**Release**

**Test Specification**

**Ignition 8**

**Version 1.0**

**September 8, 2022**

**ILS Automation Inc.**

Table of Contents

[1 Introduction 4](#_Toc113563476)

[2 System Preparation 4](#_Toc113563477)

[3 Full Installer Test Procedure 5](#_Toc113563478)

[3.1 Execute the Installer 5](#_Toc113563479)

[3.2 General Tests 5](#_Toc113563480)

[3.3 Production & Isolation Configuration 8](#_Toc113563481)

[3.4 Lab Data 9](#_Toc113563482)

[3.4.1 Setup 9](#_Toc113563483)

[3.4.2 Configuring Lab Data 11](#_Toc113563484)

[3.4.1 Testing Lab Data Processing 15](#_Toc113563485)

[3.5 Lab Data Feedback 17](#_Toc113563486)

[3.6 SFCs 18](#_Toc113563487)

[3.7 Symbolic Ai 25](#_Toc113563488)

[3.7.1 Setup 26](#_Toc113563489)

[3.7.2 Application / Family / Diagram / Final Diagnosis / Text Recommendation Test 27](#_Toc113563490)

[3.7.3 Application / State Management 34](#_Toc113563491)

[3.7.4 Property Editing Test 35](#_Toc113563492)

[3.7.5 Initial Value Propagation Test 35](#_Toc113563493)

[3.7.6 Context Sensitive Help 39](#_Toc113563494)

[3.8 DB Manager 40](#_Toc113563495)

[3.8.1 Verify Version 40](#_Toc113563496)

[3.8.2 Verify Client Tags 40](#_Toc113563497)

[3.8.3 Verify Isolation Mode Window 41](#_Toc113563498)

[3.8.4 Verify Isolation Mode State of Windows 41](#_Toc113563499)

[4 Update Installer Test Procedure 41](#_Toc113563500)

# Introduction

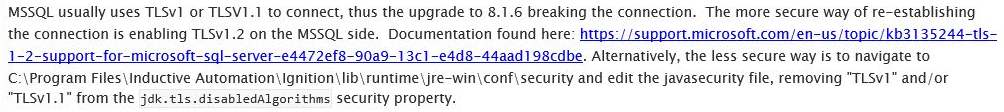
The purpose of this specification is to describe the tests to perform on the installers for a candidate release after the installers have been built but before they are delivered to the beta customer site. These tests should be performed on a test system using an Ignition image dedicated for testing this release. They cannot be performed on the same system that is used for development. These tests are not a complete test of every aspect of the system; those tests were performed as part of the QA process in the development cycle. These tests are a survey of key aspects of the system to make sure that they operate properly on an initialized test system and that the latest assets made it into the release.

There are two installers: a full installer and an update installer.

# System Preparation

If this is the first release candidate for a given release, then a new image should be created on ILSDEV4. If this is a retest of a candidate release then the image should be initialized using the clean backup. Follow these general steps:

* Using SQL\*Server, create two new, empty, databases named: XOM-Site\_XYrZ and XOM-Site\_XYrZ\_ISOLATION. If reusing a database instance, then delete all of the tables and views in the database (it might be easier to delete and create a new one)
* Create a new folder for the image in E:\Program Files\Inductive Automation on ILSDEV4
* Unzip the Ignition binary, edit the system information in data\ignition.conf
* Copy the license file, license.ipl from an existing image into the data folder
* Run Ignition’s *install-ignition.bat* as Administrator.
  + Start the service
  + Open the gateway web page and complete the install
* From the gateway web page, delete all of the unnecessary modules
* Restore a clean gateway backup from a previous install which contains the necessary database and OPC connections. It should have:
  + Database Connections:
    - XOM\_Site
    - XOM\_Site\_Isolation
    - XOMhistory
  + Tag Providers:
    - Site
    - Site\_Isolation
* Edit the database connection settings to point to the two databases created in the first step.
* Fix the java security thing described below to communicate with the database:



* Make sure that there is a user XO1TEST / XO1TEST
* Restart the gateway

# Full Installer Test Procedure

## Execute the Installer

Run the installer as follows:

