

A survey of Enterprise Network Boot and Infrastructure Automation

¹Dr.Amarsinh Varpe, Asso. Prof., Department of Computer Science and Engineering,
Sandip University, Nashik, India, amarsinh.varpe@sandipuniversity.edu.in

²Bhakhar Jay Kantilal, Student, Department of Computer Science and Engineering,
Sandip University, Nashik, India, jaybhakhar.cse@sandipuniversity.edu.in

³Belsare Piyush Milind, Student, Department of Computer Science and Engineering,
Sandip University, Nashik, India, piyushbelsare.cse@sandipuniversity.edu.in

⁴Akati Omkar Reddy, Student, Department of Computer Science and Engineering,
Sandip University, Nashik, India, akatiomkarreddy.cse@sandipuniversity.edu.in

⁵Koshti Rohan Ashok, Student, Department of Computer Science and Engineering,
Sandip University, Nashik, India, koshtirohan4@gmail.com

Abstract: Deploying Windows-based workstations in enterprise settings is often tedious and time-consuming, even with advances like PXE, WDS, and MDT for centralized installation. Post-installation tasks—such as driver and application installations—typically require manual intervention, leading to longer deployment times and inconsistencies. This survey reviews various technologies related to enterprise network booting, deployment frameworks, package management systems, and automation methods, examining tools like PXE, iPXE, iVentoy, Windows ADK, PowerShell, Winget, and Chocolatey. The study highlights their strengths, weaknesses, current trends, challenges, and research gaps in workstation deployment.

To address the identified gaps, the paper introduces NetForge: Enterprise Network Boot and Infrastructure Automation, a lightweight framework enhancing network-based Windows deployment with customizable post-installation automation. NetForge integrates PXE deployment with a user-friendly, menu-driven provisioning utility that allows selective installation of applications and drivers from various sources, including Winget and Chocolatey. This framework moves away from a fixed software image to a flexible solution, maintaining centralized automation. Its goal is to reduce deployment times, ensure consistency, streamline workstation preparation, and boost deployment efficiency in educational, corporate, and training environments.

Keywords — PXE, iVentoy, Network Boot, PowerShell, Winget, Chocolatey, Post - Installation Automation and Software Provisioning.

I. INTRODUCTION

The growing dependence on Windows-based computing infrastructure has increased the demand for efficient and scalable workstation deployment in educational institutions, corporate organizations, government agencies, and training centers. These environments frequently require large numbers of computers to be deployed with identical operating systems, applications, drivers and configuration locations. Traditional deployment methods based on bootable USB drives or optical media require administrators to install and configure each workstation individually, making the process inefficient, time-consuming and difficult to scale. To overcome these limitations, enterprise environments have adopted network-

based deployment knowledge such as the Preboot Execution Environment (PXE), Windows Deployment Services (WDS), Microsoft Deployment Toolkit (MDT), and Windows Assessment and Deployment Kit (Windows ADK). These methods enable centralized operating system deployment through network boot, unattended installation, image management, and deployment automation, significantly reducing manual effort while improving deployment consistency.

More recently, solutions such as iVentoy have simplified PXE-based deployment by providing flexible multi image network boot environments with minimal configuration requirements. Despite these advancements, operating system deployment represents only one phase of

workstation preparation. After Windows installation, administrators must typically install browsers, productivity software, development tools, runtime libraries, hardware drivers, and organization specific applications. These post-installation activities are frequently performed using separate scripts or independent deployment tools, resulting in fragmented workflows, longer deployment times, and inconsistent software configurations across deployed systems. Recent advances in automation technologies, including PowerShell, Windows Package Manager (Winget), and Chocolatey, have enabled organizations to automate software installation and system configuration more effectively. However, these technologies are often deployed independently rather than as part of a unified workstation deployment framework. Consequently, administrators continue to integrate multiple tools manually to complete the deployment lifecycle.

This survey examines contemporary research and enterprise technologies related to Windows deployment, network boot mechanisms, software provisioning, and infrastructure automation. It analyzes their capabilities, identifies existing limitations, and highlights the need for a lightweight framework that combines operating system deployment with centralized post-installation automation. Based on this analysis, the paper presents Net Forge, a conceptual deployment framework designed to integrate established technologies into a unified, scalable, and automated workstation deployment workflow.

1.1 Problem Statement

Current enterprise deployment technologies automate operating system installation but fall short in providing centralized support for post-installation software provisioning. This limitation leads administrators to use various tools or manual methods, particularly in settings like educational laboratories and enterprise offices where uniform software environments are vital.

As systems scale, the need for manual installation or executing different scripts heightens the chances of human error and results in configuration drift among machines, which were initially intended to be identical.

1.1.1 Increased Deployment Time: After the operating system is installed, administrators must manually install applications, runtime libraries, drivers, and other software components. These repetitive tasks significantly increase the overall deployment time.

1.1.2. Inconsistent Workstation Configurations: Manual software installation and configuration often result in differences between systems. Such inconsistencies can affect software compatibility, maintenance, and user experience across deployed workstations.

1.1.3. High Administrative Effort: Managing application installation, driver deployment, and system configuration

through separate tools or scripts increases the workload of IT administrators, particularly in environments with frequent deployments.

