**Name**: Saniya Galib Tamboli

**Email**: [saniyagtamboli2412@gmail.com](mailto:saniyagtamboli2412@gmail.com)

**Batch:** DWS\_B5\_25VID2550

**User Id:** 34749

**Date**: 05/08/2025

**Assignment**

**1. Why Application Packaging Matters**

* **Manages complexity**: Centralizes installation/config for hundreds of software across systems.
* **Streamlines deployment**: Automates installs and reduces IT support load.
* **Benefits**:

1. Consistent environment.
2. Efficient deployment.
3. Security risk mitigation.
4. Decreases business risk.
5. Reduced support and admin costs.

**2. Industry Trends**

* Companies want only one person to handle the entire lifecycle (discovery → packaging → UAT).
* **Reasons**:
  1. Cost cutting
  2. Speed and process efficiency.
* **Challenge**: Finding talent with full-stack packaging skills.

**3. What Packaging Cannot Do**

* It’s not a magic fix for all compatibility issues.
* Containerization helps some issues:
  1. **App-V** and **MSIX** run apps in isolated containers.
  2. For older apps, tools like **ACT**, **Desktop Analytics** or **remediation techniques** may help.

**4. End-to-End Application Packaging Steps**

* 1. **Application Discovery**
* **Goals**:
  + 1. Validate installer sources.
    2. Verify app works in your organization environment.
    3. Document every requirement.
* **Key Tasks**:
  + 1. Identify user-specific configurations.
    2. Split into main package + business-area configuration packages if needed.
    3. Check for prerequisites (isolate them into separate reusable packages).

**4.2 Application Packaging**

* Based on discovery.
* Customize based on configs.
* Follow standards for reusability (especially important in MSIX & App-V).

**4.3 User Acceptance Testing (UAT)**

* Real users test the packaged app.
* Validates performance, functionality, stability in real-world conditions.

**4.4 QA (Quality Assurance)** (Optional but it is recommended)

* Senior packager reviews the build for compliance, errors, and optimization.

**5. MSIX Modification Packages**

* MSIX allows modular customization using Modification Packages.
* Useful when different business areas need different configurations layered over the same base app.

**6. Windows 10 vs Windows 11 in App Packaging Context**

**Windows 11 Advantages:**

* Modern UI & Snap Layouts.
* Better performance & security (TPM 2.0, Windows Hello).
* Smaller, faster updates.
* AI Assistant-Copilot.
* DirectStorage for gaming and better performance apps.

**Windows 10 Advantages:**

* Familiar interface.
* Wider hardware and software compatibility.
* Stable, cost-effective for legacy systems.  
    
  **Considerations:**
* Check app compatibility.
* Consider hardware specs for Windows 11.
* Older apps may perform better on Win 10.

Windows 11 built upon Windows 10’s foundation which offers more modern and refined user experience with enhanced features and performance. Both OS are compatible with a wide range of apps, including those within an “App Pack”.

**7. MSI (Window Installer) Contexts**

|  |  |  |  |
| --- | --- | --- | --- |
| **Context** | **Runs As** | **Access Level** | **Use Case** |
| **User** | Current user | Limited to profile | Personal apps, user-specific tasks |
| **System** | SYSTEM account | Full system-wide | Global installs, services, scheduled tasks |
| **Admin** | Elevated privileges/permissions | Full + UAC prompt | Changes to services, drivers, etc. |

**8. Logon Scripts + Active Setup**

**1. Use of Active Setup**

* **Purpose**: Executes user-specific tasks at user logon
* **Action**: Copy files, update registry, run scripts within MSI.
* **Example**: Copy config files to AppData from machine-level location.

**2. Creating & Assigning Logon Scripts**

* **Script types**: .bat, .ps1 (PowerShell), .vbs – choosen based on need.
* **Function**: Copy user files/settings from shared drive to user profile folder.
* **Assignment**: Use Group Policy to assign to:
  1. Individual users
  2. User groups
  3. Organizational Units (OUs)

**3. Deployment Strategies**

* **Group Policy**:
  1. Assign logon scripts by user/group/OU.
  2. Distribute MSI that uses logon scripts/Active Setup.
* **Software Distribution**:
  1. Tools like SCCM or Intune can also deploy MSIs with scripts.
* **Language Choice**:
  1. Use batch for basic logic.
  2. Use PowerShell for complex actions (preferred for modern systems).

**4. Example Scenario – Settings Deployment**

* **Need**: App needs to load user-specific settings at login.
* **Steps**:
  1. Add **Active Setup** in MSI to trigger script at login.
  2. Script copies settings from \\server\netlogon\MyApplication to %AppData%\MyApplication.
  3. Deploy via Group Policy or Software Distribution tool.

**5. Best Practices**

* **Error Handling**: Handle failures like network issues gracefully.
* **Security**: Protect sensitive file paths & credentials.
* **Testing**: Always test script in real user environments.
* **Documentation**: Record script logic, deployment method, and config.