1. From the gateway web page, install the full installer module from U:\Work\EMC\Installers
2. Select EMC Ignition Applications from the bottom of the Blue bar on the config tab
3. Make sure that the two checkboxes are unchecked
4. Press the “View Release Notes” button and verify that the latest release notes are in the installer.
5. Install every page of the installer.
6. Select the Home tab; select the Configure Tab; select the Modules Page; verify that the installer module has been automatically uninstalled.
7. Select the Projects page; enabled both the DbManager and XOM project. Restart the gateway

## General Tests

Perform the following general checks:

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In SQL\*Server:** |  |  |
|  | Verify that tables and views were created in the production and isolation databases. | Tables and views exist | Pass |
|  | Take a survey and verify that seed data exists is certain tables. | Data exists in: LookupType, Lookup, QueueMaster, SfcRecipeDataType, SfcStepType | Pass |
|  | Select \* from TkSite | Observe a single record with gateway and site startup scripts | Pass |
|  | **In the Gateway Web Page:** |  |  |
|  | On the Status Tab, select the Modules Page; verify that the BLT, and ILS-SFC modules are installed and running and that the module build dates are correct | Installed and running | Pass |
|  | On the Configure tag, select the Projects page, verify that the DbManager, XOM-Common, and XOM-Site projects are installed and that XOM-Common is the parent of XOM-Site. | Projects are installed | Pass |
|  | Verify the version and date in the description of each project. | Date and version match the release information | Pass |
|  | **Using the File Manager** |  |  |
|  | Verify that the *user-lib/database* folder exists and that it contains all of the database update scripts. | Database update scripts exist | Pass |
|  | Verify that the jar files were installed and have the current timestamp | Verify blt-common.jar, ils-common.jar in lib/core/common. Verify block-definition.jar, blt-gateway.jar in lib/core/gateway | Pass |
|  | **In Designer:** |  |  |
|  | Open the XOM-Site project, open the tag browser: |  |  |
|  | Verify that the configuration folder exists with subfolders for each of the toolkits | Configuration tag folder structure exists | Pass |
|  | Verify that the production UDTs exist under the production tag provider | Production UDTs exist (with embedded OPC tags) | Pass |
|  | Verify that the isolation UDTs exist under the isolation tag provider (isolation UDTs should not contain any OPC tags). | Isolation UDTs exist (with embedded memory tags in place of the OPC tags) | Pass |
|  | Verify that the scan classes exist | Scan classes exist in Isolation and production | Pass |
|  | Verify that recent icon additions are loaded. | Open Tools -> Image Management. Verify that *Custom/sfcCycle.png* exists | Pass |
|  | Open the Symbolic AI context |  |  |
|  | Verify the Symbolic Ai icon in the project tree | Observer the “brain” icon for the root Symbolic Ai node. | Pass |
|  | Select the Symbolic Ai node in the project tree, select the Connectivity tab on the Symbolic Ai palette. | Observer the palette shown in Figure 1, especially the icons for Input and Output blocks. | Pass |
|  | Verify that there is a Final Diagnosis on the Conclusion tab. | There is a Final Diagnosis on the Conclusion tab. | Pass |
|  | Open a client and log in as phassler |  |  |
|  | On the splash screen verify the version and release date information | Version and date match release information | Pass |
|  | In the View menu, verify that “Vistalon Product Mooney” has been removed | Vistalon menu does not exist | Pass |

