DOI 10.34185/1562-9945-1-156-2025-02
УДК 004.42

А.С. Ганжа, С.В. Антоненко

АВТОМАТИЗАЦІЯ РОЗРОБКИ WIN32 ДЕСКТОПНИХ ДОДАТКІВ: ПРАКТИЧНІ ПІДХОДИ І СТРАТЕГІЇ

Анотація. Пропонується огляд сучасних підходів до автоматизації розробки Win32 десктопних додатків з використанням практик DevOps. Описуються ключові етапи, такі як Continuous Integration (CI) та Continuous Deployment (CD), які допомагають розробникам автоматизувати збірку, тестування та розгортання додатків, знижуючи ризики помилок. Особлива увага приділяється автоматизованому тестуванню, яке включає модульні тести, функціональні тести та тести продуктивності для забезпечення стабільності додатків. Розглядаються підходи до інфраструктури як коду (IaC), які дозволяють автоматизувати налаштування та керування середовищами розробки. Серед інструментів виділяються Terraform, Ansible, та PowerShell DSC. Також включено розгляд збереження документів у хмарному сховищі з використанням Minio, що забезпечує безпечне та масштабоване зберігання даних. Завдяки цим практикам та інструментам, розробка Win32 десктопних додатків стає більш ефективною, надійною та швидкою.

Ключові слова: *розробка програмного забезпечення, автоматизація розробки програмного забезпечення, DevOps, Continuous Integration (CI), Continuous Deployment (CD), інфраструктура як код, автоматизоване тестування, Terraform, Ansible, PowerShell DSC, Minio, збереження документів у хмарі, хмарне сховище, розгортання, Windows, хмарні технології.*

Постановка пробеми. Розробка Win32 десктопних додатків є складним процесом, який вимагає значних зусиль з боку розробників для забезпечення їхньої якості, стабільності та продуктивності. Зі зростанням вимог до функціональності, швидкості випуску оновлень і надійності, зростає необхідність в автоматизації процесів розробки, тестування та розгортання. Без впровадження автоматизації можливі часті помилки, затримки у випуску та труднощі з підтримкою високих стандартів якості. Постає питання, як ефективно впроваджувати практики DevOps для автоматизації цих процесів у середовищі розробки Win32 додатків, мінімізуючи людські помилки та підвищуючи ефективність.

Аналіз останніх досліджень і публікацій. Автоматизація розробки програмного забезпечення на основі DevOps-практик стала ключовою темою сучасних досліджень. Одним із напрямків є автоматизація управління безпекою, зокрема у промислових DevOps-проєктах. У статті M. Voggenreiter та ін. [1] розглядається впровадження мето-

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

дології для автоматизації управління безпековими виявленнями в DevOps-середовищі. Автори пропонують систему Security Flama, яка використовує семантичну базу знань для управління безпековими ризиками. На основі дослідження двох промислових проєктів було підтверджено, що впровадження цієї методології покращує управління безпековими вразливостями та сприяє зменшенню ризиків у процесі розробки. Хоча дослідження орієнтоване на промислові DevOps-проєкти, підходи до автоматизації безпеки можуть бути корисними і для Win32-додатків, де безпека залишається критичним аспектом розробки.

Дослідження S. Duque Anton та ін. [2] зосереджено на проблемах автоматизації безпеки в розподілених IoT-системах, інтегрованих у DevOps-середовище. Хоча фокус статті на IoT, багато інструментів і підходів, що описані для автоматизації безпеки та процесів розгортання, можуть бути адаптовані для десктопних Win32-додатків. Зокрема, стаття описує інструменти, що використовуються на різних етапах DevOps-процесів, і пропонує рішення для підвищення їх безпеки, враховуючи специфіку розподілених систем. Інтеграція таких методів може забезпечити надійність і для Win32-додатків, особливо у випадках складної архітектури чи розподілених середовищ.