1.1.4. Poor Scalability and Maintainability: As the number of deployed systems grows, maintaining standardized software configurations becomes increasingly difficult. Independent deployment methods can also lead to configuration drift over time.

1.1.5. Complex Multi-System Deployment: Deploying multiple computers simultaneously requires coordinating various deployment utilities, package managers, configuration scripts. This fragmented workflow increases deployment complexity and the likelihood of errors.

1.2 Motivation & Objectives

Modern automation technologies like PowerShell, Winget, and Chocolatey have changed enterprise software deployment by allowing script-based application provisioning.

However, they are often used separately from operating system deployment, necessitating manual coordination by administrators. This highlights the need for a lightweight framework that combines network boot, operating system deployment, software provisioning, and configuration automation into a single, efficient workflow for scalable workstation preparation.

The primary objectives of this survey are:

1. To review recent research on enterprise network boot and Windows deployment technologies.
2. To analyze existing approaches for automated software provisioning and post-installation configuration.
3. To compare current deployment frameworks, package management systems, and automation tools.
4. To identify research gaps in integrated workstation deployment workflows.
5. To propose “NetForge” as a framework that combines centralized Windows deployment with customizable post - installation software provisioning.

II. METHODOLOGY

This study follows a structured survey and comparative analysis methodology to investigate enterprise network boot, Windows operating system deployment, software provisioning, and infrastructure automation. The methodology was designed to identify existing technologies, compare their capabilities and limitations, determine research gaps, and formulate the conceptual NetForge framework.

Literature and Technology Identification: Relevant research studies, technical documentation, and established deployment technologies were identified based on their

relationship to enterprise workstation deployment and infrastructure automation. The reviewed areas include PXE, DHCP, TFTP, Windows Deployment Services (WDS), Windows Assessment and Deployment Kit (Windows ADK), Windows Preinstallation Environment (WinPE), Microsoft Deployment Toolkit (MDT), PowerShell, Winget, Chocolatey, Microsoft Configuration Manager (MECM), and Microsoft Intune.

Thematic Classification: The identified technologies and approaches were classified according to their primary deployment functions. The major categories considered in the study are network boot, operating system deployment, software provisioning, driver and runtime installation, system configuration, and centralized management. This classification provides a structured basis for examining the role of individual technologies within the overall deployment workflow.

Comparative Analysis: The selected technologies and deployment approaches were compared according to their functional capabilities, deployment scope, automation support, administrative requirements, and integration potential. Comparative tables were used to organize the identified characteristics and examine how different technologies address specific stages of enterprise workstation deployment.

Research Gap Identification: The analysis was used to identify limitations and gaps in existing deployment approaches. Particular attention was given to the fragmentation between operating system deployment and post-installation activities such as software provisioning, driver installation, runtime configuration, and system customization. The study also examined the complexity involved in coordinating multiple heterogeneous deployment technologies.

Alternative Approaches: Different deployment approaches were considered to provide a broader basis for comparison. These include manual installation, USB-based deployment, PXE-based deployment, MDT task-sequence approaches, configuration-management systems, and hybrid deployment models. Their characteristics were considered in relation to automation, scalability, consistency, and administrative requirements.

Framework Formulation: Based on the findings from the literature review, thematic classification, comparative analysis, and research-gap identification, NetForge was formulated as a conceptual modular framework. The proposed framework integrates network boot, operating system deployment, software provisioning, driver and runtime installation, and system configuration within a structured workflow using technologies such as PXE/iVentoy, Windows ADK/WinPE, PowerShell, Winget, Chocolatey, and local software repositories.

As this study is survey-based and presents a conceptual

framework, experimental performance measurements are not claimed. Quantitative benchmarking and controlled validation of the proposed NetForge framework are considered part of future implementation and evaluation.

III. BACKGROUND

Network Boot Technologies: Network boot technologies allow computers to start and install operating systems directly from a deployment server. PXE, DHCP, and TFTP work together to provide network-based booting, while WDS and Windows ADK extend these capabilities for enterprise Windows deployment.

Microsoft Deployment Framework: Microsoft deployment frameworks provide tools for operating system imaging, unattended installation, driver integration, and deployment customization. Solutions such as WDS (Windows Deployment Services), MDT, Windows ADK (Assessment and Deployment Kit), and Windows PE help organizations deploy Windows systems in a consistent and automated manner.

Enterprise Windows Deployment: Enterprise Windows deployment enables organizations to efficiently install and configure multiple workstations using centralized deployment servers, rather than relying on manual installation methods. Technologies such as Windows Deployment Scenarios, Windows PE, and Microsoft Deployment Toolkit (MDT) support automated operating system deployment and standardized system configuration.

Automation and Scripting: Automation technologies simplify deployment by reducing manual configuration tasks. PowerShell, Batch scripts, and modern deployment practices enable automated software installation, system configuration, and deployment management across enterprise environments.

Software Provisioning: Software provisioning ensures that deployed systems are equipped with the required applications and runtime components. Package managers such as Winget and Chocolatey, along with local software repositories; enable consistent and automated application deployment in enterprise environments.

IV. LITERATURE REVIEW

Enterprise information technology (IT) infrastructures have experienced rapid growth in complexity due to the increasing number of computing devices, frequent software updates, and the demand for standardized workstation configurations. Organizations ranging from educational institutions and research laboratories to large enterprises must deploy and maintain numerous computer systems while ensuring consistency, reliability, and security. Traditional deployment methods, which rely heavily on manual installation and configuration, are time-consuming, error-prone, and difficult to scale.

