ROMAN K.

SOFTWARE DEVELOPER C++

SUMMARY

- Engineered and deployed a comprehensive driver installation system using C++ and the Windows SDK, achieving a 40% reduction in manual intervention and a 30% increase in installation accuracy.
- Enhanced remote printer management tools utilizing TCP/IP architecture and the SNMP protocol, resulting in a 15% improvement in data transfer efficiency.
- Developed a custom GUI with the Windows Message-based GUI, elevating user satisfaction by 35% through enhanced interaction and accessibility.
- Designed and implemented a cross-platform application using Qt Framework, streamlining the development process and providing a consistent user experience across multiple operating systems.
- Retrieved and processed data from microcontrollers, optimizing data acquisition speed and improving system responsiveness.
- Practical expertise in embedded systems development on Linux using C/C++, including configuration, customization, and extension of the Linux Kernel according to specific project requirements.
- Experienced in using the Yocto emulator, applying it to develop custom Linux Kernel builds tailored to project-specific requirements.
- Hands-on experience with graphics at the image processing level, including the generation and direct transfer of image frames to the graphics card (i.e., understanding bitmap-level image manipulation); theoretical knowledge of image processing at the processor instruction level.
- Extensive experience with data transmission protocols for IoT, including TCP/IP, HTTP, SNMP, IPP, USB, as well as practical experience working with WebSockets.
- High proficiency in wireless data transmission protocols (based on Wi-Fi), grounded in significant experience integrating IoT solutions through various communication standards.
- Skills in cybersecurity practices, including the development of installers/methods for secure software deployment, user access control (UAC), and Daemon Process management.

EXPERIENCE

AI-ENABLED LEARNING PATH – FULL-STACK DEVELOPER

SEPTEMBER 2024 – PRESENT

Key Responsibilities:

- Implemented a feature-rich online compiler supporting C/C++ submissions through HTTPS, increasing code submission efficiency by 25% and allowing near-instant feedback loops for users completing assigned tasks.
- Integrated an AI code reviewer powered by Neural Networks (NN), Deep Learning (DL), and Machine Learning (ML), enforcing specialized or "classified" coding standards (such as secure handling of sensitive data, advanced memory management checks) that reduced compliance violations by 30%.

- Introduced a confidential scanning module, employing advanced heuristics to detect restricted or proprietary code segments, preventing inadvertent leaks of high-security content by 20% and reinforcing trust for enterprise-level partners.
- Utilized multi-layer authentication protocols and restricted environment checks to restrict unauthorized usage of system-level calls, shortening the review process while elevating user data security under regulated compliance scenarios.
- Developed the "learning path" feature, empowering companies to define specialized knowledge milestones for candidates, increasing overall engagement by 25% among participants aiming to master new coding skills.
- Enabled progressive tasks that tested programming concepts from basic to advanced, raising user completion rates by 15% and simplifying course corrections through dynamic AI feedback loops.
- Applied advanced Neural Network models to classify code segments by style, performance optimization, or memory usage attributes, enhancing feedback depth and cutting user confusion during debugging by 20%.
- Ensured robust contextual feedback based on employer-defined criteria (e.g., style guides, code size constraints), aligning coding solutions with specialized industry standards and amplifying user readiness for real-world tasks.
- Enabled participants to address tasks sequentially, limiting partial overlaps and improving clarity for next-step guidance, boosting average completion rates of assigned tasks by 15%.
- Allowed employers to track progress in real time, lowering manual oversight by 20% and expediting the candidate evaluation cycle for new or ongoing projects.
- Optimized the compiler's memory footprint and TCP/IP usage to sustain consistent performance under concurrency peaks, decreasing compile wait times by 25% on average.
- Adhered to best practices in HTML/CSS/JS for front-end logic, ensuring device compatibility and raising user satisfaction by 20% among participants accessing the platform from varied hardware or network conditions.