Іншим важливим напрямком є дослідження ролі штучного інтелекту (AI) у DevOps-процесах. У статті M. Alenezi та ін. [3] розглядається, як AI може значно покращити автоматизацію процесів, таких як тестування, моніторинг та аналіз логів. AI допомагає виявляти потенційні проблеми на ранніх стадіях і оптимізувати роботу програмного забезпечення. Для Win32-додатків ці технології можуть бути використані для покращення якості продукту через вдосконалене тестування та прогнозування можливих помилок. Стаття підкреслює важливість інтеграції AI у DevOps для підвищення продуктивності команд і забезпечення швидкої реакції на зміни.

Культурні аспекти впровадження DevOps також відіграють важливу роль у успішній інтеграції цих практик. У дослідженні W. Luz та ін. [4] увага приділяється не тільки автоматизації інструментів, але й співпраці між командами розробки та експлуатації. Автори виявили, що успішна реалізація DevOps залежить не стільки від використання конкретних інструментів, скільки від побудови ефективної культури взаємодії в організації. Це дослідження охоплює аналіз впровадження DevOps у 15 компаніях із різних країн та пропонує модель успішного впровадження практик DevOps. Впровадження таких культурних змін може бути корисним і для команд, що працюють з Win32-додатками, оскільки вони часто включають міждисциплінарну співпрацю між розробниками, тестувальниками та операційними командами.

Загалом, хоча більшість досліджень присвячена DevOps у загальному контексті або на прикладі IoT та промислових систем, принципи автоматизації, управління безпекою та культурних змін можуть бути успішно адаптовані і для Win32-додатків, що відкриває простір для подальших досліджень у цьому напрямку.

Мета дослідження. Вивчення та впровадження сучасних практик автоматизації розробки Win32 десктопних додатків з використанням DevOps-методологій. Дослідження спрямоване на аналіз основних етапів автоматизації, таких як Continuous Integration (CI), Continuous Deployment (CD), автоматизоване тестування, а також застосу-

вання інфраструктури як коду (IaC) для управління середовищами розробки. Особлива увага приділяється використанню інструментів, таких як Terraform, Ansible, PowerShell DSC, та зберіганню даних у хмарі за допомогою Minio.

Вступ. Розроблення десктопних додатків для операційної системи Windows завжди була важливим аспектом програмної інженерії. З плином часу вимоги до додатків постійно зростають, а також змінюються стратегії розробки та впровадження. У сучасному світі важливо не лише швидко розробляти, але і ефективно впроваджувати додатки, щоб вони задовольняли потреби користувачів та відповідали вимогам бізнесу.

Ця стаття присвячена розгляду практичних підходів і стратегій, які допоможуть розробникам десктопних додатків на платформі Windows максимально автоматизувати розробку проєктів, забезпечуючи ефективність та швидкість у розробці.

Розглядаються важливі аспекти DevOps такі як Continuous Integration (CI) і Continuous Deployment (CD), автоматизоване тестування, використання інфраструктури як коду та інші практики, які можуть бути використані для оптимізації процесу розробки Win32-десктопних додатків.

Ця стаття призначена для розробників, які зацікавлені в покращенні своїх навичок у розробці десктопних додатків та впровадженні сучасних практик розробки. Безперечно, правильне застосування автоматизації може значно спростити і прискорити процес розробки, що в свою чергу дозволить команді більш ефективно відповідати на вимоги ринку та задовольняти потреби користувачів.

Важливість автоматизації в розробці Win32 десктопних додатків. Розробка десктопних додатків для платформи Windows може бути складним та часовитратним процесом, особливо з урахуванням постійно зростаючих вимог до функціональності та якості продукту. У таких умовах важливо мати ефективні інструменти та стратегії, які дозволяють прискорити процес розробки та забезпечити якість продукту.

Автоматизація в розробці є однією з ключових стратегій, яка дозволяє досягти цих цілей. Шляхом автоматизації різних аспектів розробки, таких як збірка, тестування, розгортання та моніторинг, розробники можуть значно скоротити час витрачений на ці процеси та знизити ймовірність помилок.

Одним з основних переваг автоматизації є збільшення швидкості розробки. Замість того, щоб витрачати час на рутинні завдання, розробники можуть сконцентруватися на більш важливих аспектах розробки, таких як розробка нового функціоналу або вдосконалення інтерфейсу користувача.