Consequently, researchers and industry practitioners have proposed various technologies to automate different stages of the deployment lifecycle. The existing literature demonstrates that enterprise deployment automation has evolved through several technological phases. Early research primarily focused on network-based operating system installation using technologies such as the Preboot Execution Environment (PXE), Dynamic Host Configuration Protocol (DHCP), and Trivial File Transfer Protocol (TFTP).

These technologies enabled computers to boot over a network and retrieve operating system images from centralized servers, eliminating the need for physical installation media. Later developments introduced deployment frameworks such as Windows Deployment Services (WDS), Microsoft Deployment Toolkit (MDT), and the Windows Assessment and Deployment Kit (Windows ADK), which automated operating system imaging, unattended installation, and deployment task sequencing.

More recently, automation has expanded beyond operating system deployment to include software provisioning, configuration management, scripting, and infrastructure automation through tools such as PowerShell, Winget, Chocolatey, and Infrastructure as Code (IaC) frameworks. Although these technologies have substantially improved enterprise deployment efficiency, the literature indicates that most existing solutions address only specific stages of the deployment lifecycle. Network boot technologies focus on operating system initialization, deployment frameworks automate image installation, and scripting tools simplify software configuration.

However, these components are frequently deployed as independent solutions, requiring administrators to integrate multiple technologies to achieve complete workstation provisioning. This observation motivates a comprehensive review of existing deployment technologies and their respective strengths and limitations. This section reviews the major technologies and research contributions related to enterprise deployment automation, beginning with network boot technologies and Windows deployment frameworks before examining scripting, package management, and enterprise configuration systems in subsequent sections.

4.1 Network Boot Technologies & Windows Deployment Frameworks

Several studies have emphasized that centralized deployment frameworks improve deployment consistency and scalability across enterprise environments. Wurster et al. highlighted the importance of standardized deployment automation models, while Arcangeli et al. discussed deployment architectures for distributed systems. Fitzgerald and Stol further demonstrated how continuous deployment practices enhance software delivery and infrastructure

management.

The principal advantage of PXE lies in its ability to support large-scale operating system deployment while reducing administrative effort. Because boot images are maintained centrally, updates can be distributed efficiently without modifying individual workstations. Furthermore, PXE minimizes the need for physical installation media, making it particularly suitable for enterprise environments, educational laboratories, and data centres. Despite these advantages, the literature identifies several limitations. PXE itself provides only a network boot mechanism and does not manage software installation, driver deployment, configuration automation, or post-installation customization.

These activities require additional deployment frameworks and automation tools operating after the operating system has been installed. Network boot technologies have transformed enterprise operating system deployment by enabling client systems to boot directly from deployment servers without requiring removable installation media. The Preboot Execution Environment (PXE), together with DHCP and TFTP, forms the foundation of network-based deployment by allowing clients to obtain network configuration and download boot files automatically. To improve deployment efficiency, Microsoft introduced enterprise deployment solutions such as Windows Deployment Services (WDS), Windows Assessment and Deployment Kit (Windows ADK), Windows Preinstallation Environment (Windows PE), and the Microsoft Deployment Toolkit (MDT). These technologies support unattended installation, image management, driver integration, and deployment customization, significantly reducing manual intervention during Windows deployment.

Although these technologies effectively automate operating system installation, they primarily focus on image deployment and do not fully address software provisioning, application installation, or post-deployment workstation configuration. Consequently, organizations often combine multiple deployment tools to complete the workstation preparation process.

4.2 PowerShell Automation, Winget, Chocolatey & Software Provisioning

Enterprise deployment extends beyond operating system installation to include application deployment, runtime installation, driver integration, and system configuration.

Modern automation technologies such as PowerShell enable administrators to automate repetitive deployment tasks through scripting, improving deployment consistency and reducing administrative effort. Software provisioning has been further simplified through package management systems such as Windows Package Manager (Winget) and Chocolatey, which automate software installation,

dependency management, and application updates from centralized repositories. These tools provide a flexible and scalable approach for deploying enterprise applications across multiple workstations. Research on Continuous Integration, Continuous Delivery, and Infrastructure as Code (IaC) demonstrates that automation significantly improves deployment reliability and reduces manual configuration errors. Studies by Shahin et al., Humble and Farley, Morris, Rahman et al., and Leite et al. emphasize the importance of deployment pipelines, infrastructure automation, and DevOps practices in modern enterprise environments. Despite these advancements, organizations frequently rely on separate scripting frameworks, package managers, and local software repositories to complete workstation preparation. This fragmented approach increases deployment complexity and motivates the need for integrated deployment frameworks.

