Automate Your Revit Add-In Testing with Unit Testing

Patrick Fernbach

Software Engineer

Corey Smith

Software Business Analyst

AUTODESK. UNIVERSITY

Patrick Fernbach

Lead Associate | Mechanical Engineering Software Engineer
KLH Engineers, PSC

Patrick Fernbach specializes in HVAC and plumbing system design at KLH Engineers, PSC, and serves on the software development team. He translates the MEP engineers' needs to software engineers' in order to develop process improvements. He also assists in the creation of Revit add-ins and leads quality assurance and testing efforts to ensure the custom tools are of high quality. Patrick holds a Bachelor of Science in mechanical engineering from the University of Cincinnati.

Corey Smith

Mechanical Designer

Software Business Analyst

KLH Engineers, PSC

Corey Smith is a lead mechanical designer at KLH Engineers, PSC with over 10 years of experience designing commercial, retail and hospitality buildings. As an expert in CAD and Revit, Corey leads a team of in-house software programmers that develop custom tools and workflows that enhance construction document production. He holds a Bachelor of Science in industrial technology with a focus in computer aided drafting from Morehead State University.

Firm Overview

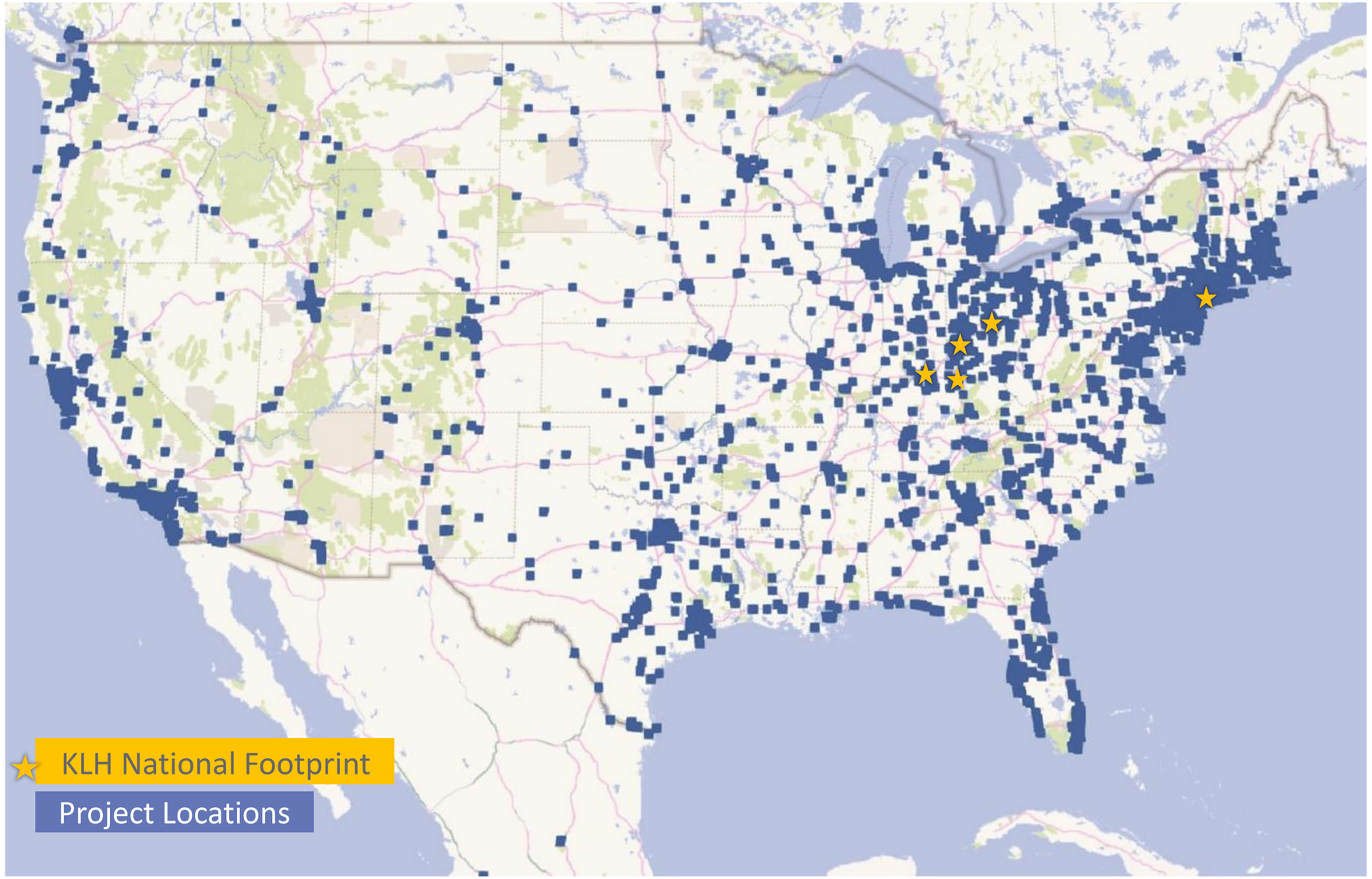
Offices:
Ft. Thomas, KY
Lexington, KY
Louisville, KY
Columbus, OH
New York, NY

Studios:
Healthcare
Education
Commercial
Civic
Retail

Services:
Mechanical
Electrical
Plumbing
Fire Protection
Technology

21,800 projects in 6 years

Licensed in all 50 states



KLH Engineers Background

KLH Engineers is a national MEP firm with a newly formed software department.

Implemented Revit in 2006; went 100% Revit on all projects in 2016.

Developed 1st custom tool in 2008; formed a software department in 2018.

- 2 Software Teams formed; Revit Development Team and a Enterprise Resource Planning (ERP) Development Team

A team of superusers was formed to test tools then roll out to the company. They work with the developers to ensure the software is working correctly.

As a newly formed department, our initial focus wasn’t with testing. To refocus on QA and build and maintain internal relationships, the team had to develop an agile mentality.

Today, we will highlight these obstacles and share solutions, including a step-by-step process to setup unit testing in Visual Studio.

Who are you?

Architects/Engineers/BIM Manager?

Contractors?

Software Engineers? QA?

Learning Objectives

1. EXPLORE END-TO-END TESTS, INTEGRATION TESTS, COMPATIBILITY TESTS, AND UNIT TESTS TO CREATE BETTER REVIT ADD-INS

2. LEARN HOW TO CREATE A UNIT TEST PROJECT IN MICROSOFT VISUAL STUDIO

3. LEARN HOW TO RUN A UNIT TEST ON REVIT

4. LEARN HOW TO APPLY TESTING PHILOSOPHIES TO YOUR TEAM/COMPANY TO CREATE BETTER PRODUCTS AND INCREASE DEVELOPMENT SPEED

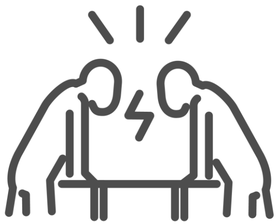
Problems to Overcome

REVIT ADD-INS HAVE BUGS UPON RELEASE

Bugs are present on releases. New features don't work, or new features do work, and old features are broken due to lack of testing. Superusers only testing on their project types, Revit version they used, and scenarios in which were familiar.

TESTING IS NOT CONSISTENT AND IS TAKING TOO LONG

The initial testing process was not consistent. "Hey, can you test this tool?" was the basis of testing and hoping that the "tester" tried to break the tool. Testing was also taking too long; feedback was not getting back to the developers.