Крім того, автоматизація дозволяє забезпечити більшу стабільність та надійність додатків. Автоматизовані тести можуть виявляти помилки та проблеми раніше, ніж вони вплинуть на кінцевого користувача, що дозволяє швидко їх виправляти та підвищує якість продукту.

Не менш важливим є забезпечення консистентності та відновлюваності процесів розробки. Завдяки автоматизації, кожен етап розробки може бути чітко визначений і повторюваний, що сприяє стандартизації та полегшує спільну роботу команди розробників [13].

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Continuous Integration (CI) для Win32 десктопних додатків. Continuous Integration (CI) є однією з ключових практик розробки програмного забезпечення, яка полягає в автоматизованому процесі збірки та перевірки додатка при кожній зміні вихідного коду. Для Win32 десктопних додатків, CI може стати незамінним інструментом для забезпечення якості продукту та швидкості розробки.

Основні складові Continuous Integration для Win32 десктопних додатків включають:

1. **Автоматизована збірка додатку**: Налаштування скриптів або конфігураційних файлів, які автоматично збирають додаток з вихідного коду при кожній зміні. Це дозволяє впевнитися, що новий код компілюється правильно та не видає помилок.

2. **Модульні тести**: Включення автоматизованих модульних тестів у процес CI для перевірки правильності роботи окремих компонентів додатку. Це допомагає виявити помилки та недоліки на ранніх етапах розробки.

3. **Автоматизовані тести взаємодії на рівні GUI**: Додавання автоматизованих тестів взаємодії на рівні графічного інтерфейсу користувача для перевірки функціональності та коректності роботи додатку з точки зору кінцевого користувача.

4. **Інтеграція з системою керування версіями**: Налаштування CI-системи для автоматичного відслідковування змін у вихідному коді та запуску процесу CI при кожному новому коміті у репозиторій.

Застосування Continuous Integration для Win32 десктопних додатків дозволяє розробникам ефективно впроваджувати новий функціонал та вчасно виявляти та виправляти помилки, забезпечуючи стабільність та якість продукту [10].

Continuous Deployment стратегії для Win32 десктопних додатків. Continuous Deployment (CD) є важливим етапом у процесі розробки десктопних додатків для операційної системи Windows. Він передбачає автоматизоване розгортання нових змін та оновлень у додатку в продуктивному середовищі. Для Win32 десктопних додатків, які можуть мати складну структуру та залежності, важливо мати чітку та ефективну стратегію CD.

Основні стратегії Continuous Deployment для Win32 десктопних додатків включають:

1. **Автоматичне розгортання на тестове середовище**: Після успішного завершення процесу Continuous Integration (CI), збірка додатку автоматично розгортається на тестове середовище. Це дозволяє тестувати новий функціонал та виправлення помилок перед їхнім впровадженням у продуктивне середовище.

2. **Умовне розгортання на продуктивне середовище**: Перед повним розгортанням нових змін у продуктивне середовище можна використовувати стратегії умовного розгортання. Наприклад, можна впроваджувати новий **функціонал** лише для обмеженої кількості користувачів або географічних регіонів, щоб виявити потенційні проблеми перед широким впровадженням.

3. **Автоматичне розгортання в продуктивне середовище**: Після успішного завершення всіх тестів і підтвердження якості збірки, додаток автоматично розгортається

в продуктивне середовище. Це дозволяє **забезпечити** найшвидше розгортання нового функціоналу та виправлень помилок для користувачів.

Застосування стратегій Continuous Deployment дозволяє розробникам зберігати високу швидкість розробки та зменшувати час між розробкою нового функціоналу та його впровадженням для кінцевих користувачів [11].

Автоматизоване тестування Win32 десктопних додатків. Автоматизоване тестування є необхідною складовою процесу розробки десктопних додатків для операційної системи Windows. Для Win32 десктопних додатків, які мають складну логіку та інтерфейс, важливо мати надійні та ефективні автоматизовані тести, які перевіряють різні аспекти додатку.

