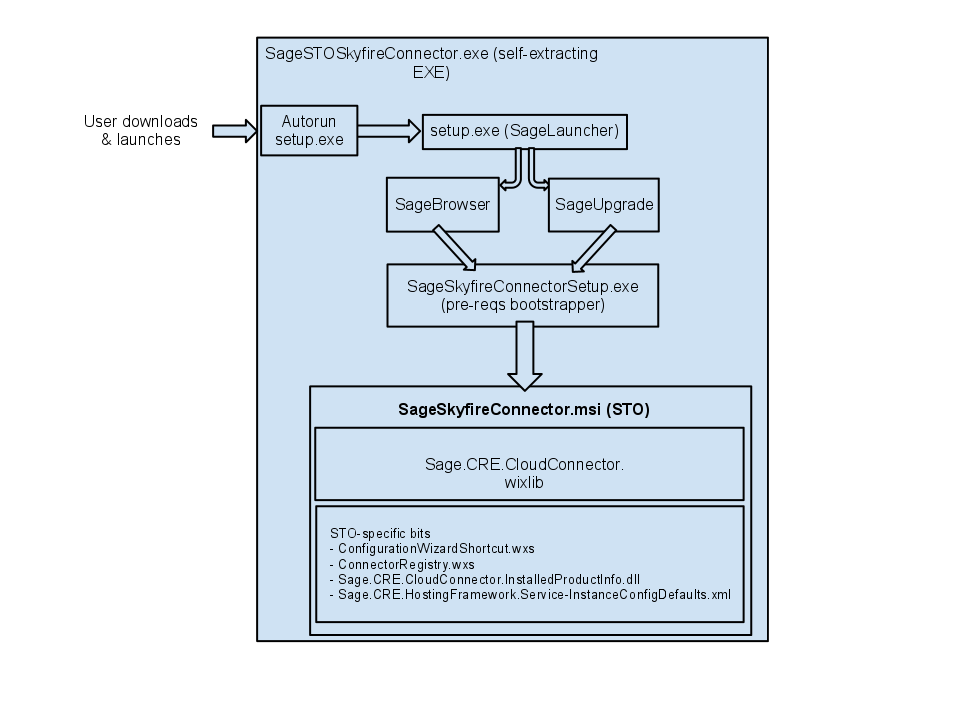
# Skyfire Connector Installation Usage Scenarios, Packages, Image Structure, and Design

There are 2 basic usage scenarios supported by the Skyfire Connector design:

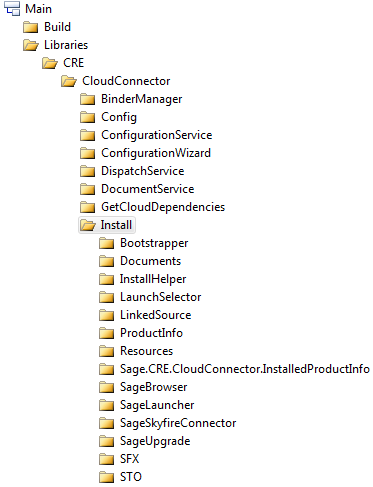
1. Full install from scratch – this is what occurs when the user downloads and runs the Skyfire Connector Install package on a machine which doesn't currently have a Skyfire Connector on it
2. Upgrade install – this is what occurs when the user downloads and runs the Skyfire Connector Install package on a machine which already has a prior Skyfire Connector version on it; this will normally happen when the user is notified, via various notification mechanisms, of a Skyfire Connector upgrade (either optional, or required)

******

## Product-Specific Install Design

The Install codebase is located in the CloudConnector\Install folder. The code is structured to enable creation of multiple product-specific Skyfire Connector installations. The reason for having a product-specific install is so that we can manage the upgrade/service scenarios for each integrated product completely independently from each other, it also supports the requirement of allowing us to install multiple Skyfire integrations on a single machine.

Although the end result produces is a product-specific install, the bulk of the code in CloudConnector\Install tree is entirely product-neutral. During the build a copy of the code is made for each product-specific install … that code is then customized by the product-specific logic.



"Bootstrapper" – contains the packages needed to generate the SageSkyfireConnectorSetup.exe bootstrapper package. These packages are snapshots taken from the Windows SDK.

"Documents" – contains documents displayed in the SageBrowser/SageUpgrade.

"InstallHelper" – an unmanaged custom action binary that can be scripted into the install … to facilitate executing arbitrary code during install, upgrade, or uninstall.

"LaunchSelector" – an unmanaged EXE used to select between auto-running SageBrowser or SageUpgrade when the install package is executed.

"LinkedSource" – shared source files compiled into multiple Install projects

"ProductInfo" – an unmanaged static library which exposes functions for determining if a product is installed

"Sage.CRE.CloudConnector.InstalledProductInfo" – a .NET assembly containing utility functions for discovering, at runtime, information about what Skyfire Connector product is installed

"SageBrowser" – the install screen browser user experience

"SageLauncher" - a helper tool which facilitates the launching of other programs. This is used by various install/upgrade processes to invoke programs that require arguments and/or special path knowledge.

"SageUpgrade" - the upgrade process for Skyfire Connector. This program would be run as an alternative to SageBrowser.exe by a customer who has already installed a prior version of the. SageUpgrade is a derivative of the SageBrowser codebase. The SageUpgrade.exe is meant to perform a full upgrade.

"SFX" – source files used to create the self-extracting EXE.