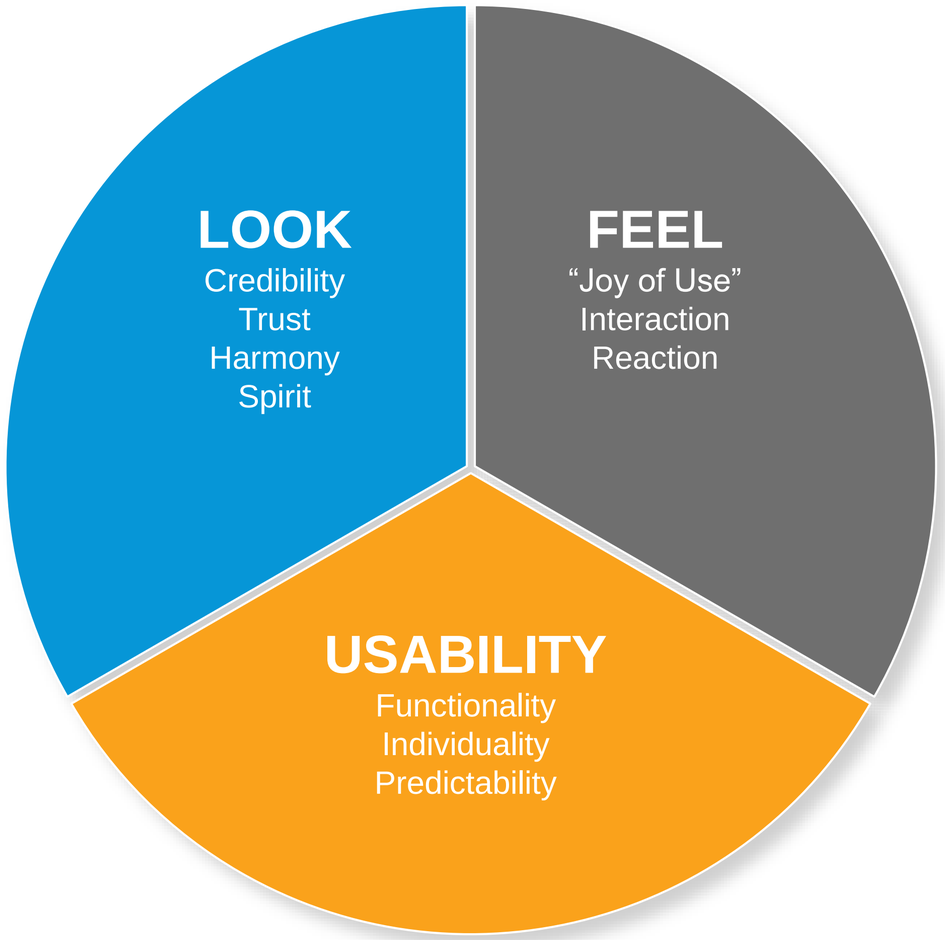
RELATIONSHIPS DETERIORATING BETWEEN SOFTWARE DEVELOPERS AND USERS

Due to bugs and FATAL ERRORS, relationships between developers were strained.

Testing Method 1: End-to-End Testing

SIMULATING THE USER EXPERIENCE

- E2E testing should focus on the user experience.
  - Does the software behave as expected?
  - What reaction do you have when you use the software?
  - How does the software interact with other applications that was developed by another team?

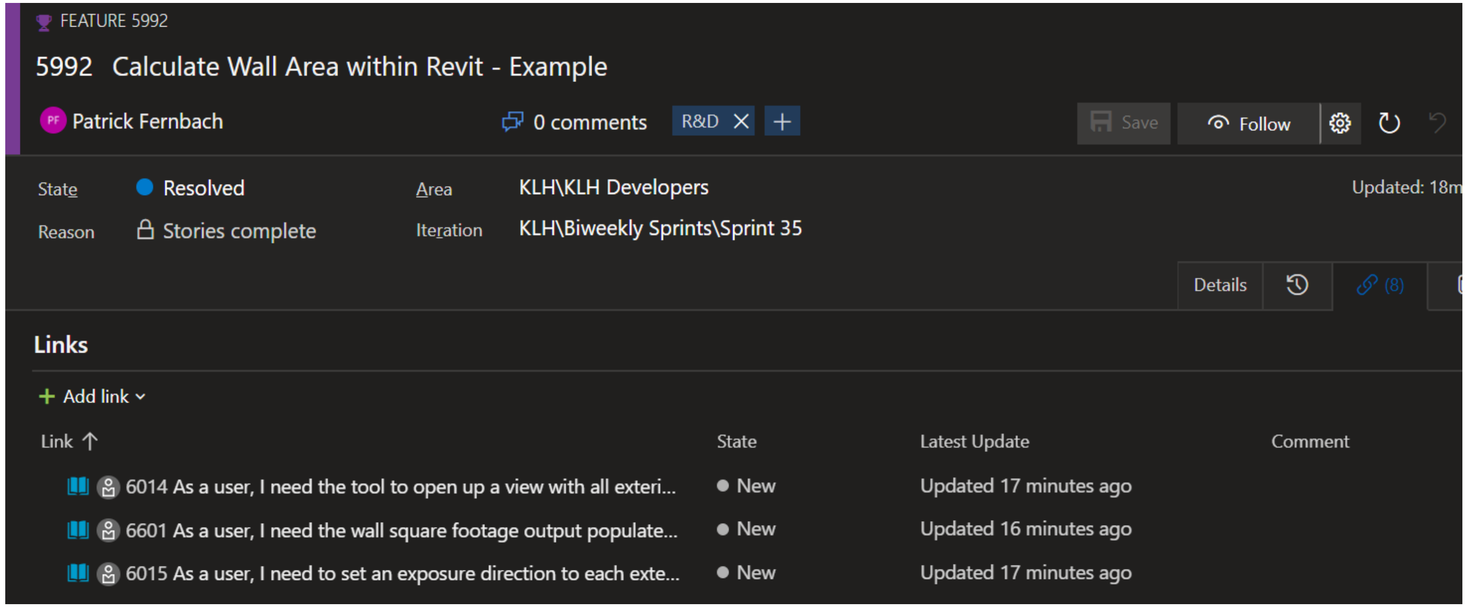


User Experience

Testing Method 1: End-to-End Testing

END TO END TESTING EXAMPLE

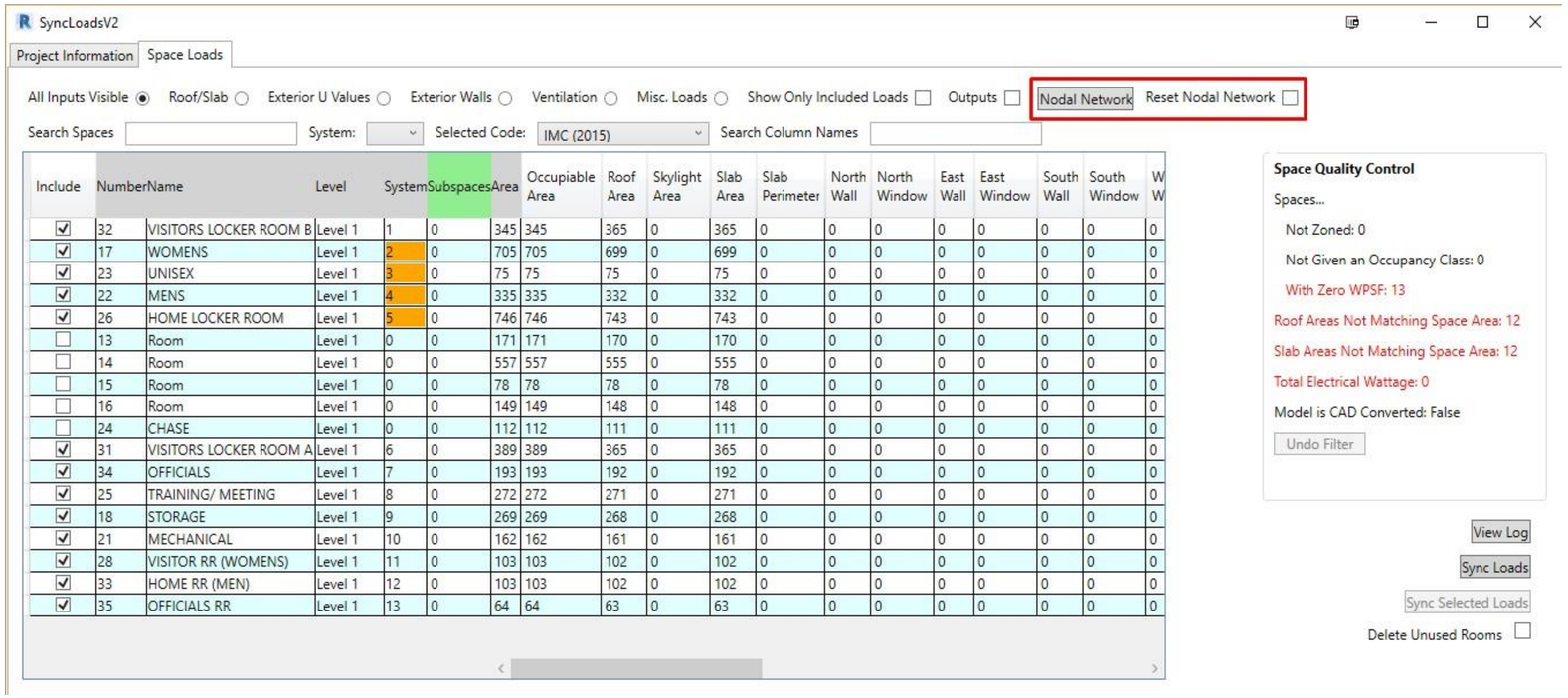
- Feature was requested to have the ability to select an exterior wall in Revit and get the correct square footage of the wall in the correct orientation.
  - **Step 1 – Review the User Stories for the Features**