Основні види автоматизованих тестів для Win32 десктопних додатків включають:

1. **Модульні тести**: Це тести, які перевіряють окремі компоненти програмного забезпечення. У випадку Win32 десктопних додатків, це може бути тестування окремих функцій або методів.
2. **Функціональні тести**: Ці тести перевіряють функціональність додатку з точки зору кінцевого користувача. Вони дозволяють переконатися, що програма працює так, як очікується, і виконує всі необхідні функції.
3. **Тести взаємодії на рівні GUI**: Ці тести перевіряють коректність роботи графічного інтерфейсу користувача. Вони переконуються, що всі елементи і кнопки на формах працюють правильно та реагують на взаємодію користувача.
4. **Тести продуктивності**: Ці тести перевіряють швидкість та продуктивність додатку в різних умовах навантаження. Вони дозволяють виявити можливі проблеми з продуктивністю та оптимізувати роботу додатку.
5. **Тести безпеки**: Ці тести перевіряють захищеність додатку від різних видів атак та вразливостей. Вони допомагають забезпечити безпеку додатку перед випуском в продуктивне середовище.

Застосування автоматизованих тестів для Win32 десктопних додатків допомагає забезпечити якість та стабільність програми, зменшуючи кількість помилок та проблем, які можуть виникнути в процесі розробки [12].

Інфраструктура як код (IaC) для управління середовищами розробки. Інфраструктура як код (Infrastructure as Code, IaC) — це підхід до управління інфраструктурою, який полягає в автоматизованому конфігуруванні та управлінні інфраструктурними ресурсами за допомогою коду. Для розробки Win32 десктопних додатків, IaC може бути корисним для створення та управління тестовими, розробочими та продуктивними середовищами.

Основні переваги використання IaC для управління середовищами розробки включають:

1. **Консистентність середовищ**: За допомогою IaC, середовища розробки можуть бути легко створені та розгорнуті з точністю до деталей. Це дозволяє забезпечити, що кожен член команди працює з однаково налаштованим середовищем, що сприяє консистентності та уникненню конфліктів.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

2. **Швидкість розгортання**: Замість ручного налаштування середовищ для розробки, IaC дозволяє автоматизувати процес створення та розгортання середовищ, що прискорює розгортання нових версій додатку та відлагодження проблем.

3. **Скорочення часу відновлення**: У випадку виникнення проблем або потреби у відновленні середовища, IaC дозволяє легко відновити попередній стан за допомогою автоматизованих процесів.

4. **Відслідковування змін**: Код IaC зберігається в системі керування версіями, що дозволяє відстежувати всі зміни до інфраструктури та відкатувати їх у випадку необхідності.

Застосування IaC для управління середовищами розробки дозволяє забезпечити швидкість, консистентність та надійність інфраструктури, що використовується для розробки Win32 десктопних додатків [13]. Далі буде наведено декілька ключових інструментів і практик, які можуть бути використані для реалізації Інфраструктури як коду при розробці Win32 додатків:

Terraform є потужним інструментом для управління інфраструктурою, що підтримує різноманітні платформи, включаючи хмарні сервіси, такі як AWS, Azure, Google Cloud та локальну інфраструктуру. Він дозволяє створювати інфраструктуру за допомогою декларативного коду. У випадку з Win32 додатками, Terraform може бути використаний для автоматизації створення віртуальних машин, мереж та інших ресурсів, які необхідні для роботи та тестування додатку [10].

Приклад використання Terraform:

- створення та налаштування віртуальних машин Windows для автоматизованого тестування та розгортання Win32 додатків;
- налаштування середовищ з необхідними залежностями та конфігураціями для кожного члена команди.

Ansible — це інструмент для автоматизації конфігурації, який може використовуватися для керування середовищами розробки та тестування. Ansible використовує прості YAML-файли (плейбуки) для опису конфігурацій інфраструктури та може керувати налаштуванням як локальних серверів, так і хмарних середовищ [11].

Приклад використання Ansible:

- автоматизація встановлення необхідних пакетів і залежностей для Win32 додатків на віртуальних машинах або фізичних серверах;
- управління оновленнями операційних систем та налаштуванням середовищ для розробки та тестування.

