# Common Libraries for Deployment in SICCARV3

## Approach

I’ve put this paper together as it turned out not to be as straight forward as first thought.

Common libraries are libraries that are required by more than one project – typically definitions of data required at each end of a messaging communication.

Note: After much thought, I’ve included the Wallet Library (and one other wallet library) although its currently only used by the wallet service. It’s possible in future it may be needed for use by another component (E.G., a browser wallet) so for consistency with our other libraries and for possible sharing in the future, I have included it as a shared library.

## Siccar Libraries

|  |  |  |
| --- | --- | --- |
| Library | Consumer Services | Notes |
|  |  |  |
| SiccarPlatformCommon.dll | Register Service | I think we can do away with this library with a little effort |
|  | Peer Service |  |
|  | Tenant Service |  |
|  | Wallet Service |  |
|  | Validator Service |  |
|  | Blueprint Service |  |
|  | Action Service? |  |
|  |  |  |
| SiccarPlatformRegisters.dll | Register Service | I’d remove the Transaction model as it’s the only one that’s required by everything. The rest are all used together, I think. This minimises dependencies. |
|  | Peer Service |  |
|  | Validator Service |  |
|  | Wallet Service |  |
|  | Admin UI Client |  |
|  |  |  |
| SiccarPlatformCryptography.dll | Wallet Service | Not shared to date! |
|  |  |  |
| SiccarPlatformWallets.dll | Wallet Service | Not shared to date! |
|  |  |  |
| PeersCore.dll | Peer Service | PeersCore should be only deployed with the Peer Service |
|  | Admin UI Client | This dependency looks bad |
|  |  |  |
|  |  |  |
| Blueprint DTO dll | Blueprint Service | ? |
|  | Action Service |  |
|  |  |  |
| Transaction DTO .dll |  | ? |
|  |  |  |

## Library Pipeline

Each library has its own CI pipeline defined in Azure DevOps as a .yml file. This file is also git controlled and by convention is placed at the same level as the .csproj file – *No Touch Please!.* This pipeline creates and also publishes the NuGet library package to an Azure DevOps Artifacts Feed. Each time the main branch is updated via a merge from a PR, the pipeline will be triggered, and a new version of the package will be created, packed and then published to the Azure DevOps Artifact Feed.

**Important**: the trigger for the pipeline fires from a branch naming convention in this format:

feature/<name-of-library>library/\*

You must use this format when creating your branch otherwise the pipeline will not trigger, and your package will not be published.

e.g.

feature/blueprintlibrary/1234-my-work-item

This is deliberately similar to the naming convention for our microservices:

feature/peerservice/1234-my-work-item

## Importing a NuGet Package from Azure Artifiacts Feeds

These instructions are for the consuming project – the project that needs a reference to the shared library. Once the CI is setup for each shared library you should ***remove*** the reference to the shared library project in your calling project’s .csproj file. The shared library should then be added as a NuGet reference. The main difference being that you now use our private Azure Artifact Feed as the NuGet source and not Nuget.org (the default and also public).

Additionally, how you reference the library is also different depending on what IDE you use.

### Visual Studio

1. In Solution Explorer, right-click the *References* node on your project (not solution) and select *Manage NuGet Packages.*
2. Click the *Browse* tab and change the Package source to **siccarv3feed**. Select the package you want from the list and click *Install* to add it to the project.

### VS Code / Dotnet CLI

1. Ensure you have installed the latest version of the Azure Artifacts Credential Provider ([Download and install the credential provider](https://go.microsoft.com/fwlink/?linkid=2099625))
2. Add a nuget.config file to your project, in the same folder as the .csproj file (or .sln).

<?xml version="1.0" encoding="utf-8"?>

<configuration>

  <packageSources>

    <clear />

    <add key="siccarv3feed" value="https://projectbob.pkgs.visualstudio.com/SICCARV3/\_packaging/siccarv3feed/nuget/v3/index.json" />

  </packageSources>

</configuration>

1. Add a reference to the library package (and version) that you want to use (in the .csproj file)

    <PackageReference Include="Siccar.Platform.Registers" Version="1.0.2"/>

1. Restore packages (using the interactive flag, which allows dotnet to prompt you for credentials)

dotnet restore --interactive

Note: If you have problems accessing the package in Azure DevOps Feeds, then you probably need to check the Azure DevOps permissions. See [feed permissions](https://docs.microsoft.com/en-us/azure/devops/artifacts/feeds/feed-permissions?view=azure-devops).

## Versioning and Updating a Library

Versioning is obviously from the perspective of the library author. The version number will auto increment using the Azure DevOps Pipeline BuildId. We can change this behaviour later if we want to.