Testing Method 1: End-to-End Testing

END TO END TESTING EXAMPLE

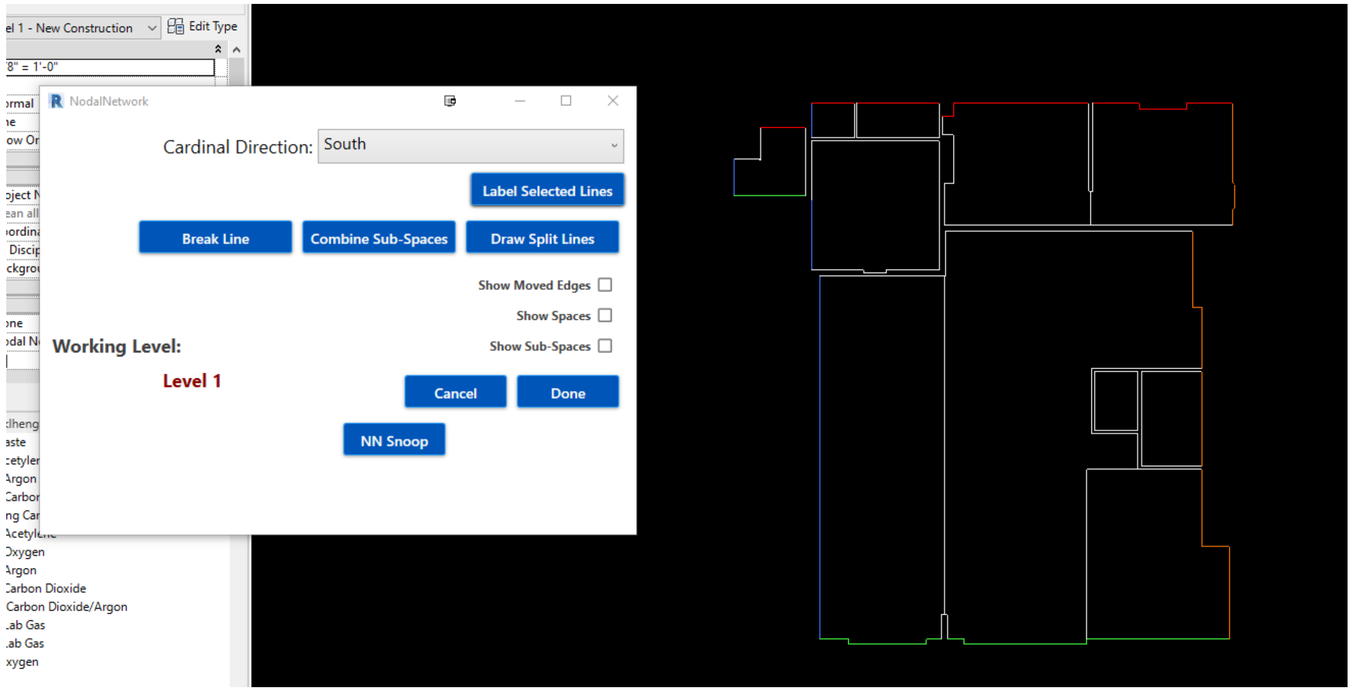
- Step 1 – Review the User Stories for the Features
- **Step 2 – Open the user interface**
- **Step 3 – Select the button to launch the application. New view should appear with outline of spaces.**



Testing Method 1: End-to-End Testing

END TO END TESTING EXAMPLE

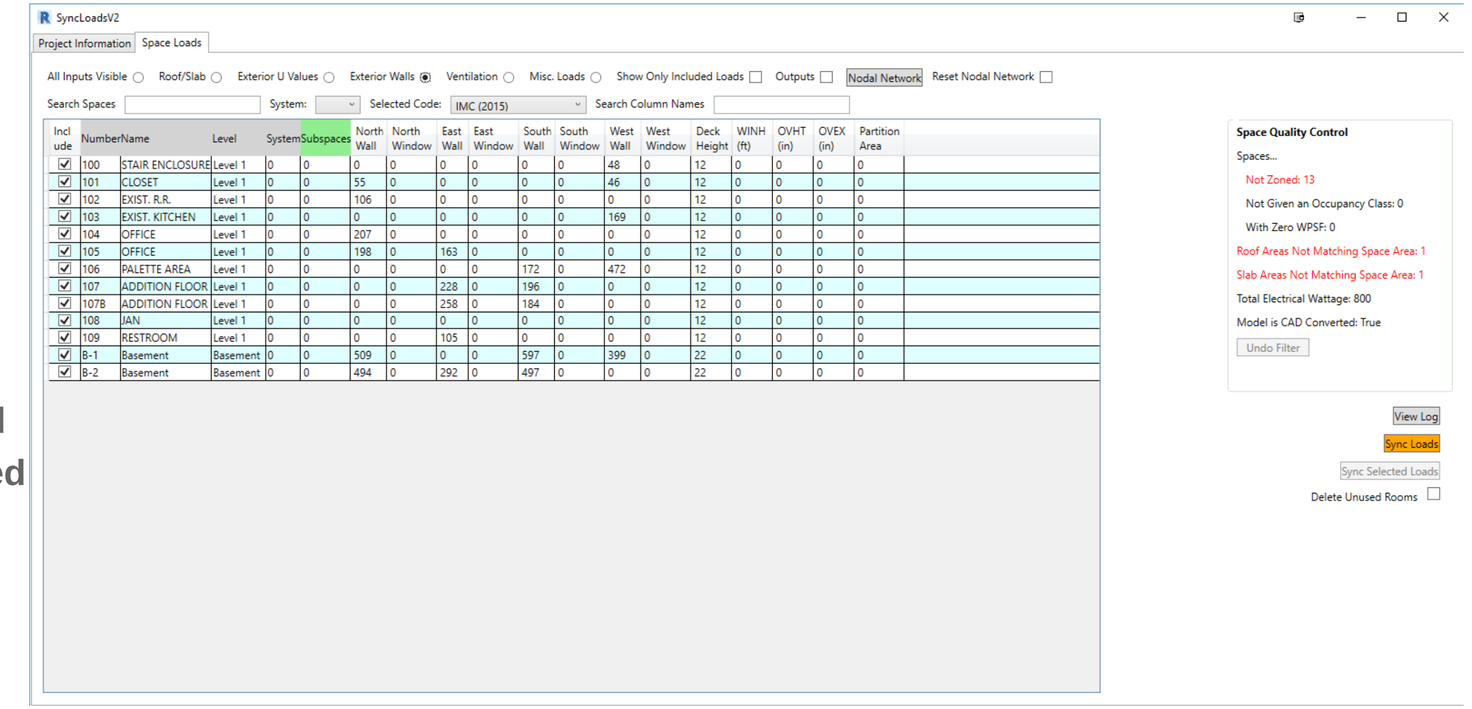
- Step 1 – Review the User Stories for the Features
- Step 2 – Open the user interface
- Step 3 – Select the button to launch the application. New view should appear with outline of spaces.
- **Step 4 – Highlight the wall and set a cardinal direction. The tool should alert the user that there has been a change.**