4.3 Enterprise Deployment Frameworks, Analysis, Literature Summary & Research Trends

Recent research has focused on developing enterprise deployment frameworks that integrate automation, configuration management, and endpoint provisioning into unified deployment solutions. Technologies such as Microsoft Configuration Manager (MECM), Microsoft Intune, CFEngine, Windows Autopilot, and Provisioning Packages support centralized device management, policy enforcement, automated provisioning, and configuration compliance across enterprise environments. The reviewed literature consistently demonstrates that automation improves deployment speed, scalability, and configuration consistency. However, existing solutions often specialize in individual aspects of deployment, such as operating system imaging, endpoint management, software provisioning, or infrastructure automation, rather than providing a complete end-to-end deployment workflow. Overall, current research trends indicate increasing adoption of DevOps, Infrastructure as Code, cloud-based endpoint management, and policy-driven automation. Nevertheless, limited attention has been given to lightweight frameworks that integrate network boot, operating system deployment, software provisioning, driver installation, runtime configuration, and system customization into a single workflow. This research gap forms the motivation for the proposed NetForge framework, which aims to provide a modular and unified enterprise workstation deployment solution.

V. COMPARATIVE ANALYSIS AND DISCUSSION

The technologies reviewed in the previous section address different stages of enterprise workstation deployment. While each technology contributes to improving deployment efficiency, no single solution provides a complete workflow that covers network boot, operating system installation, software provisioning, driver deployment, system configuration, and deployment

monitoring within a unified framework.

5.1 Study of Enterprise Deployment Solutions

Table 1 compares the major technologies and research studies reviewed in this survey based on their primary functionality, strengths, limitations, and relevance to enterprise workstation deployment.

Table 1. Comparative Analysis of Existing Enterprise Deployment Technologies and Research Studies

Technology / Study	Primary Focus	Strength	Limitation
Essential Deployment Metamodel	Deployment automation	Standardized deployment model	No implementation framework
Automatic Deployment	Distributed software deployment	Flexible deployment architecture	Limited enterprise provisioning
Continuous Software Engineering	Continuous deployment	Improves software delivery	Focuses on software lifecycle
PXE	Network boot	Centralized OS boot	No software provisioning
DHCP	Network configuration	Automatic IP assignment	Supports only boot initialization
TFTP	Boot file transfer	Lightweight file transfer	Limited security
WDS	Windows deployment	Centralized OS deployment	Windows-only
Windows ADK	Deployment toolkit	Image creation and customization	Requires additional automation
Winget	Package management	Simple software installation	Limited offline support
PowerShell	Automation	Powerful scripting	Script maintenance required
CI/CD Review	Deployment automation	Deployment pipeline	Software-focused
Continuous Delivery Architecture	Deployment architecture	Scalable deployment	Limited workstation focus
Windows Deployment Scenarios	Deployment guidance	Best practices	Documentation only
Windows PE	Preinstallation environment	Lightweight deployment OS	Used only during deployment
MDT	Automated deployment	Task sequence automation	Windows ecosystem only
MECM	Enterprise management	Complete endpoint management	High infrastructure cost
Intune	Cloud management	Modern endpoint management	Internet dependency
Continuous Delivery	Release automation	Reliable deployment	General software deployment
Windows Autopilot	Device provisioning	Modern cloud deployment	Windows ecosystem only
Provisioning Packages	Device configuration	Post-installation customization	Limited deployment scope
Infrastructure Automation	Configuration automation	Automated remediation	Documentation-based

The table 1 summarizes the primary focus, strengths, and

limitations of the reviewed technologies and studies. The findings show that existing approaches address different aspects of deployment, including network boot, OS deployment, software provisioning, configuration, and enterprise management. However, their limitations indicate that these capabilities are often distributed across separate tools, supporting the need for an integrated approach such as NetForge.

5.2 Comparative Discussion

The reviewed studies demonstrate that enterprise deployment has evolved through advancements in network boot technologies, deployment frameworks, scripting platforms, and infrastructure automation.

Technologies such as PXE, WDS, and Windows ADK primarily support operating system deployment, while PowerShell, Winget, and Chocolatey simplify post-installation software provisioning. Enterprise platforms such as MECM, Microsoft Intune, and Windows Autopilot provide centralized device management and cloud-based provisioning. However, these capabilities are distributed across multiple tools, with no single lightweight solution integrating network boot, OS deployment, software provisioning, driver installation, runtime configuration, and workstation customization. This increases deployment complexity and administrative effort, highlighting the need for an integrated and modular framework such as NetForge.

5.3 Comparative Analysis of Deployment Approaches

Table 2. Comparison of Deployment Approaches

Deployment Approach	Scalability & Automation Level	Administrative Effort	Typical Use Case
Manual Installation	Low	Very High	Small organizations
USB-Based Imaging	Moderate	High	Laboratory deployment
PXE-Based Deployment	High	Moderate	Enterprise OS deployment
Task Sequence Deployment (MDT)	High	Moderate	Automated Windows deployment
Configuration	Very High	Low after	Large

Management Platforms		deployment	enterprise environments
Hybrid Deployment Workflow	High	Low	Flexible workstation provisioning

Enterprise deployment solutions may be classified according to the level of automation they provide. Table 2 compares these deployment approaches based on scalability, automation level, administrative effort, and typical use case. Modern enterprises increasingly prefer network-based deployment because it reduces installation time, minimizes human error, and improves deployment consistency.