PowerShell DSC (Desired State Configuration) — це інструмент управління конфігурацією, який дозволяє керувати станом серверів Windows. Він використовує декларативний підхід для забезпечення того, щоб сервери залишалися в бажаному стані. Це особливо корисно для забезпечення стабільності та надійності середовищ для розробки Win32 додатків [12].

Приклад використання PowerShell DSC:

- налаштування та підтримка стабільного середовища для розробки та тестування Win32 додатків;

– автоматизація оновлень програмного забезпечення та налаштувань серверів Windows.

Практики реалізації IaC для Win32 додатків

1. **Стандартизація конфігурацій**: Використання IaC допомагає стандартизувати конфігурації середовищ для всіх етапів розробки, тестування та розгортання. Створення чітких та узгоджених конфігурацій підвищує стабільність та прогнозованість процесів.

2. **Автоматизоване налаштування середовищ**: За допомогою IaC можна автоматизувати не тільки створення інфраструктури, але й її налаштування. Це включає встановлення необхідних пакетів, налаштування мережевих параметрів та конфігурацію системного програмного забезпечення, необхідного для роботи додатка.

3. **Керування версіями інфраструктури**: IaC дозволяє використовувати системи керування версіями, такі як Git, для контролю над змінами в конфігурації інфраструктури. Це дозволяє легко відстежувати зміни, відкотити їх у випадку проблем або автоматично застосовувати нові версії.

4. **Тестування інфраструктури**: Як і код додатків, конфігурації IaC можуть бути протестовані перед їх впровадженням. Це дозволяє виявляти помилки в інфраструктурі до її розгортання, що знижує ризик простоїв та збоїв.

Інструменти та технології для автоматизації розробки Win32 десктопних додатків. Існує багато інструментів та технологій, які можна використовувати для автоматизації розробки Win32 десктопних додатків. Деякі з них спрямовані на автоматизацію конкретних аспектів розробки, таких як збірка чи тестування, тоді як інші надають повний набір інструментів для управління розробкою та розгортанням.

Ось деякі з найпопулярніших інструментів та технологій для автоматизації розробки Win32 десктопних додатків:

1. Visual Studio: Інтегроване середовище розробки (IDE) для мови програмування C#, яке надає широкі можливості для автоматизованої збірки, тестування та розгортання Win32 десктопних додатків.

2. Azure DevOps: Платформа для управління життєвим циклом розробки програмного забезпечення, яка надає інструменти для Continuous Integration, Continuous Deployment та інші автоматизовані процеси.

3. TeamCity: Інструмент Continuous Integration та Continuous Deployment від компанії JetBrains, який підтримує різні мови програмування, включаючи C#.

4. Selenium: Фреймворк для автоматизованого тестування вебдодатків, який може бути використаний для тестування вебінтерфейсу десктопних додатків, що використовують браузерні компоненти.

5. PowerShell: Мова сценаріїв для автоматизації адміністративних завдань у середовищі Windows, яка може бути використана для автоматизації різних аспектів розробки, таких як збірка, тестування та розгортання.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

6. Docker: Платформа контейнеризації, яка дозволяє упаковувати додатки та їхні залежності в контейнери для забезпечення консистентності середовищ розробки та розгортання.

7. Jenkins: Інструмент Continuous Integration та Continuous Deployment, який надає гнучкі можливості конфігурації та інтеграцію з різними іншими інструментами.

Ці інструменти та технології допомагають розробникам забезпечити ефективну та автоматизовану розробку, тестування та розгортання Win32 десктопних додатків [11].

Збереження документів у хмарному сховищі з використанням Minio. Збереження документів є важливою складовою багатьох десктопних додатків, включаючи Win32 додатки. Один з найпоширеніших підходів до збереження даних — використання хмарних сховищ. Minio — це відкрите програмне забезпечення, яке надає об'єктове сховище, сумісне з Amazon S3, і може бути використано для збереження документів у хмарному середовищі.

Основні переваги використання Minio для збереження документів у хмарному сховищі включають:

1. **Масштабованість**: Minio дозволяє легко масштабувати сховище даних в залежності від потреб вашого додатку. Ви можете почати з невеликої конфігурації та збільшувати ресурси по мірі зростання обсягу даних.