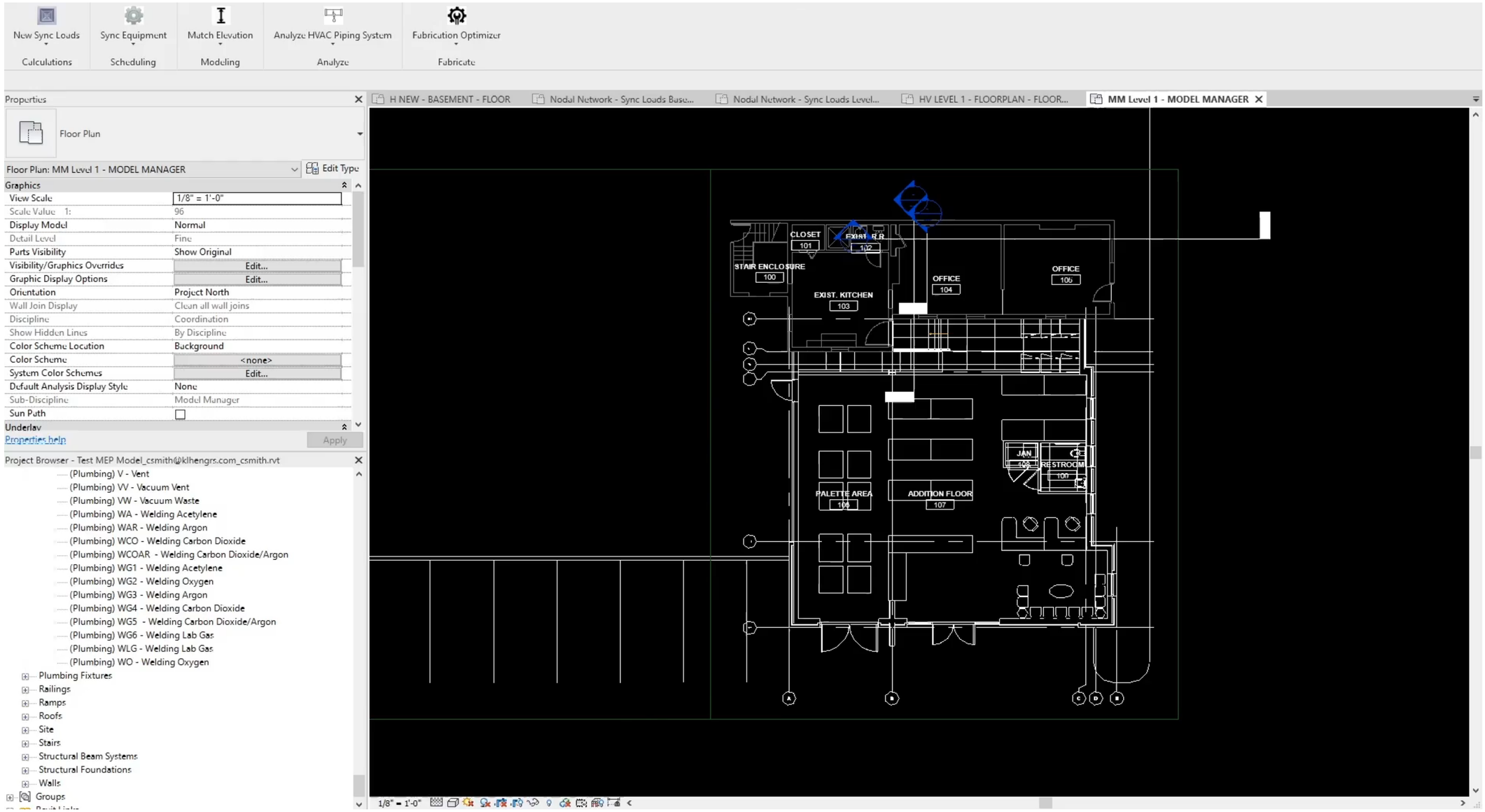
Testing Method 1: End-to-End Testing

END TO END TESTING EXAMPLE

- Step 1 – Review the User Stories for the Features
- Step 2 – Open the user interface
- Step 3 – Select the button to launch the application. New view should appear with outline of spaces.
- Step 4 – Highlight the wall and set a cardinal direction. The tool should alert the user that there has been a change.
- **Step 5 – User finishes setting wall types and executes a Finish function to push calculated data to the correct location.**



Testing Method 1: End-to-End Testing



Testing Method 1: End-to-End Testing

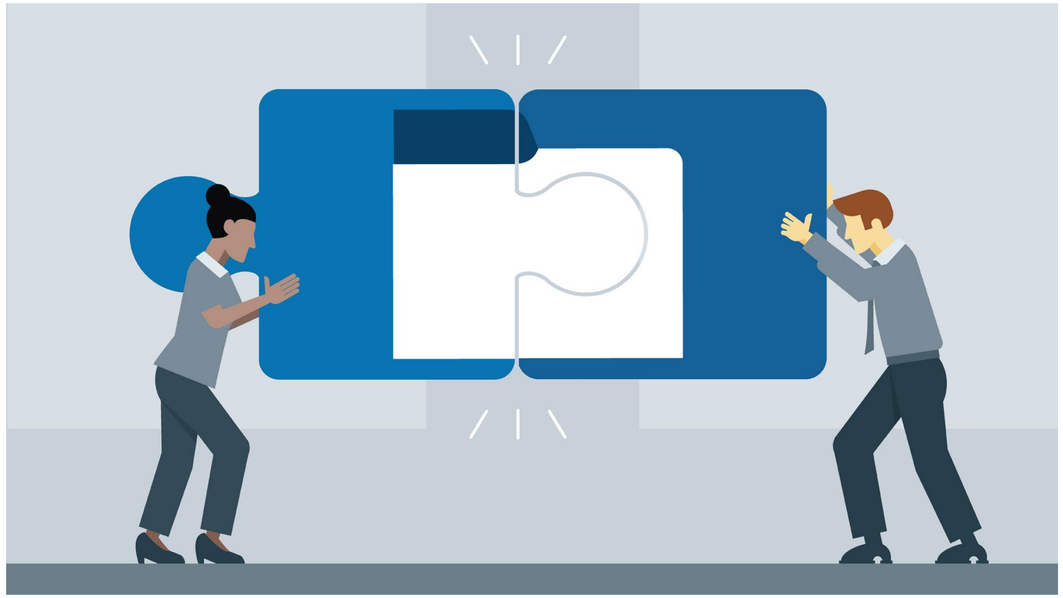
SUPERUSERS COLLABORATING WITH SOFTWARE TEAM

Superusers have the knowledge of how the tool should work as a whole.

- They understand the technology and the process
- They provide valuable feedback to Software Team.
  - Feedback comes in several forms:
    - Screen recording of tool in action and any accompanying notes.
    - Working session with the developer to discuss workflow issues.

Testing Method 2: Integration Testing

Integration testing involves taking the individual software modules and testing them as a group.



- KLH has a Revit development team along with a software team developing an ERP service.
- The Revit Team leverages the ERP teams' method and classes
- When the ERP team makes updates it’s critical to perform Integration testing to ensure the tools are working correctly before being rolled out to the clients.

Testing Method 2: Integration Testing

Structured Merged System

KLH utilizes 3 Branches with DevOps to perform Integration Testing

Development

Features logged by Software Engineer.

Testing

Software tested by QA team and superusers.

Production

Rolled out to company.

Bugs are addressed in production or testing branch and merged up.

DevOps Development Board

Testing Method 3: Compatibility Testing

- Compatibility Testing is a type of software testing to check whether your software is proficient enough to run in different environments.
- You application should be check against:
  - ***Versions***
  - ***Network***
  - Hardware
  - Operating Systems
  - Browsers
  - Mobile Devices

Testing Method 3: Compatibility Testing

Testing in all versions of Revit

- As a MEP firm the Revit model version is often dictated by the architect.
  - KLH maintains (4) version of Revit to ensure flexibility.
    - Software engineers develop in 2020. It is the responsibility of the engineers and developers to perform ***Backward Compability Testing*** to verify if the software will work with older versions of Revit.