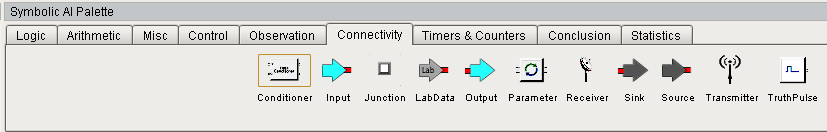


Figure 1 - Symbolic Ai Connectivity Palette

## Production & Isolation Configuration

This tests configuring the production and isolation database, tag provider, and time factor.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In the client:** |  |  |
|  | Open Admin -> Configuration -> Toolkit Configuration | The Production and Isolation Definitions window is opened. It contains two rows, one for each project: XOM\_Common and XOM\_Site. | Pass |
|  | Cells of Project column are not editable. | XOM\_Common and XOM\_Site cannot be edited. | Pass |
|  | Rows cannot be added or deleted. | Unable to add or delete rows. | Pass |
|  | Select the Production Database cell for the XOM\_Site row. | Cell is editable | Pass |
|  | Type “XOM\_Site” and press tab | XOM\_Site is added, cursor advances to Production Tag Provider cell | Pass |
|  | Type “Site” and press tab | Site is added, cursor advances to Production Time Factor cell | Pass |
|  | Type 1.0 and press tab | 1.0 is added, cursor advances to Isolation Database cell | Pass |
|  | Type “XOM\_Site\_Isolation” and press tab | XOM\_Site\_Isolation is added, cursor advances to Isolation Tag Provider cell | Pass |
|  | Type “Site\_Isolation” and press tab | Site is added, cursor advances to Isolation Time Factor cell | Pass |
|  | Type 0.1 and press Tab | 0.1 is accepted | Pass |
|  | Press the refresh button. | No change to the window. | Pass |
|  | Press the red “X” button to close the window | Window is closed | Pass |
|  | Reopen the window by selecting Admin -> Configuration -> Toolkit Configuration | Window is opened with all of the values supplied above displayed in the row for XOM\_Site. | Pass |
|  | Close ALL clients | (The structure setup during this session are read during client startup) |  |

## Lab Data

This tests the basics of Lab Data and the interaction between Lab Data, Symbolic Ai, and SQC Plotting including:

* Configuring a Lab Data Value
* Configuring SQC Limits
* Configuring a Lab Viewer
* Testing Lab Value Processing

### Setup

The following configurations should exist.

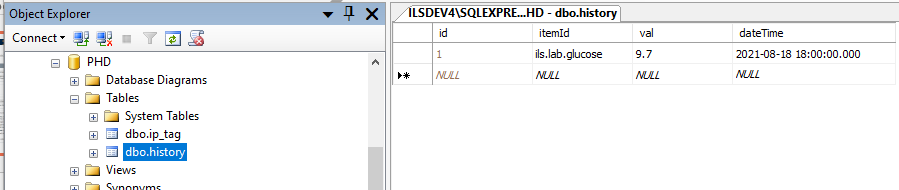
#### Gateway Configuration for Simulating HDA Data

A SQL\*Server database connection can be used to simulate a PHD HDA connection. Configure the following connection:



#### SQL\*Server HDA Simulation Configuration

The following records should exist in SQL\*Server in table history



### Configuring Lab Data

This tests the basics of Lab Data and the interaction between Lab Data, Symbolic Ai, and SQC Plotting.

Open a designer and a client.