2. **Надійність**: Minio надає можливість реплікації даних для забезпечення надійності та доступності. Ви можете налаштувати кілька копій даних для автоматичного відновлення в разі втрати даних.

3. **Безпека**: Minio підтримує шифрування даних в спокої (at-rest) та шифрування даних в русі (in-transit), що дозволяє забезпечити конфіденційність та цілісність даних.

4. **Простота інтеграції**: Minio забезпечує API сумісне з Amazon S3, що робить його легко інтегрованим з багатьма додатками та сервісами, які підтримують роботу з Amazon S3.

Додавання можливостей збереження документів у хмарному сховищі, такому як Minio, до вашого Win32 десктопного додатку дозволить забезпечити ефективне та надійне збереження даних користувачів [13].

Висновки

У роботі розглянуто різноманітні аспекти автоматизації розробки Win32 десктопних додатків та можливість збереження вхідних і вихідних даних у хмарному сховищі з використанням Minio. Ці практики та інструменти дозволяють розробникам ефективно керувати розробкою, тестуванням, розгортанням та збереженням даних додатків.

Першим кроком у напрямку автоматизації розробки Win32 десктопних додатків є використання Continuous Integration (CI) та Continuous Deployment (CD) для автоматизованої збірки, тестування та розгортання змін. Це дозволяє розробникам швидко та надійно впроваджувати новий функціонал та виправлення помилок.

Потім була розглянута важливість автоматизованого тестування для забезпечення якості та стабільності Win32 десктопних додатків. Автоматизовані тести дозволяють виявляти та виправляти помилки на ранніх етапах розробки, зменшуючи час та зусилля, необхідні для виправлення недоліків.

Також розглянуто необхідність використання Інфраструктури як коду (IaC) для управління середовищами розробки та забезпечення їхньої консистентності та ефективності.

Розглянуто можливості збереження документів у хмарному сховищі за допомогою Minio. Цей підхід дозволяє забезпечити безпеку, масштабованість та надійність збереження даних додатків у хмарному середовищі.

В цілому, автоматизація розробки та збереження даних у хмарному сховищі є важливими компонентами сучасної розробки десктопних додатків, що допомагають розробникам забезпечити швидку, надійну та якісну розробку програмного забезпечення.

ЛІТЕРАТУРА / REFERENCES

1. Voggenreiter, M., Angermeir, F., Moyón, F., Schöpp, U., & Bonvin, P. Automated Security Findings Management: A Case Study in Industrial DevOps. *arXiv preprint* , 2024. [Електронний ресурс] — режим доступу до ресурсу: http://arxiv.org/abs/2401.06602v1.

2. Duque Anton, S., Fraunholz, D., Krohmer, D., et al. Creating it from SCRATCh: A Practical Approach for Enhancing the Security of IoT-Systems in a DevOps-enabled Software Development Environment. *arXiv preprint*, 2020. [Електронний ресурс] — режим доступу до ресурсу: http://arxiv.org/abs/2010.14865v1.

3. Alenezi, M., Zarour, M., & Akour, M. Can Artificial Intelligence Transform DevOps? *arXiv preprint* 2022. [Електронний ресурс] — режим доступу до ресурсу: http://arxiv.org/abs/2206.00225v1.

4. Luz, W. P., Pinto, G., & Bonifácio, R. Building a Collaborative Culture: A Grounded Theory of Well Succeeded DevOps Adoption in Practice. *arXiv preprint*, 2018. [Електронний ресурс] — режим доступу до ресурсу: http://arxiv.org/abs/1809.05415v1.

5. Chris Sells "Windows Forms Programming in C# (Microsoft .NET Development Series)" " Addison-Wesley Professional, 2004.

6. Paul M. Duvall, Steve Matyas, Andrew Glover. "Continuous Integration: Improving Software Quality and Reducing Risk." Addison-Wesley Professional, 2007.

7. Jez Humble, David Farley. "Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation." Addison-Wesley Professional, 2010.