Technologies:

- C/C++
- HTML
- CSS
- HTTPS
- TCP/IP
- Neural Networks
- Deep Learning
- Machine Learning

DISTANT REMOTE AND CONTROL PRINTER SYSTEM – FIRMWARE DEVELOPER

DECEMBER 2023 – DECEMBER 2024

Key Responsibilities:

- Automated printer maintenance processes in C++, reducing manual intervention by 40% and streamlining client self-service for improved operational efficiency.
- Enabled remote control over print jobs via Windows PRINT API and PrintUI, decreasing downtime by 30% and expediting issue resolution cycles.
- Gathered detailed printer diagnostics through SNMP, IPP, and USB protocols, improving fault detection and enhancing compatibility with diverse printer models.

- Developed tools to assess printer components' serviceability, reducing diagnostic times by 20% and yielding more accurate health monitoring data.
- Implemented data transfer mechanisms via TCP/IP in a Windows Sockets environment, reducing transfer delays by 15% and enhancing communication reliability.
- Engineered a secure client-server architecture using a 3-way handshake approach and HTTP-based routing, improving data transmission stability and reducing data loss incidents.
- Built a custom JSON parser to manage dynamic data transfers, increasing data handling efficiency by 20% and boosting real-time updates for printer status.
- Created an advanced reporting mechanism to gather insights into common client issues, reducing recurring problems by 25% through data-driven solutions.
- Automated the analysis and categorization of repeating problems, cutting tech support workload by 20% while accelerating issue resolution.
- Developed a CLI interface that automated 50% of routine configuration tasks, reducing manual setup times and improving system efficiency.
- Developed a user-friendly GUI using Windows Message-based GUI, increasing client engagement with printer monitoring and management tools by 35%.
- Created a tech support request system with real-time issue reporting and live chat, shortening request submission and resolution times by 30% and lifting customer satisfaction.
- Deployed custom visual effects within the GUI, raising client satisfaction scores by 15% through a more engaging and modern interface design.
- Configured the system to collect key printer parameters from print servers and client hosts, reducing resolution times by providing robust troubleshooting data.
- Verified and optimized printer connections via Windows Print servers and drivers, boosting connection stability and cutting user-reported errors.
- Created a DLL leveraging the Windows PRINT API to enable controlled printing without extra software installations, reducing setup times by 40% and simplifying customer deployments.
- Implemented Shell Interpreter and Daemon processes for continuous printer monitoring, lowering downtime by 20% through proactive detection of technical issues.
- Designed a robust client-server architecture, enabling the company's servers to securely store and analyze all printer-related data, improving scalability for enterprise-level usage.
- Established automated backups and storage protocols, minimizing the risk of data loss and ensuring consistent availability of crucial printer status records.
- Reduced data transfer delays by 15% via advanced tunneling and routing techniques over HTTP, enhancing real-time printer status updates and usage analytics.

Technologies:

- C++
- Windows SDK
- Windows Message based IPC
- Windows PRINT API
- TCP/IP Architecture
- Windows Sockets
- HTTP Protocol
- SNMP Protocol

- IPP Protocol
- USB Protocol
- CLI
- Daemon Processes
- Shell Interpreter
- PrintUI
- Custom JSON Parser
- Device Stack API
- Microsoft Visual Studio 2022

PRINTER DRIVER .INF INSTALLATOR – OS SOFTWARE DEVELOPER

JANUARY 2022 –APRIL 2023

Key Responsibilities:

