**Design**

**For**

**SURGE Auto Updater**

**Version 1.4**

**Submitted by:**

**SOFTSOL RESOURCES, INC.**

**42840 Christy St. Suite 231**

**Fremont, CA 94538**

**www.softsol.com**

**Approval Details**

|  |  |  |  |
| --- | --- | --- | --- |
|  | Prepared by | Reviewed by | Approved by |
| Name | Amar Pai |  |  |
| Designation | Architect |  |  |

**Version History**

|  |  |  |  |
| --- | --- | --- | --- |
| Version  number | Date | Brief description of changes | Author |
| 1.0 | 07/03/2025 | Initial design document | Amar Pai |
| 1.1 | 07/09/2025 | Added option to update using MSI | Amar Pai |
| 1.2 | 07/10/2025 | Added authentication to download update | Amar Pai |
| 1.3 | 08/04/2025 | Changes based on 8/4 meeting  Use ZIP instead of MSI  Added MECM requirements | Amar Pai |
| 1.4 | 09/11/2025 | Updates based on 9/9 meeting | Amar Pai |

**Contents**

1. **Overview**

This document outlines the requirements and design for the SURGE .NET automatic update module. The information in this document will supersede any prior discussions, emails or documents that were shared between SoftSol and IBM.

1. **Reference**

The following information was referenced in preparing this design document.

* SURGE .NET Thick Client Setup.docx
* Relevant figures from the document are included in the appendices.
* Meeting on 08/04/2025 between all stakeholders.
* Meeting on 09/09/2025 between all stakeholders.
* Meeting on 09/17/2025 that confirmed using ZIP file approach

1. **Key Requirements**

DHCS has three high-level requirements as it relates to installation and updates. The scope of this document is to outline these requirements, identify the in-scope and out-of-scope components and provide a design for the in-scope components.

1. **Initial Installation**

The key requirements of the initial installation component are:

1. Generate an install package for each environment
2. Transfer the install package to the client workstation
3. Install the package on the client workstation
4. Grant permissions to the install folder for all users
5. Create desktop and program file shortcuts
6. **Automatic Updates**

The key requirements of the automatic update component are:

1. Generate an update package for each environment
2. Determine whether the thick client needs to be updated (upgrade or downgrade)
3. If update is required, download the update package to the client workstation
4. Extract the update package on the client workstation
5. Install the update package on the client workstation
6. **Environment Switching**

The key requirements of environment switching are:

1. Provide users the ability  to run SURGE in different environments (e.g., UAT, PRD)
2. End user should not have to perform low-level tasks such as setting environment variables, copying files/folders, etc.
3. Determination of which environments a user requires on their workstation will be handled through service request.
4. **In Scope**

The following components are in scope for this project.

1. **Initial Installation**
2. Provide specifications to DevOps team to build the initial installation ZIP package.
3. Provide specifications to MECM team in deploying the ZIP package to client workstation.
4. Collaborate on troubleshooting and fine-tuning the solution.
5. **Automatic Updates**
6. Provide specifications to DevOps team to build the update ZIP package.
7. Develop an endpoint for the thick client to validate it is running the correct version.
8. Develop an endpoint to download the latest update package to the thick client.
9. Validate that thick client is running the correct version and whether update is required.
10. Download and install the update package on the thick client.
11. Collaborate on troubleshooting and fine-tuning the solution.
12. **Environment Switching**
13. Enhance SURGE.NET to use configurations instead of environment variables to facilitate environment switching.
14. **Out Of Scope**

The following are out of scope of this project.

1. Design and development of MECM components are out of scope.
2. Design and development of the DevOps pipeline scripts are out of scope.
3. Validation of user access to specific environments is out of scope.
4. Partial or delta updates are out of scope – update package will contain entire application.
5. Environment switching from within the SURGE.NET application is out of scope.
6. Graphical user interface for environment switching is out of scope.
7. **Assumptions & Constraints**
8. **Assumptions**
9. Version update will be performed after user login. If user is unable to login to SURGE or does not have permissions for the environment, then the application will not update for that environment.
10. Environment switching will not validate whether a user has access to an environment or not. Access control will be handled by the SURGE.NET application as is currently done.
11. Client workstation will be configured with the required permissions to terminate running thick client application.
12. Client workstation will be configured with the required permissions to add, update or delete folders and files within the SURGE.NET installation directory.
13. Client workstation will have high-speed network access and sufficient bandwidth to download the thick client update package.
14. Thick client will force an update if the current version does not match the expected version from the server. Any failure during update process will render the thick client unusable unless the update error is remediated.
15. IIS has been configured to allow large file downloads.
16. Icons will be available for SURGE shortcuts for each environment.
17. **Design Constraints**
18. None
19. **Module Design**
20. **DevOps Pipeline Changes**

The following steps should be added to the DevOps pipeline that builds the thick client. The same process will apply to both initial install and auto update.

1. **One-time IIS Setup**
2. Configure IIS to support large file downloads.
3. IIS by default supports a maximum of ~30MB file download.
4. This should be changed to 120MB to support the thick client ZIP file size of ~90MB plus a buffer.
5. **Thick Client Build**
6. Build the thick client as currently implemented.
7. The output from the build is a directory that contains the thick client EXE, DLL’s, configuration files, etc.
8. A screenshot of a computer

   AI-generated content may be incorrect., PictureScreenshot is shown below.

1. Determine the current version of the SURGE.NET thick client.
2. From the folder in step 3.1, locate a file called Cammis.Surge.Client.exe.
3. User PowerShell to get the version information for this file. This is the thick client version.
4. This version number will be used later in the process.
5. A screenshot is included below.