5.4 Evaluation of Existing Enterprise Deployment Frameworks

A comparison of existing deployment frameworks reveals that most solutions are designed to address a specific phase of the workstation provisioning lifecycle rather than delivering a fully integrated end-to-end deployment environment. In practice, individual technologies tend to focus on isolated responsibilities such as network-based bootstrapping, operating system image deployment, driver injection, software installation, or post-deployment configuration, without extending their functionality across the entire preparation pipeline. As a result, administrators are often required to combine multiple independent tools and manually coordinate their interactions in order to achieve complete workstation readiness, which introduces additional complexity and increases the potential for configuration inconsistencies across enterprise deployments.

The comparison table 3 shows that different technologies provide different deployment capabilities. PXE, WDS, and MDT mainly support network boot and OS deployment, while PowerShell, Winget, and Chocolatey support software installation and configuration automation. SCCM provides broader centralized deployment and management capabilities. The results highlight the need to coordinate multiple technologies, which supports the proposed NetForge framework.

Table 3. Functional Comparison of Existing Frameworks

Feature	PXE	WDS	MDT	PowerShell	Winget	Chocolatey	SCCM
Network Boot	✓	✓	✓	✗	✗	✗	✓
OS Installation	✗	✓	✓	✗	✗	✗	✓
Driver Deployment	✗	Limited	✓	✓	✗	✗	✓
Application Installation	✗	✗	Limited	✓	✓	✓	✓
Runtime Installation	✗	✗	Limited	✓	✓	✓	✓
Configuration Automation	✗	✗	Limited	✓	Limited	Limited	✓

Remote Administration	X	X	X	Limited	X	X	✓
Centralized Management	X	Limited	Limited	X	X	X	✓

VI. RESEARCH GAP AND PROPOSED FRAMEWORK (NETFORGE)

Research Gap: The literature survey demonstrates that significant advancements have been made in enterprise workstation deployment through technologies such as Preboot Execution Environment (PXE), Windows Deployment Services (WDS), Microsoft Deployment Toolkit (MDT), Windows Assessment and Deployment Kit (Windows ADK), PowerShell automation, and package management tools including Winget and Chocolatey.

These technologies have transformed Windows deployment by enabling centralized operating system installation, unattended deployment, network booting, image management, and automated software installation. Collectively, they have reduced deployment time, improved deployment consistency, minimized manual intervention, and enhanced the scalability of enterprise deployment infrastructures.

Despite these advancements, the reviewed literature indicates that current deployment solutions primarily address individual phases of the deployment lifecycle rather than providing a unified end-to-end workstation provisioning framework. Most research focuses on operating system imaging, deployment automation, Infrastructure as Code, DevOps practices, or software package management as separate research domains.

The first major research gap lies in post-installation software provisioning. Existing deployment frameworks effectively automate operating system installation but provide limited support for integrating application installation, runtime libraries, device drivers, and organization-specific software into the same deployment workflow. In practice, these tasks are commonly performed through separate PowerShell scripts, package managers, deployment utilities, or manual administrative procedures after the operating system has been installed. This fragmented approach increases deployment time, introduces configuration inconsistencies, and reduces overall deployment efficiency, particularly in environments where large numbers of workstations require identical software configurations.

The second research gap concerns integrating heterogeneous deployment technologies. Enterprise deployment often uses separate tools such as PXE, WDS/MDT, Windows ADK, PowerShell, Winget, Chocolatey, and local repositories. Since these tools operate independently, administrators must configure and maintain multiple components. This increases administrative complexity, deployment overhead, troubleshooting effort,

and long-term maintenance requirements. An integrated framework is therefore needed to coordinate these technologies through a structured and manageable deployment workflow.

The third limitation identified in the literature is the complexity of enterprise endpoint management platforms. Comprehensive deployment and endpoint management solutions provide extensive capabilities, including operating system deployment, software distribution, inventory management, and security administration.

However, these platforms often require dedicated server infrastructure, licensing costs, specialized administrative expertise, and considerable implementation effort. Such comprehensive solutions may exceed the operational requirements of educational institutions, computer laboratories, training centers, startups, and small or medium-sized organizations that require an efficient, lightweight, and cost-effective deployment framework without unnecessary infrastructure complexity.

Another important observation is that existing research places greater emphasis on operating system deployment than on flexible workstation provisioning. Most deployment solutions assume that software packages, drivers, and configuration tasks are predefined within deployment images or task sequences. Comparatively fewer studies investigate deployment frameworks that allow administrators or end users to selectively install applications, runtime libraries, development tools, device drivers, and other software components according to organizational or user-specific requirements while maintaining centralized deployment management.

This limitation reduces deployment flexibility and often results in either oversized operating system images or additional post-deployment administrative effort. Furthermore, relatively limited attention has been given to lightweight deployment frameworks that combine network-based operating system deployment with customizable post-installation automation using modern package management systems and scripting technologies.

The growing availability of tools such as PowerShell, Winget, Chocolatey, and local software repositories creates an opportunity to design deployment solutions that automate both operating system installation and software provisioning without relying on complex enterprise management infrastructures. Such an approach can simplify deployment workflows while maintaining flexibility, scalability, and ease of administration. Based on these observations, this survey identifies a clear research opportunity for the development of a modular and

centralized deployment framework that integrates network boot technologies, automated operating system deployment, and customizable post-installation software provisioning into a single workflow. Addressing this gap has the potential to reduce deployment time, improve workstation consistency, simplify administrative tasks, minimize configuration drift, and provide a practical deployment solution for enterprise organizations, educational institutions, training laboratories, and other environments that require efficient and repeatable workstation deployment.