Testing Method 3: Compatibility Testing

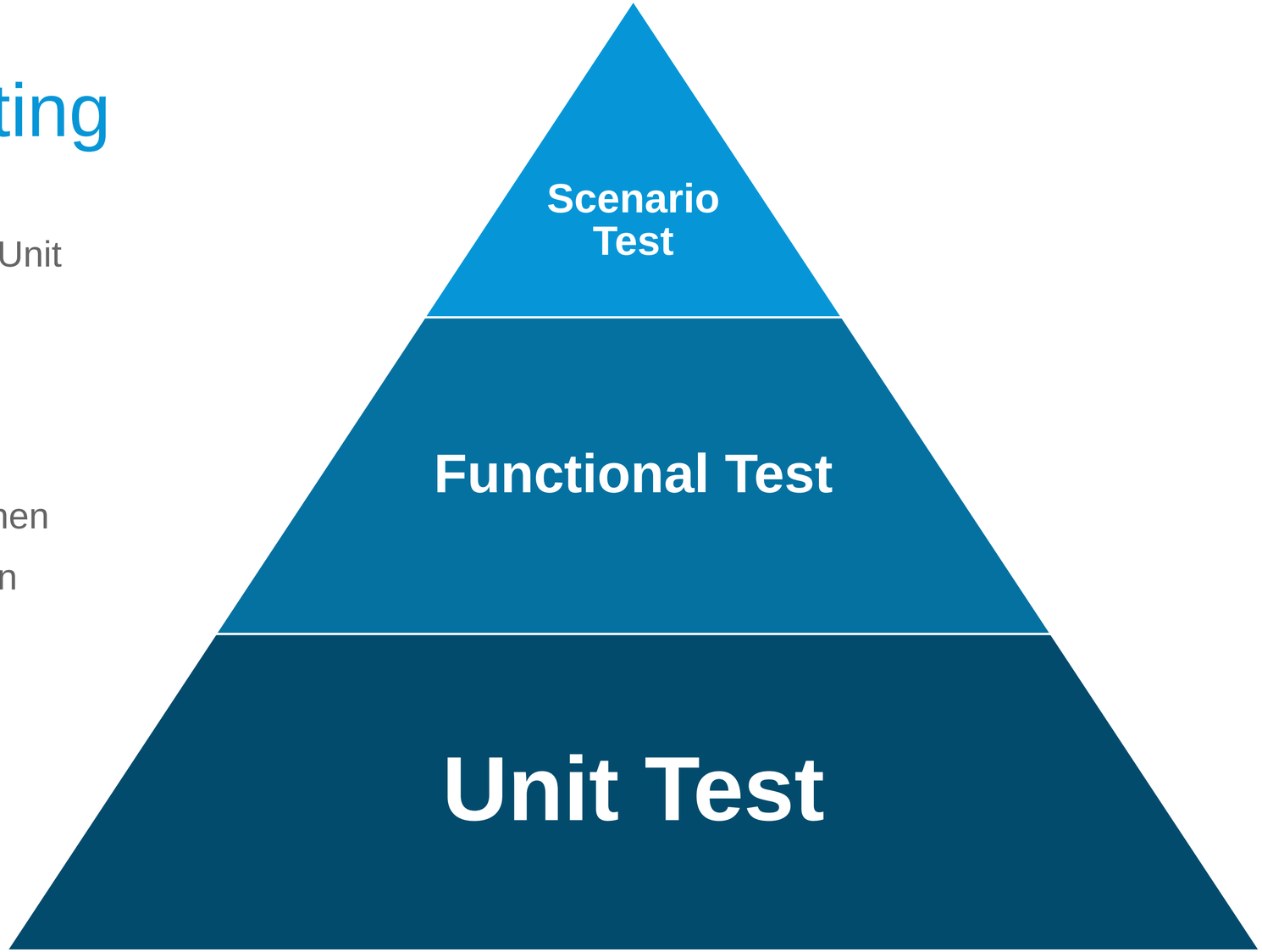
Connection Speed Tests

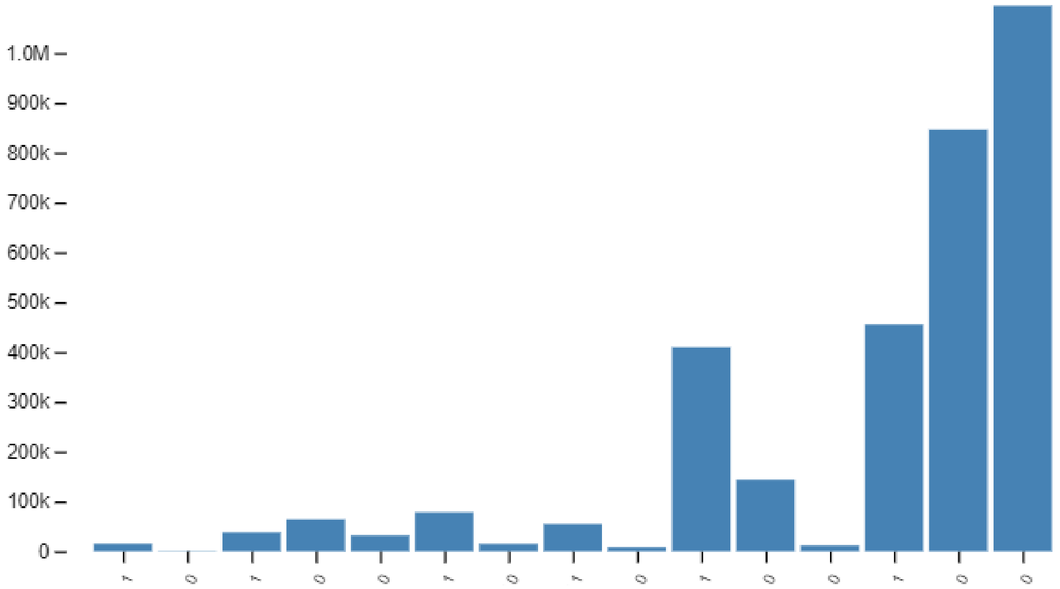
- KLH has (4) regional office all of which have different connection speeds.
  - Software Engineers perform tests that throttle connection speeds to simulate the user experience from the regional offices.
    - Software engineers explore optimization solution if load times are not acceptable.

Testing Method 4: ***Automated*** Unit Testing

- Unit testing is the foundation of testing. Unit tests are typically ran on individual units/functions of a large application.

- Unit tests are typically more effective when the person writing them is not the person who is or did write the code.





Downloads for 15 Latest Package Versions (Last 6 weeks)

NUnit

NUnit is a unit-testing framework that is compatible with all .NET languages. Nunit is Open Source software, with version 3 being under the MIT license.

Contributors of RTF

RTF

The Revit Testing Framework is a framework that should be referenced into the test project to use. It also has an executable file that facilitates running all the written tests.

Download here: https://github.com/DynamoDS/RevitTestFramework

KLH’S BATCH FILE

```
xcopy “…File Path to addin file to copy…" . /D /Y /K /R /H /C /F
xcopy “… File Path to test model to copy… " . /D /Y /K /R /H /C /F

…\RevitTestFrameworkConsole.exe --dir . -a …\KLH.Revit.Testing.dll -r results.xml -revit:"C:\Program Files\Autodesk\Revit 2019\Revit.exe" --continuous --groupByModel --clean

del ./*.log
del KLH_Ribbon.addin
del KLH2019TestModel.rvt
```

Visual Studio & Batch File

Visual studio is a standard IDE from Microsoft. The batch file allows KLH to have a standard batch file that can be copied to multiple developers to run the same tests without needing to know the syntax. **NUnit and RTF need to be marked as a reference to the test project.**

Revit Test File

Create a Revit test model to run the tests. It should be setup the same way that a user would set it up, or as close as possible. KLH uses our standard setup to ensure we are running test in the environment of most users.