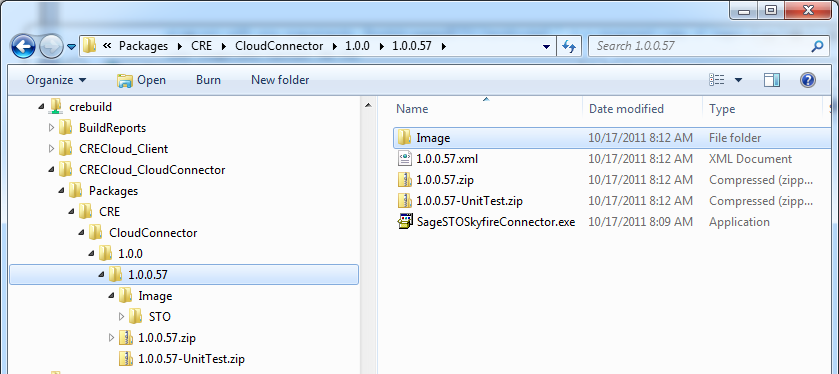
"STO" – contains the product-specific code bits for the Sage Timberline Office Skyfire Connector install

## Install Building

The CRE\_CloudConnector.fbp7 contains 2 Action Lists (i.e., BuildProductInstallTools and BuildProductInstall) which do the work of creating a product-specific install from the product-neutral codebase located at CloudConnector\Install.

When the install is built it places a self-extracting EXE, which is the installation package, in the drop folder (e.g., SageSTOSkyfireConnector.exe).

Following is a description of the contents of the drop folder:



“Image” – this is literally the uncompressed contents of the self-extracting EXE's for each Skyfire integration product (e.g., STO, STE, SMB, etc.). A detailed description of the contents of the Image folder follows in the next section of this document.

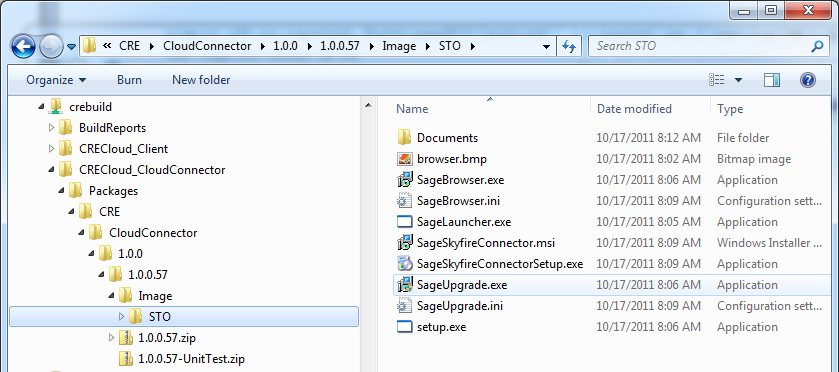
“1.0.0.57.xml” – this is a version file produced automatically by the build system for every build project; **it is produced as a build system convention**

“1.0.0.57.zip” – the “dev package” for the CloudConnector project; contains source code and as well the above “Image” folder contents; **it is produced as a build system convention**

“1.0.0.57-UnitTest.zip” – the “unit test package” for the CloudConnector project; contains (theoretically) compiled unit tests that are owned by the CloudConnector library; **it is produced as a build system convention**

SageSTOSkyfireConnector.exe – this is the self-extracting EXE installer for STO integration with Skyfire

Following is a description of the contents of the Image\STO folder:



“SageBrowser.exe, SageBrowser.ini, browser.bmp” – the files which make up the SageBrowser described previously

“SageLauncher.exe” – the SageLauncher described previously

“SageSkyfireConnector.msi” – the installation MSI package.

“SageSkyfireConnectorSetup.exe” – the bootstrapper EXE for the MSI install. Ensures that pre-requisites are installed and then chain-installs the SageSkyfireConnector.msi.

“SageUpgrade.exe, SageUpgrade.ini, browser.bmp” – the files which make up the SageUpgrade described previously

"setup.exe" – actually a renamed copy of the LaunchSelector.exe … a simple application which determines whether a prior version of Skyfire Connector is install and then launches either SageBrowser.exe or SageUpgrade.exe accordingly. The setup.exe is the command that automatically is executed when the self-extracting EXE completes initial extraction.

## Knowledge Transfer Summary Points

1. Install consists of a MSI deployment (Windows Installer) wrapped inside a self-extracting EXE, suitable for download
2. MSI is built using WiX (Windows Installer XML) Toolset; full install is comprised of multiple wixlib's built in both Platform and CloudConnector
3. Product-specific logic should be isolated (e.g., in the CloudConnector\Install\STO folder for STO product specific logic)
4. Design imperative: do as little during the actual install as possible … lay down files, and that’s it. Install is install, configuration is **after** install.
   1. LibraryConfigTool.exe takes as input a LibraryConfig-Library.xml; it then outputs \*.wxs (WiX source files)
   2. \*.wxs is input into candle.exe & light.exe (WiX tools); they output a \*.wixlib
   3. The SageSkyfireConnector.wxs + \*.wixlibs are input into candle.exe & light.exe; the output is the SageSkyfireConnector.msi
5. Upgrades are implemented as “MSI Major Upgrade”; this means that with each upgrade the product is, essentially, uninstalled and reinstalled … this approach affords us maximum flexibility in what kinds of changes can be in a given release (i.e., don’t have to worry about limitations of deleting/renaming files in an upgrade)
6. Each build of the Skyfire Connector install produces a full self-extracting EXE, which includes support for both “clean install” and “upgrade install” scenarios
7. Each new release requires a new “MSI product code” … so that Windows Installer will process the upgrade as a “MSI Major Upgrade”