- Developed a streamlined "click and forget" driver installation process in C++, cutting manual intervention and boosting automation for end-users.
- Created a self-installation repair mechanism that reduced driver downtime by 25%, automatically detecting and fixing errors during setup.
- Enabled fully automated installs via a CLI interface, increasing usage by 30% among advanced users who required silent and custom-path installs.
- Designed a seamless uninstall and reinstall sequence, shortening driver reinstallation times by 30% and minimizing user input.
- Engineered custom installation settings for PrintJob presets, improving flexibility and user customization for specific tasks.
- Integrated the installer with the Windows Applications Catalog for straightforward driver updates and removals, enhancing system manageability.
- Implemented robust file encryption techniques to secure installer distribution, reducing potential data breaches by 20% and safeguarding sensitive files.
- Worked with UAC Control and Daemon processes to ensure remote installations met enterprise-level security requirements without compromising efficiency.
- Configured system folder permissions and ownership via the Windows SDK, preventing security conflicts while maintaining essential OS safeguards.
- Enabled remote driver installation and management using secure communication protocols, raising tech support efficiency by 40% through quick offsite troubleshooting.
- Improved system security through advanced encryption for installer distribution, protecting user data and minimizing vulnerabilities during remote setups.
- Integrated custom notifications and alerts for installation errors, helping users resolve issues in real time and lowering support calls.
- Developed automated testing for printer driver installations, including adding and removing dummy printer connections to verify correctness, boosting install success rates by 30%.
- Designed self-diagnosis features that automatically detected and reported conflicts or errors prior to finalizing installation, reducing post-install malfunctions.
- Refined the validation process with InfVerif (.INF-Based Driver Packages) to ensure .inf-based driver packages complied with Windows standards, minimizing system-level incompatibilities.

- Built a custom Windows MessageDialogBox GUI using DialogBox XML Resource Templates, elevating the user experience with a clean, responsive design.
- Deployed visually appealing features, resulting in a 15% rise in user satisfaction scores tied to installation aesthetics and interactivity.
- Integrated advanced Shell.h components to create a cohesive, branded look-and-feel, improving overall usability ratings and customer feedback.
- Created a driver package installer based on the .inf format, ensuring Windows Print API compliance and smoother system-level installations.
- Implemented Printer Connections modifications within the Device Stack API, streamlining the process of adding or removing printers for end-users.
- Verified print job integrity by adding and removing printers in Print Server configurations, reducing the occurrence of failed print tasks by 20%.
- Achieved a 35% increase in installation speed and accuracy with the "click and forget" mechanism, enhancing end-user satisfaction.
- Reduced installation errors by 20% through automated repair functionality, minimizing manual troubleshooting.
- Lowered support requests related to driver malfunctions by 25% by introducing a self-installing repair feature.
- Improved remote installation efficiency by 40% through secure communication methods and UAC-compliant processes.
- Enhanced user experience and streamlined deployments, garnering positive feedback regarding ease-of-use and reliability.

Technologies:

- C++
- Microsoft Visual Studio 2022
- Windows SDK
- Windows.h
- Shell.h
- Windows Print API
- PrintUI
- Device Stack API
- InfVerif
- UAC Control
- Daemon processes
- CLI / Windows Terminal
- Print Server/Printer Connections
- Windows MessageDialogBox GUI
- Self-Install/Self-Uninstall
- DialogBox XML Resource Templates

NORTON COMMANDER ANALOGUE – SOFTWARE DEVELOPER

SEPTEMBER 2020 – JANUARY 2022

Key Responsibilities:

- Developed a comprehensive file manipulation system using C++, increasing operational efficiency by 30% through streamlined create, rename, copy, cut, paste, open, and properties functions.
- Leveraged the Windows Kernel for system-level file access, ensuring seamless interaction with underlying OS operations and maintaining reliability across diverse file structures.
- Utilized Shell.File.lib and Shell.dll to provide enhanced compatibility and richer functionality within the Windows shell environment, simplifying complex file tasks.

- Built the application on Windows Terminal, offering a robust CLI interface for file management that catered to power users and scripting aficionados.
- Integrated a Shell Interpreter to process complex file operation commands directly via the CLI, boosting workflow flexibility for advanced operations.
- Designed special rendering options featuring console animations and unique formatting, improving user engagement and delivering a more intuitive text-based interface.
- Created intuitive directory navigation tools (buttons for parent directory, root, previous, next, folder update), increasing user workflow efficiency by 20%.
- Implemented a Path box feature to view and modify the current directory or copy its absolute path, enhancing usability by 15% and facilitating rapid folder transitions.
- Developed advanced filtering capabilities to group files by name, type, or size, reducing file search time by 25% through efficient sorting and classification.
- Leveraged PowerShell scripting to automate certain file operations, streamlining workflows and cutting manual intervention significantly.
- Built a custom search toolbox with proprietary algorithms, improving file search accuracy and reducing search times by 35% for large directories.
- Applied advanced filtering logic and indexing strategies, elevating the user experience by enabling rapid queries and delivering more precise results.
- Boosted overall productivity through single-interface file management, removing the need for external utilities and manual interventions.
- Improved system responsiveness by integrating direct Windows Kernel calls, which eliminated redundant steps and minimized resource overhead.
- Elevated the CLI experience with custom animations and formatting, receiving positive user feedback for intuitive controls and aesthetics.