Console Version Rules

If the developer types "RevitTestFrameworkConsole –h” in the command line, the below options will be populated:

--dir=[VALUE] The full path to the working directory. The working directory is the directory in which RTF will generate the journal and the addin to Run Revit. Revit's run-by-journal capability requires that all addins which need to be loaded are in the same directory as the journal file. So, if you're testing other addins on top of Revit using RTF, you'll need to put those addins in whatever directory you specify as the working directory.

-a, --assembly=[VALUE] The full path to the assembly containing your tests.

-r, --results=[VALUE] This is the full path to an .xml file that will contain the results.

-f, --fixture=[VALUE] The full name (with namespace) of a test fixture to run. If no fixture, no category and no test names are specified, RTF will run all tests in the assembly.(OPTIONAL)

-t, --testName[=VALUE] The name of a test to run. If no fixture, no category and no test names are specified, RTF will run all tests in the assembly. (OPTIONAL)

--category[=VALUE] The name of a test category to run. If no fixture, no category and no test names are specified, RTF will run all tests in the assembly. (OPTIONAL)

--exclude[=VALUE] The name of a test category to exclude. This has a higher priortiy than other settings. If a specified category is set here, any test cases that belongs to that category will not be run. (OPTIONAL)

-c, --concatenate Concatenate the results from this run of RTF with an existing results file if one exists at the path specified. The default behavior is to replace the existing results file. (OPTIONAL)

--revit[=VALUE] The Revit executable to be used for testing. If no executable is specified, RTF will use the first version of Revit that is found on the machine using the RevitAddinUtility. (OPTIONAL)

--copyAddins Specify whether to copy the addins from the Revit folder to the current working directory. Copying the addins from the Revit folder will cause the test process to simulate the typical setup on your machine. (OPTIONAL)

--dry Conduct a dry run. (OPTIONAL)

-x, --clean Cleanup journal files after test completion. (OPTIONAL)

--continuous Run all selected tests in one Revit session. (OPTIONAL)

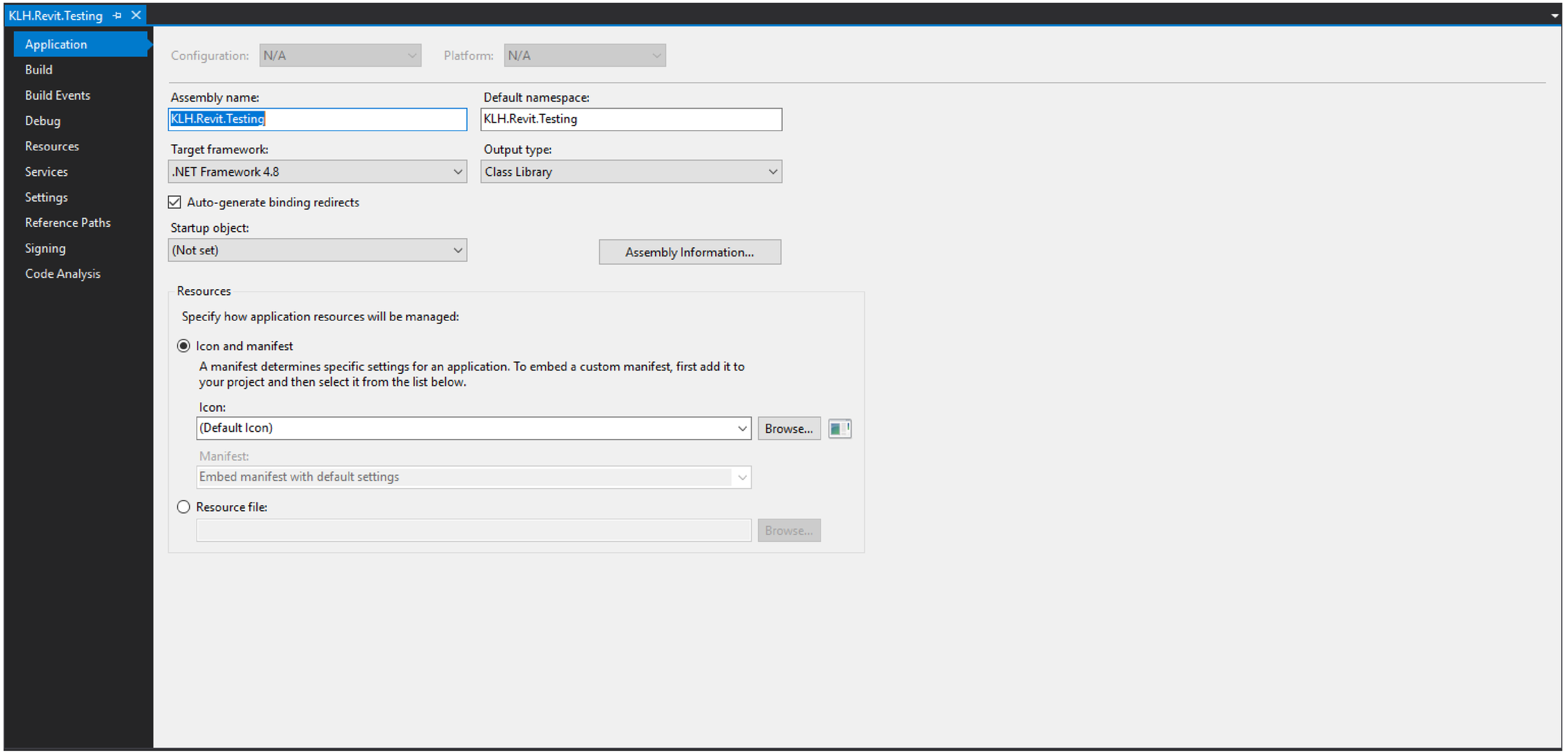
--groupByModel Run tests with same model without reopening the model for faster execution, requires --continuous. (OPTIONAL)

--time The time, in milliseconds, after which RTF will close the testing process automatically. (OPTIONAL)

-d, --debug Should RTF attempt to attach to a debugger?. (OPTIONAL)

-h, --help Show this message and exit. (OPTIONAL)

What project??



Write A Test!

```
[TestFixture]
0 references | Jacob Reiter, 20 days ago | 2 authors, 3 changes | 2 reviews
public class IsPointOnUnboundLine_Tests {

    public XYZ _point;
    public IEnumerable<XYZ> _points;
    public int _precision;

    [SetUp]
    0 references | Jacob Reiter, 73 days ago | 2 authors, 2 changes | 1 review
    public void SetupPoints()
    {
        _points = new XYZ[]
        {
            new XYZ(0,0,0),
            new XYZ(0,0,1)
        };

        _precision = 7;
    }

    [Test]
    [TestModel(Variables.TestModel)]
    0 references | Jacob Reiter, 20 days ago | 2 authors, 3 changes | 2 reviews
    public void TestOnLine()
    {
        // arrange
        _point = new XYZ(0, 0, 0.5);
        bool result;

        // act
        result = IsPointOnUnBoundLine(_point, _points, _precision);

        // assert

        Assert.IsTrue(result);
    }
```

.xml Results File


```
<?xml version="1.0" encoding="utf-8"?>
<!--This file represents the results of running a test suite-->
<test-results name = "...KLH.Revit.Testing\obj\KLH.Revit.Testing.dll" total="38" failures="0" not-run="0" date="2019-11-04" time="09:14:05" errors="0" inconclusive="0" ignored="0" skipped="0" invalid="0">
  <test-suite name="DynamoTestFrameworkTests" description="Unit tests in Revit." time="13.0229419" asserts="0" type="TestFixture" result="Success" executed="True">
    <results>
      <test-suite name="BoundingBoxOfPoints_Tests" description="Unit tests in Revit." time="0.0285923" asserts="0" type="TestFixture" result="Success" executed="True">
        <results>
          <test-case name="TestCorrectlyRounded" success="True" time="0.0285923" executed="True" asserts="0" result="Success" />
        </results>
      </test-suite>
      <test-suite name="BoundingBoxXyzContains_Tests" description="Unit tests in Revit." time="0.0026858" asserts="0" type="TestFixture" result="Success" executed="True">
        <results>
          <test-case name="CorrectlyIDPointIn" success="True" time="0.0007379" executed="True" asserts="0" result="Success" />
          <test-case name="CorrectlyIDPointOnBox" success="True" time="0.0002673" executed="True" asserts="0" result="Success" />
          <test-case name="CorrectlyIDPointOut" success="True" time="0.0016806" executed="True" asserts="0" result="Success" />
        </results>
      </results>
  </test-suite>
</test-results>
```

.xml Results File

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

<!--This file represents the results of running a test suite-->

<test-results name = “…KLH.Revit.Testing\obj\KLH.Revit.Testing.dll" total="38" failures="0" not-run="0" date="2019-11-04" time="09:14:05" errors="0" inconclusive="0" ignored="0" skipped="0" invalid="0">

This line gives a summary of all tests, 38 total test with zero fails and zero not run. It gives the date initialized and the total time ran. There were not any tests that had errors or were inconclusive, ignored, skipped or invalid.

