечує доступність для глобальної бази користувачів. Крім того, програма дає змогу користувачам запускати портативну версію, що полегшує управління утилітами на різних пристроях без необхідності повної інсталяції.

Інтеграція моделей даних та провайдерів у програмі відіграла ключову роль у досягненні точного та своєчасного оновлення утиліт та їх конфігурацій. Це забезпечує постійну наявність останніх версій, надаючи користувачам надійний та ефективний досвід використання. Провайдери даних, розроблені для безшовної взаємодії з різними джерелами даних, полегшують отримання інформації в реальному часі та управління нею. Моделі використовують ці дані для надання користувачу найактуальніших і релевантних варіантів утиліт. Завдяки цьому архітектура системи значно знижує частоту помилок та підвищує загальну стабільність процесу управління утилітами.

З погляду безпеки та конфіденційності платформа надає пріоритет безпеці та конфіденційності користувачів, впроваджуючи суворі заходи безпеки та дотримуючись найкращих галузевих практик. Передбачено такі функції, як захищені канали завантаження і повне шифрування всіх конфіденційних даних користувача. Програму розроблено так, щоб забезпечити дотримання відповідних правил захисту даних, таких як GDPR, щоб захистити конфіденційність користувачів.

Удосконалення програми спрямовані на усунення недоліків наявних платформ, таких як Windows Package Manager, Chocolatey, Ninite і платформа Windows Store, ураховуючи комплексне сховище програмного забезпечення, інтуїтивно зрозумілий інтерфейс користувача, адаптивність, розширюваність, підтримку програм, що потребують адміністративного доступу, а також розширені можливості управління пакетами і підтримки користувачів. Адаптивність і розширюваність програми дають користувачам змогу без особливих зусиль вводити нові програми без модифікації основного коду, спрощуючи додавання та розповсюдження нових програм за допомогою конфігураційних файлів JSON.

Список літератури

1. *Titus C. (2021). Microsoft's New Windows Package Manager: A Deep Dive. IEEE Consumer Electronics Magazine [Online], 118–121. DOI: https://doi.org/ 10.1109/MCE.2021.3069863*
2. *Minocha B., Goyal N., (2018). A Comparative Study of Windows Store Apps for Education and Business Purpose. IEEE 2nd International Conference on Information and Computer [Online], 232–238. DOI: https://doi.org/ 10.1109/ICICCT.2018.8473065*
3. *Chocolatey (2022). Chocolatey Software | Packages. Chocolatey [Online]. Available: https://chocolatey.org/packages (Accessed 02/20/2023)*
4. *Ninite (2022). Ninite - Install or Update Multiple Apps at Once. Ninite [Online]. Available: https://ninite.com/ (Accessed 02/20/2023)*
5. *Zaiats T., Bilenko, V., Hlukhov, V. (2022). Features of Using Large Keys in "Kalyna" Algorithm. Advances in Cyber-Physical Systems, Vol. 7, No. 1, 55–62. DOI: https://doi.org/10.23939/acps2022.01.055*
6. *Cui, W., Fu, S., Hu, Z., (2022). Dynamic-Link Library. Encyclopedia of Ocean Engineering. Springer, Singapore [Online], 375–375. DOI: https://doi.org/10.1007/978-981-10-6946-8_300187*
7. *T. Bray, Ed. (2017). The JavaScript Object Notation (JSON) Data Interchange Format. IETF [Online], 1–16. DOI: https://doi.org/10.17487/RFC8259*
8. *Cplusplus.com, (2022). C++ Standard Library – System Programming. cplusplus.com. [Online]. Available: https://www.cplusplus.com/doc/tutorial/system/ (Accessed 02/20/2023)*
9. *Microsoft (2022). C# Programming Guide. Microsoft Docs. [Online]. Available: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/index (Accessed 02/20/2023)*
10. *Solis D. (2009). Introduction to Windows Presentation Foundation. Illustrated WPF [Online], 1–16. DOI: https://doi.org/10.1007/978-1-4302-1911-8_1*
11. *Roof L., Fergus D., (2003). The .NET Compact Framework. The Definitive Guide to the .NET Compact Framework [Online], 1–32. DOI: https://doi.org/10.1007/978-1-4302-0789-4_1*
12. *Schlee M., (2018). Interview, or model-representation. Qt 5.10 Professional Programming with C++ [Online],189–217. ISBN: 978-5-9775-0010-6*
13. *Erich G., Richard H., Ralph J., John V., (1994). Proxy pattern. Design Patterns:Elements of Reusable Object-Oriented Software [Online], 112–120. ISBN: 0-201-63361-2*

ALGORITHMIC AND SOFTWARE TOOLS FOR AUTOMATIC INSTALLATION AND STARTING OF UTILITIES IN THE WINDOWS ENVIRONMENT

O. Yatskiv, Y. Klushyn

Lviv Polytechnic National University,
Computer Engineering Department

In today's world, the automation of installation and management processes of software in the Windows environment is a key element in ensuring user convenience and efficiency. Within this direction, a specialized program has been developed aimed at significantly simplifying the processes of installation and management of utilities. The program is equipped with an intuitive user interface that facilitates seamless integration with the operating system and provides users with easy access to the necessary tools.

The architecture of the program is constructed using modular models and flexible data providers that ensure dynamic installation and updating of utilities. The core models define the data structure and the logic of interaction with the user interface and other system components, while the data providers are adapted to read, update, and distribute information from various sources, including local files, servers, and remote systems.

The implementation of the program involved a detailed analysis of existing utility management methods, with attention to identified shortcomings and limitations. In view of this, requirements for the program's functionality were formulated that would ensure increased productivity and reduce potential errors and disappointments during the installation and configuration of utilities.

Thorough testing and user surveys helped assess the program from the perspectives of ease of use, functionality, and overall satisfaction. The findings of the study revealed a high level of program effectiveness, confirming its ability to achieve set goals and positively influence the user experience with utilities.

Key words: Windows; service; software; work optimization; performance.