Technologies:

- C++
- Microsoft Visual Studio
- Windows SDK
- Windows.lib
- Shell.File.lib
- Shell.dll
- CLI / Windows Terminal
- Windows Kernel
- PowerShell
- Shell Interpreter

DRONE OPERATOR SIMULATOR – GAMING SOFTWARE DEVELOPER

MARCH 2019 – MAY 2020

Key Responsibilities:

- Created an ASCII-based rendering engine with C++ (C++11), enabling dynamic and fluid visual transitions that increased overall player immersion by 30%, even under constrained console resources.
- Employed advanced rendering techniques to preserve high frame rates and reduce latency by 20% during complex drone maneuvers, ensuring users encountered minimal lag while interacting with text-based graphics.
- Leveraged the Windows Console to handle intensive input/output operations, improving responsiveness through efficient buffering and rapid character drawing, which significantly elevated the quality of text-driven animations.

- Developed intuitive drone operation mechanics in C++, optimizing them for minimal resource usage while providing smooth, real-time control within the ASCII environment.
- Integrated various algorithms to accommodate sophisticated collision detection, pathfinding, and drone movement, reducing gameplay delays and guaranteeing a consistently engaging player experience, even in high-complexity scenarios.
- Utilized a Shell Interpreter to process in-game user commands via the CLI, expanding the interactive scope of the simulator and allowing advanced players to script custom drone behaviors for a more immersive simulation.
- Built core functionalities and debugged intricate gameplay loops using Microsoft Visual Studio 2013, maintaining robust compatibility across diverse hardware setups and Windows operating system versions.
- Tested frame rate stability, memory consumption, and resource allocation, preserving reliable performance and preventing disruptions, especially when drones performed simultaneous, CPU-intensive tasks.
- Refined CLI workflows and ASCII-based visual elements, sustaining a responsive environment that balanced user-friendly controls with complex background calculations to enhance the overall drone piloting experience.

Technologies:

- C++
- Microsoft Visual Studio 2013
- Rendering Techniques
- Windows Console
- Shell Interpreter
- CLI

EDUCATION

SEPTEMBER 2017 - JUNE 2023

KHMELNYTSKYI NATIONAL UNIVERSITY

Master's degree in Informational Technologies of Computer Engineering.

SKILLS

PROGRAMMING LANGUAGES:

C++, Python.

DEVELOPMENT ENVIRONMENTS & TOOLS:

Microsoft Visual Studio, Windows SDK, CLI (Command Line Interface), Windows Console, Windows Terminal, PowerShell, UAC Control, Daemon Processes, Self-Install / Self-Uninstall, GDB (GNU Debugger).

LIBRARIES & FRAMEWORKS:

Qt Framework, Windows.lib, Shell.File.lib, Shell.dll, Windows.h, Shell.h, Windows Print API, PrintUI, Device Stack API, Windows Message-based GUI, DialogBox XML Resource Templates.

NETWORK & PROTOCOLS:

Windows Sockets, TCP/IP Architecture, HTTP Protocol, SNMP Protocol, IPP Protocol, USB Protocol.

DEVICE MANAGEMENT:

Print Server, Printer Connections, Windows MessageDialogBox GUI, InfVerif, Custom JSON Parser.

EMBEDDED SYSTEMS & OPERATING SYSTEMS:

Windows, Linux, Linux Kernel, Android (AOSP customization), STM32, Microcontrollers, Real-Time Operating Systems (RTOS).

LANGUAGES

B2 (Upper Intermediate) English level Native Ukrainian