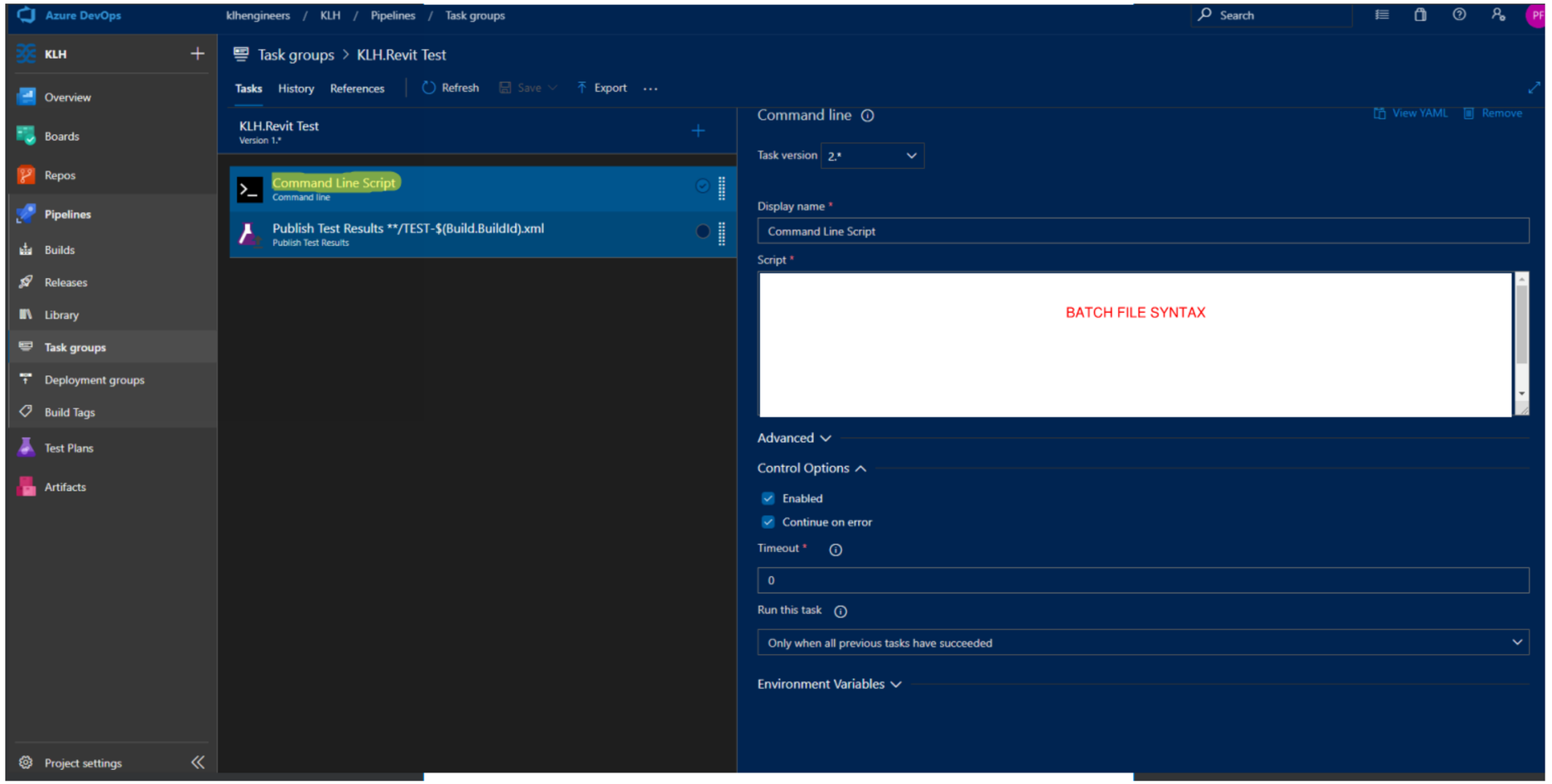
.xml Results File

```
<test-suite name="DynamoTestFrameworkTests" description="Unit tests in Revit." time="13.0229419" asserts="0" type="TestFixture" result="Success" executed="True">
    <results>
      <test-suite name="BoundingBoxOfPoints_Tests" description="Unit tests in Revit." time="0.0285923" asserts="0" type="TestFixture" result="Success" executed="True">
        <results>
          <test-case name="TestCorrectlyRounded" success="True" time="0.0285923" executed="True" asserts="0" result="Success" />
        </results>
      </test-suite>
```

The body of the xml file is broken up into all of the test suites with each test case under each test suite. Each test case shows the name, description, time, asserts, type, result, and if it was executed.