6.1 Proposed Research Work

To address the identified research gap, this survey proposes NetForge: Enterprise Network Boot and Infrastructure Automation as a conceptual deployment framework for enterprise workstation provisioning. Unlike existing solutions that focus on individual deployment stages, NetForge integrates network boot, operating system deployment, software provisioning, driver installation, runtime configuration, and system automation into a unified workflow. The framework utilizes established technologies, including PXE, iVentoy, Windows ADK, PowerShell, Winget, Chocolatey, and local software repositories, to automate workstation preparation while maintaining flexibility and scalability. The proposed framework aims to reduce deployment time, minimize manual intervention, improve deployment consistency, and simplify enterprise workstation management without replacing existing deployment technologies.

6.2 Proposed Framework: NetForge

Enterprise Network Boot and Infrastructure Automation, a modular framework for centralized enterprise workstation deployment. This framework integrates existing technologies into a unified workflow that automates not only operating system deployment but also post-installation software provisioning, while remaining adaptable to various organizational needs. Unlike traditional solutions focusing solely on OS installation, NetForge enhances the deployment lifecycle by automating software provisioning, driver installation, runtime deployment, and workstation configuration within a singular process. Rather than replacing established technologies, it orchestrates standard deployment tools to deliver an efficient and scalable solution for workstation deployment.

The proposed framework is built upon the following core technologies:

PXE (Preboot Execution Environment): Enables client systems to boot directly from the network without requiring bootable USB drives or optical media.

iVentoy: Provides a flexible PXE-based network boot environment capable of serving multiple operating system images through a centralized boot menu.

Windows ADK and Windows PE: Support deployment image preparation, unattended Windows installation, and deployment customization.

PowerShell and Batch Scripts: Automate deployment tasks, software installation, system configuration, validation, and post-installation operations.

Windows Package Manager (Winget): Installs and manages Microsoft-supported and community-maintained software packages.

Chocolatey: Extends software provisioning by providing access to additional package repositories and automation capabilities.

By integrating these technologies into a unified framework, NetForge aims to reduce deployment time, minimize manual administrative effort, improve workstation consistency, simplify deployment management, and provide a scalable solution for enterprise organizations, educational institutions, laboratories, and training environments.

6.3 Proposed Deployment Workflow

The proposed deployment workflow consists of a sequence of automated stages that transform a newly booted client system into a fully configured workstation.

Stage 1: Network Boot: The client computer initiates the deployment process by booting through PXE. The iVentoy server responds to the PXE request and presents the available operating system deployment options without requiring local installation media.

Stage 2: Operating System Deployment: The selected Windows operating system image is deployed using Windows ADK and Windows PE with unattended installation settings, enabling automated disk partitioning, operating system installation, and initial system configuration.

Stage 3: System Initialization: Following operating system installation, PowerShell and Batch scripts initialize the deployment environment, verify successful installation, prepare required directories, and configure the system for software provisioning.

Stage 4: Software Selection: The deployment utility presents administrators or users with a menu-driven interface from which they can select one or more required open-source applications based on organizational or deployment requirements. This approach provides greater flexibility than deploying a fixed software image.

Stage 5: Automated Software Provisioning: Selected applications are automatically installed using the most appropriate software source. Depending on package availability, installations are performed through Windows Package Manager (Winget), Chocolatey, or locally

maintained installer repositories, ensuring support for both online and offline deployment scenarios.

Stage 6: Driver and Runtime Installation: The framework automatically installs hardware drivers, Microsoft Visual C++ Redistributables, .NET Runtime, and other required software dependencies to ensure that the workstation is fully operational immediately after deployment.

Stage 7: System Configuration: PowerShell and Batch scripts apply predefined Windows settings, configure network parameters, enable required Windows features, execute organization-specific configuration scripts, and perform additional post-installation customization according to deployment policies.

Stage 8: Deployment Completion and Verification: The deployment process concludes with automated verification of installation, software provisioning, driver deployment, and system configuration. The workstation is then prepared for immediate operational use, reducing the need for further manual administrative intervention.



Figure 1. Deployment Workflow

Figure 1 illustrates the proposed NetForge deployment workflow from network boot to deployment completion. The workflow combines operating system deployment, system initialization, software selection, application provisioning, driver installation, and system configuration into a sequential process.

This integrated workflow reduces the need to perform these activities separately and provides a structured approach to workstation preparation.

The proposed workflow integrates OS deployment, software provisioning, driver installation, and system configuration into one automated process while maintaining administrator control. It addresses identified gaps by reducing deployment complexity, improving consistency, minimizing configuration drift, and supporting scalability.

VII. CONCLUSION

This survey reviewed enterprise network boot, Windows operating system deployment, software provisioning, driver and runtime installation, and system configuration

automation. The comparative analysis found that these capabilities are distributed across specialized technologies rather than being provided through a lightweight workflow. PXE, DHCP, and TFTP provide the network-boot infrastructure, while WDS, Windows ADK, and WinPE support operating-system deployment. PowerShell provides system-level automation, whereas Winget and Chocolatey support application provisioning. Enterprise deployment and management platforms provide centralized management capabilities.