8. Mark Fewster, Dorothy Graham. "Software Test Automation." Addison-Wesley, 1999.

9. Kief Morris. "Infrastructure as Code: Managing Servers in the Cloud." O'Reilly Media, 2016.

10. Yevgeniy Brikman. "Terraform: Up & Running: Writing Infrastructure as Code." O'Reilly Media, 2019.

11. Lorin Hochstein, Rene Moser. "Ansible: Up and Running: Automating Configuration Management and Deployment the Easy Way." O'Reilly Media, 2017.

12. Ed Wilson. "Windows PowerShell Desired State Configuration Revealed." Apress, 2014.

13. MinIO Documentation [Електронний ресурс] — режим доступу до ресурсу: https://min.io/docs/minio/linux/index.html

Received 10.01.2025.
Accepted 13.01.2025.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)

Automation of development of Win32 desktop applications: Practical approaches and strategies

The article presents a comprehensive analysis of modern approaches to automating the development of Win32 desktop applications using DevOps practices. Given the complexity and growing requirements of desktop applications in today's software industry, automation becomes an essential part of the development process. The article examines key phases such as Continuous Integration (CI) and Continuous Deployment (CD), which are vital for streamlining the development lifecycle by automating builds, tests, and deployments. These practices help developers reduce the time spent on manual operations and mitigate the risk of human errors.

The Continuous Integration section explores the benefits of setting up automated pipelines that compile and build applications whenever new code is committed. Automated tests are run as part of this process to ensure that changes do not introduce bugs or regressions. Continuous Deployment strategies are discussed in detail, with a focus on conditional deployment techniques that allow for gradual rollouts to production environments, ensuring stability and performance.

Automated testing plays a central role in maintaining the quality and stability of Win32 desktop applications. This paper highlights different types of testing, including unit tests, functional tests, performance tests, and GUI-based tests. The integration of these tests into CI/CD pipelines enables continuous validation of the application throughout its lifecycle, ensuring that any issues are detected early and resolved promptly.

Infrastructure as Code (IaC) is another crucial topic discussed in the article. The concept of IaC allows development teams to automate the creation, configuration, and management of the infrastructure needed for the development and deployment of Win32 applications. The paper provides an overview of popular IaC tools such as Terraform, Ansible, and PowerShell DSC, detailing how they can be used to standardize and automate the provisioning of environments across development, testing, and production stages. This automation contributes to increased consistency, repeatability, and efficiency in managing infrastructure.

In addition to automation in development and deployment, the article addresses the importance of cloud-based storage solutions for Win32 applications. Minio, an open-source, S3-compatible object storage system, is examined as a viable option for storing documents and application data in the cloud. The paper discusses Minio's scalability, reliability, and security features, emphasizing its role in providing efficient, secure, and resilient data storage for desktop applications.

Overall, the article outlines how the adoption of DevOps practices, such as CI/CD, automated testing, IaC, and cloud storage solutions, can significantly improve the development process of Win32 desktop applications. By reducing manual intervention and providing robust automation, these practices help teams increase their productivity, ensure higher software quality, and accelerate time-to-market.

This article is targeted at software developers, DevOps engineers, and technical professionals interested in improving their knowledge of automation strategies and applying them to desktop application development. The provided insights and tools offer practical guidance for

leveraging DevOps methodologies to streamline workflows and enhance the efficiency of Win32 application development.

Keywords: software engineering, development automation, DevOps, Continuous Integration (CI), Continuous Deployment (CD), infrastructure as code, automated testing, Terraform, Ansible, PowerShell DSC, Minio, document storage in the cloud, cloud storage, deployment, Windows, cloud technologies.

Ганжа Андрій Сергійович – аспірант 3-го року навчання Дніпровського національного університету імені Олеся Гончара.
Антоненко Світлана Валентинівна – кандидат технічних наук, доцент Дніпровського національного університету імені Олеся Гончара.

Hanzha Andrii - 3rd year postgraduate student at Oles Honchar Dnipro National University.
Antonenko Svitlana Valentynivna - Candidate of Technical Sciences, Associate Professor at Oles Honchar Dnipro National University.

ISSN 1562-9945 (Print)
ISSN 2707-7977 (Online)