A screenshot of a computer

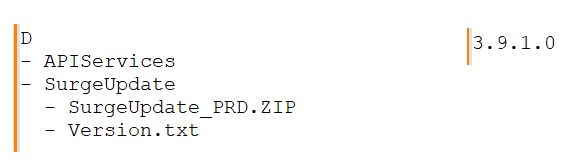
AI-generated content may be incorrect., Picture

1. Prepare the update package.
2. Create a folder with name SurgeUpdate.
3. Copy the contents of the build directory from step 1 to the SurgeUpdate folder.
4. Copy environment specific configurations to the SurgeUpdate folder.
5. The environment specific configurations are in the Config folder in GIT.
6. For example, for UAT environment, copy the three files from UAT folder to SurgeUpdate folder.
7. This will overwrite the original three files in SurgeUpdate folder.
8. A screen shot of a computer program

   AI-generated content may be incorrect., PictureFolder structure is shown below.

1. Build the ZIP file.
2. The ZIP file name template is SurgeUpdate\_env.ZIP.
3. For example, for UAT, the file name will be SurgeUpdate\_UAT.ZIP.
4. ZIP the SurgeUpdate folder from step 2 to the above file name.
5. Publish the ZIP file to GIT.
6. The file should be uploaded to the tar-surge-client-deployment repository.
7. A screen shot of a computer program

   AI-generated content may be incorrect., PictureUpload the file to folder tar-surge-client in above repository.

1. Publish the ZIP and version files to the SURGE API server.
2. Create a directory D:\SurgeUpdate on each of the API servers.
3. Copy the ZIP file from step 4 to this directory.
4. Create a version file in this directory using the version from step 2.
5. Version file name is Version.TXT
6. It contains a single line with the version number.
7. Ensure this is propagated to all API servers.
8. **MECM Changes**

The following steps should be included in the MECM package. This list is not intended to be exhaustive but a guideline. Four packages will be created – one for each of the environments DEV, UAT, SIT and PRD. The steps below are for a single environment and are identified by the placeholder “env”.

1. From GIT, use the ZIP file created by the DevOps pipeline matching the package environment.
2. For example, when building the SIT install package, use the file SurgeUpdate\_SIT.ZIP.
3. Build the MECM install package (not in this document scope).
4. Transfer the install package to the client workstation (not in this document scope).
5. Install the package on the client workstation. While not in this document scope, some of the key steps are listed below.
6. Terminate any running instances of Forte SURGE application.
7. Terminate any running instances of .NET SURGE application.
8. EXE name is Cammis.Surge.Client.exe
9. Check if the environment specific SURGE folder exists
10. For example, for SIT, the folder is C:\SurgeApp\SIT
11. If it exists, delete the folder and all subfolders.
12. Unzip the contents of the SurgeUpdate\_env.ZIP file to the folder C:\SurgeApp\env.
13. For example, for SIT, the SurgeUpdate\_SIT.ZIP will be unzipped to C:\SurgeApp\SIT
14. Grant full access to all users to the SurgeApp folder and all subfolders.
15. Create shortcut on the desktop.
16. The EXE file name is C:\SurgeApp\env\Cammis.Surge.Client.exe
17. The icon file is C:\SurgeApp\env\icon.ico
18. A white background with black text

    AI-generated content may be incorrect., PictureScreenshot of the install folder is shown below.
19. **Server Side – Get Client Version**
20. Enhance the GetThickClientConfiguration action in ConfigMgr to return the version file.
21. Read the SurgeUpdateVersion.json file from the above directory.
22. Include the version in the API response.
23. **Server Side – Get Client Update Package**
24. Add new action GetThickClientUpdatePackage in ConfigMgr.
25. On invocation, read the SurgeUpdate.ZIP file from the above directory.
26. Stream the file to the thick client as part of controller action response.
27. **Client Side – Check For Version Update**
28. Update the startup Program file to check for version update.
29. Capture the version number from the response to the GetThickClientConfiguration endpoint.
30. Compare the version number to the version of the thick client executable.
31. If the version number is different, initiate the download process.
32. Delete directories created by prior updates in the thick client root directory.
33. Directory for backup – backup
34. Directory for downloaded file - download
35. Create a directory called backup in the thick client root directory. This directory will be used to restore the application in case the update fails.
36. Copy the contents of the thick client root directory into the backup directory.
37. Create a directory called download in the thick client root directory.
38. Invoke the GetThickClientUpdatePackage to get the update package file from the API server.
39. Save the ZIP file to the download directory.
40. Extract the contents of the ZIP file to the download directory.
41. Execute the SurgeUpdate.bat batch from the client root directory.
42. This should update the thick client and launch it.
43. **Client Side – Execute Update**
44. Create a new batch file called SurgeUpdate.bat in the SURGE thick client source root directory.
45. When executed, the batch file will perform the following actions.
46. Terminate the thick client program if it is running.
47. Delete the thick client executable files from the thick client root directory.
48. Copy the contents of the download directory to the thick client root directory.
49. Launch the thick client application.
50. **Environment Switching**
51. Update SURGE.NET code to not require environment variables on client workstations.
52. Integrate any environment specific requirements into the install and update package.
53. **Error & Exception Handling**

|  |  |
| --- | --- |
| **Error** | **Action** |
| Failure to download the application | Application will display error to the end user and include in log file. Application will not launch. |
| Failure to install an update | Application will restore the previous version from local backup created during the update process. |
| Failure to restore the previous version | Depending on the cause of the failure, user may have to perform a fresh install of the application. |
| Failure to read the update version file in the API server | Application will display error to the end user and include in log file. Application will not launch |

**Appendix A – Installation Workflow**

A diagram of a computer

AI-generated content may be incorrect., Picture

**Appendix B – Auto Update Workflow**

A computer with a computer system

AI-generated content may be incorrect., Picture