A key finding of the investigation is that post-installation activities remain fragmented across different tools and deployment stages. Software installation, driver deployment, runtime configuration, and system customization require additional automation after operating system deployment. The analysis also found a trade-off between comprehensive enterprise deployment platforms and lightweight, modular approaches, particularly regarding infrastructure complexity and administrative requirements.

Based on these findings, NetForge: Enterprise Network Boot and Infrastructure Automation is proposed as a conceptual integration framework for coordinating network boot, operating-system deployment, software provisioning, driver and runtime installation, and system configuration within a structured workflow. Its contribution is the integration and orchestration of existing technologies rather than replacing them.

As this study is a survey and conceptual proposal, the identified advantages of NetForge are research-derived expectations rather than experimentally measured improvements. Future implementation and controlled evaluation are required to measure deployment time, resource utilization, reliability, scalability, and consistency. The findings establish the basis for developing and evaluating NetForge.

FUTURE SCOPE

Although NetForge focuses on Windows workstation deployment and software provisioning, the framework can be extended in several directions to enhance its capabilities and applicability.

Future enhancements may include:

- Automated Windows Update and patch management.
- Deployment scheduling through a web-based dashboard.
- Cross-platform deployment support for Linux and Windows.
- Deployment history
- Additional package managers and software repositories
- Cloud-assisted deployment and centralized management.

These enhancements would transform NetForge into a comprehensive enterprise endpoint provisioning platform capable of supporting modern, scalable, and automated infrastructure management.

ACKNOWLEDGMENT

The authors sincerely thank the Department of Computer Science and Engineering, SOCSE, Sandip University, Nashik, Maharashtra, India, for providing the academic environment and facilities to carry out this survey work. They express their heartfelt gratitude to Dr. Amarsinh Varpe for his valuable guidance, continuous support, and constructive suggestions throughout the research. The authors also thank the faculty members of the department for their encouragement and academic assistance, which greatly contributed to the successful completion of this survey paper.

REFERENCES

- [1] M. Wurster, U. Breitenbücher, M. Falkenthal, C. Krieger, F. Leymann, K. Saatkamp, and J. Soldani, "The essential deployment metamodel: A systematic review of deployment automation technologies," *SICS Software-Intensive Cyber-Physical Systems*, vol. 35, nos. 1–2, pp. 63–75, doi: 10.1007/s00450-019-00412-x, Aug. 2020.
- [2] J.-P. Arcangeli, R. Boujbel, and S. Leriche, "Automatic deployment of distributed software systems: Definitions and state of the art," *Journal of Systems and Software*, vol. 103, pp. 198–218, doi: 10.1016/j.jss.2015.01.040, May 2015.
- [3] B. Fitzgerald and K.-J. Stol, "Continuous software engineering: A roadmap and agenda," *Journal of Systems and Software*, vol. 123, pp. 176–189, doi: 10.1016/j.jss.2015.06.063, Jan. 2017.
- [4] Intel Corporation, *Preboot Execution Environment (PXE) Specification*, Version 2.1, Intel Corporation, Sept. 1999.
- [5] R. Droms, "Dynamic Host Configuration Protocol," RFC 2131, Internet Engineering Task Force (IETF), Mar. 1997.
- [6] K. Sollins, "The TFTP Protocol (Revision 2)," RFC 1350, Internet Engineering Task Force (IETF), Jul. 1992.
- [7] Microsoft Corporation, "Windows Deployment Services (WDS)," *Microsoft Learn*. [Online]. Available: [Accessed: Jul. 28, 2026].
- [8] Microsoft Corporation, "Windows Assessment and Deployment Kit (Windows ADK)," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/windows-hardware/get-started/adk-install>. [Accessed: Jul. 28, 2026].
- [9] Microsoft Corporation, "Windows Package Manager (WinGet)," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/windows/package-manager/>. [Accessed: Jul. 28, 2026].
- [10] Microsoft Corporation, "PowerShell Documentation," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/powershell/>. [Accessed: Jul. 28, 2026].
- [11] M. Shahin, M. A. Babar, and L. Zhu, "Continuous integration, delivery and deployment: A systematic review on approaches, tools, challenges and practices," *IEEE Access*, vol. 5, pp. 3909–3943, doi: 10.1109/ACCESS.2017.2685629, 2017.
- [12] M. Shahin, M. Zahedi, M. A. Babar, and L. Zhu, "An empirical study of architecting for continuous delivery and deployment," in *Proc. ACM/IEEE Int. Symp. Empirical Software Engineering and Measurement (ESEM)*, 2018.
- [13] Microsoft Corporation, "Windows Deployment Scenarios," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/windows/deployment/>. [Accessed: Jul. 28, 2026].
- [14] Microsoft Corporation, "Windows Preinstallation Environment (Windows PE)," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/windows-hardware/manufacture/desktop/winpe-intro>. [Accessed: Jul. 28, 2026].
- [15] Microsoft Corporation, "Microsoft Deployment Toolkit (MDT) Documentation," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/previous-versions/windows/it-pro/windows-10/deployment/deploy-windows-mdt/>. [Accessed: Jul. 28, 2026].
- [16] Microsoft Corporation, "Microsoft Configuration Manager (MECM) Documentation," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/mem/configmgr/>. [Accessed: Jul. 28, 2026].
- [17] Microsoft Corporation, "Microsoft Intune Documentation," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/mem/intune/>. [Accessed: Jul. 28, 2026].
- [18] J. Humble and D. Farley, *Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation*. Boston, MA, USA: Addison-Wesley Professional, 2010.
- [19] K. Morris, *Infrastructure as Code: Managing Servers in the Cloud*, 1st ed. Sebastopol, CA, USA: O'Reilly Media, 2016.
- [20] A. Proulx and F. Raymond, "Problems and Solutions of Continuous Deployment: A Systematic Review," *arXiv preprint arXiv:1812.08939*, doi: 10.48550/arXiv.1812.08939, Dec. 2018.
- [21] A. Rahman, R. Mahdavi-Hezaveh, and L. Williams, "A systematic mapping study of infrastructure as code research," *Information and Software Technology*, vol. 108, pp. 65–77, doi: 10.1016/j.infsof.2018.12.004, Apr. 2019.
- [22] L. Leite, C. Rocha, F. Kon, D. Milojevic, and P. Meirelles, "A survey of DevOps concepts and challenges," *ACM Computing Surveys*, vol. 52, no. 6, Art. no. 127, doi: 10.1145/3359981, Nov. 2019.
- [23] R. N. Rajapakse, M. Zahedi, M. A. Babar, and H. Shen, "Challenges and solutions when adopting DevSecOps: A systematic review," *ACM Computing Surveys*, vol. 55, no. 10, Art. no. 214, doi: 10.1145/3539731, Oct. 2022.
- [24] D. Taibi, V. Lenarduzzi, and C. Pahl, "Continuous architecting with microservices and DevOps: A systematic mapping study," *Journal of Systems and Software*, vol. 182, Art. no. 111065, doi: 10.1016/j.jss.2021.111065, Dec. 2021.
- [25] M. Burgess, *Promise Theory: Principles and Applications*. Chichester, U.K.: Wiley, 2015.
- [26] M. Burgess, "CFEngine: A site configuration engine," *USENIX Computing Systems*, vol. 8, no. 3, pp. 309–337, Summer 1995.
- [27] Northern.tech, "CFEngine Documentation – How CFEngine Works." [Online]. Available: <https://docs.cfengine.com/>. [Accessed: Jul. 28, 2026].
- [28] Microsoft Corporation, "Windows Autopilot Documentation," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/autopilot/>. [Accessed: Jul. 28, 2026].
- [29] Microsoft Corporation, "Provisioning Packages for Windows," *Microsoft Learn*. [Online]. Available: <https://learn.microsoft.com/windows/configuration/provisioning-packages/>. [Accessed: Jul. 28, 2026].