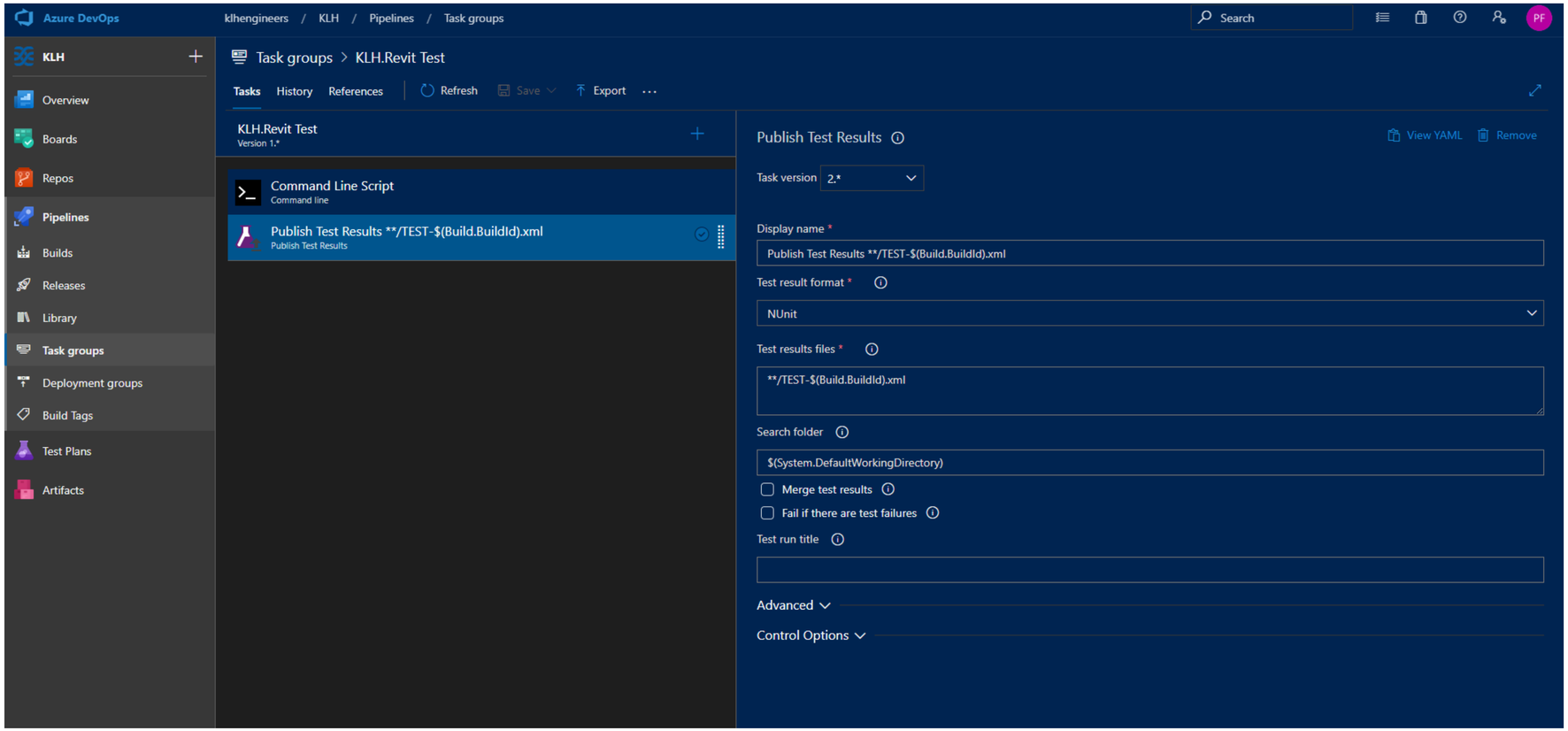
Automate Testing with Automated Builds

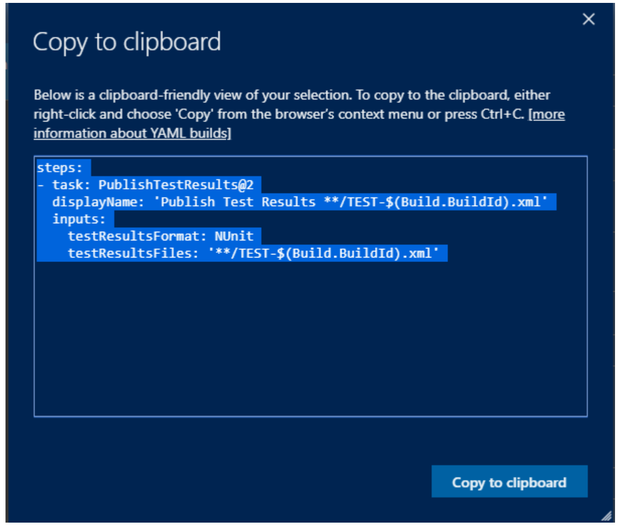
Task Group: Command Line



Automate Testing with Automated Builds

Task Group: Publish Test Results





Applying Testing Philosophies

ONE TEAM

It is challenging to send testing to multiple users, especially when the users vary. One QA team is critical to manage the tests to ensure consistency and quality of testing.

ONE LOCATION

The QA members don’t have to be in the same geographic proximity, but the logging and reporting of the tests needs to live in one spot.

ONE CULTURE

Testing needs to be embedded in the culture, especially when the same company is doing the development and the testing. There needs to be a healthy culture between the developers and the QA team, in addition to the relationship between the developers and the end users.

The Process

Feedback loop

KT **KLH Training** Follow – October 1 at 2:14 PM from Microsoft PowerApps and Flow

Feedback Form: Software Feedback Revit Electrical Tools J-R
Tool: Multi-Circuiter
Feedback Type: Bug
From: lheil@KLHENGRS.COM
Comments: Conn's 21570 - After placing all the electrical devices using the group inserter, I ran multi-circuiter using the connection parameter. There are groups that should have been circuited together, however they were all circuited as one circuit per connection.

LIKE REPLY SHARE ••• Seen by 52

#Bug #Multi Circuiter #Tools #Electrical #Revit

PF **Patrick Fernbach** – October 1 at 4:44 PM

Luke Heil, this is logged 5587. Thanks for the feedback.

cc: Luke Heil

LIKE REPLY SHARE •••

JM **Jeff Mills** – November 4 at 12:51 PM

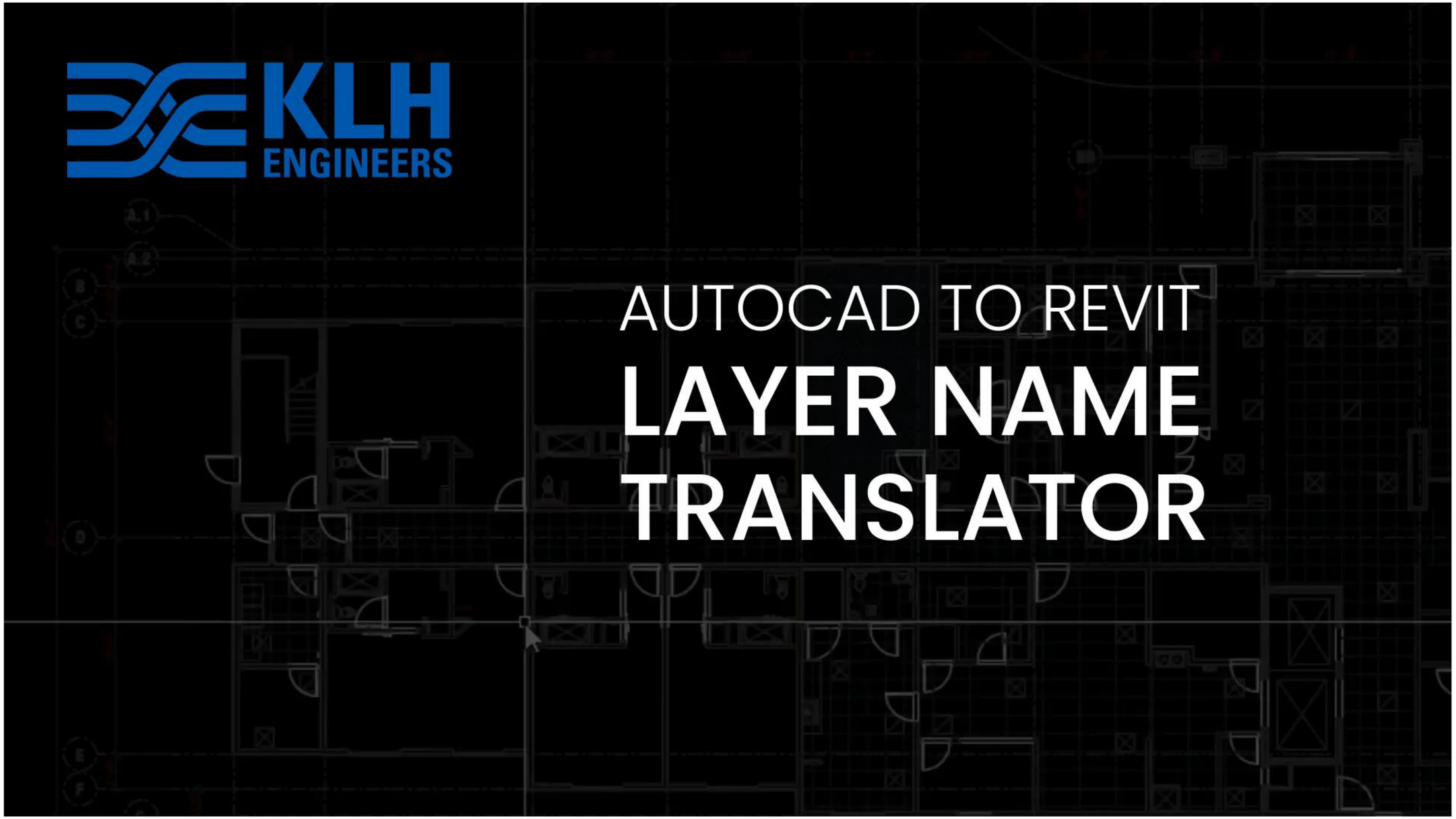
This is fixed

LIKE REPLY SHARE •••

Luke Heil likes this

Write a reply

Additional Classes taught by KLH Engineers
Thursday 8:00 – 9:00
A Practical Use of Machine Learning in the AEC Industry



AUTODESK®

Make anything™