- [30] Northern.tech, "CFEngine Documentation – Infrastructure Automation." [Online]. Available: <https://docs.cfengine.com/>. [Accessed: Jul. 28, 2026].
- [31] A. B. Varpe, Y. D. Rajendra, A. D. Vibhute, S. V. Gaikwad, and K. V. Kale, "Identification of plant species using non-imaging hyperspectral data," in *Proc. 2015 International Conference on Man and Machine Interfacing (MAMI)*, pp. 1–4, doi: 10.1109/MAMI.2015.7456613, Dec. 2015.
- [32] A. B. Varpe, R. R. Surase, A. D. Vibhute, S. V. Gaikwad, Y. D. Rajendra, K. V. Kale, and S. C. Mehrotra, "Synygium cumini plant photosynthetic pigment detection from hyperspectral datasets using spectral indices," in *Proc. 2017 2nd International Conference on Man and Machine Interfacing (MAMI)*, pp. 1–6, doi: 10.1109/MAMI.2017.8307882, Dec. 2017.
- [33] A. B. Varpe, R. R. Surase, A. D. Vibhute, D. B. Nalawade, and K. V. Kale, "Role of lens position and illumination source for acquiring non-imaging hyperspectral data to estimate biophysical characteristics of leaves," in *Recent Trends in Image Processing and Pattern Recognition*, pp. 391–396, Springer, Singapore, doi: 10.1007/978-981-13-9187-3_35, 2019.
- [34] M. El Rajab, L. Yang, and A. Shami, "Zero-touch networks: Towards next-generation network automation," *Computer Networks*, vol. 243, Art. no. 110294, doi: 10.1016/j.comnet.2024.110294, Apr. 2024.
- [35] L. Yang, M. El Rajab, A. Shami, and S. Muhaidat, "Enabling AutoML for Zero-Touch Network Security: Use-Case Driven Analysis," *IEEE Transactions on Network and Service Management*, vol. 21, no. 3, pp. 3555–3582, doi: 10.1109/TNSM.2024.3376631, Jun. 2024.
- [36] M. Fuchs, L. E. Ardila-Perez, T. Mehner, and O. Sander, "Split Boot—True Network-Based Booting on Heterogeneous MPSoCs," *IEEE Transactions on Nuclear Science*, vol. 70, no. 6, pp. 1157–1163, doi: 10.1109/TNS.2023.3237968, Jun. 2023.
- [37] P. Stöckle, B. Grobauer, and A. Pretschner, "Automated Implementation of Windows-related Security-Configuration Guides," in *Proc. 35th IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pp. 598–610, 2020, doi: 10.1145/3324884.3416540. Sep 2020.