| Item | Test | Expected Results | Compliance |
| --- | --- | --- | --- |
|  | **Configure Lab Data System Interfaces** |  |  |
|  | From the client: Select “Admin -> Lab Data -> Configure OPC and HDA Lab Data Interfaces” | “Lab Data Configuration” window is displayed and both tables are empty. | Pass |
|  | Press the “Download” button just to the right of the “OPC Interface” table. | The table is populated with “PKS-ASYNC” and “PKS-SYNC” | Pass |
|  | Select “PKS-SYNC” and press the red “Delete” button. | “PKS-SYNC” is removed from the list. | Pass |
|  | Press the “Download” button just to the right of the “HDA Interface” table. | The table is populated with “PHD-HDA” | Pass |
|  | Press the red “X” button | Window is closed | Pass |
|  | Configure Lab Data Window Initial State |  |  |
|  | From the client: Select “Admin -> Lab Data -> Configure Lab Data” | “Lab Data Configuration” window is displayed. | Pass |
|  | Select Unit “Test” | Test is selected | Pass |
|  | Click each of the 3 tabs: PHD, DCS, Local. | Each table is empty | Pass |
|  | **Create a PHD Lab Data.** |  |  |
|  | Select the PHD Tab and press the green “plus” button. | An empty PHD Lab Data window is displayed on top of the Lab Data Configuration window and cannot be covered by the Lab Data Configuration window. | Pass |
|  | Enter “GLUCOSE”, “Glucose lab value”, 4, “PHD-HDA”, and “ils.lab.glucose” into the Name, Description, Decimals, OPC HDA Server, and item Id fields and press OK | Window is dismissed and GLUCOSE is displayed in the table with all of the values entered from the window. | Pass |
|  | Lab Data UDT created | Using Designer, observe new Lab Value UDT named GLUCOSE in LabData/Test | Pass |
|  | Lab Data database record | A record was added to LtValue and LtPHDValue | Pass |
|  | Create a SQC limit |  |  |
|  | On the “PHD” tab, select “GLUCOSE” | The green “+” arrow by the Limits table at the bottom of the window becomes enabled. | Pass |
|  | Press the “+” button by the Limits table at the bottom of the window | “New Lab Limit Popup” window is displayed. “GLUCOSE” is populated in the “Lab Value” field; the combo boxes display “<Select One>”, all of the numeric fields are disabled and contain the value 0. | Pass |
|  | Select “SQC” in the Type field | Validity, SQC, Target, and Standard Deviation fields are enabled. | Pass |
|  | Select “Constant” in the Source field | “Calculate” button and text instructions become visible. | Pass |
|  | Enter 15 into the Upper SQC Limit field and 5 into the Lower SQC Limit field; press “Calculate”. | Upper Validity, Target, Standard Deviation, and Lower Validity fields are updated with: 17.5, 10, 1.67, and 2.5 | Pass |
|  | Press “Create New Limit” | New row added to the “Limits” table at the bottom of the “Lab Data Configuration” window. | Pass |
|  |  | A new UDT named GLUCOSE-SQC is created and initialized with data from the form. | Pass |
|  |  | A new record is added to the LtLimit table. | Pass |
|  | **Configure a DCS Lab Value** |  |  |
|  | Select the DCS Tab and press the green “plus” button. | An empty DCS Lab Data window is displayed on top of the Lab Data Configuration window and cannot be covered by the Lab Data Configuration window. | Pass |
|  | Enter “GLUCOSE-DCS”, “Glucose online value”, 3, “PKS-ASYNC”, “ils.lab.glucose”, 300, and 10 into the Name, Description, Decimals, OPC Server, item Id, Minimum time between samples, and Sample Time Offset fields and press OK | Window is dismissed and GLUCOSE-DCS is displayed in the table with all of the values entered from the window. | Pass |
|  | Lab Data UDTs are created | Using Designer, observe new Lab Value UDT named GLUCOSE-DCS in LabData/Test and a plain OPC tag named GLUCOSE-DCS in a new folder LabData/Test/DCS-Lab-Values | Pass |
|  | Lab Data database record | A record was added to LtValue and LtDCSValue |  |
|  | **Configure a Display Table** |  |  |
|  | Select “Admin -> Lab Data -> Configure Display Tables” from a client. | “Lab Data Display Table Configuration” window is opened and is empty. | Pass |
|  | Select Post “Test” from the post combo box. | Test is selected, the rest of the window is still empty. | Pass |
|  | Press the green “plus” button adjacent to the top table. | A modal input message box is displayed with the prompt ”Insert New Table Name:” | Pass |
|  | Enter “Nutrients” and press OK | A new row is added to the upper table with values: Nutrients, 1, False and a row is added to database table LtDisplayTable | Pass |
|  | Check the “Display” check box | Check box is checked, database row is updated. | Pass |
|  | Select the “Nutrients” Row | The green “+” arrow by the bottom table becomes enabled. | Pass |
|  | Press the bottom green “+” button. | A Popup window is displayed with a combo box of defined lab values. | Pass |
|  | Select “GLUCOSE” and press OK | “GLUCOSE” is added to the bottom table. | Pass |
|  | Press the bottom green “+” button again. | A Popup window is displayed with a combo box of defined lab values. | Pass |
|  | Select “GLUCOSE-DCS” and press OK | “GLUCOSE-DCS” is added to the bottom table. | Pass |
|  | Dismiss the “Configure Display Table” menu |  |  |
|  | Test Display Table |  |  |
|  | Select “Test Console” from the View -> Consoles Menu | Console is displayed | Pass |
|  | Press the “Lab” button in the lower left corner of the Common Console. | “Lab Data Table Chooser” window is opened. | Pass |
|  | Select the “Test” Post from the combo box | “Test” is selected and “Nutrients” is displayed in the body of the window. | Pass |
|  | Press the “Nutrients” button | A lab data table window is displayed with two tables titled: Glucose lab value” and “Glucose online value”. Both tables are empty. | Pass |

### Testing Lab Data Processing

This tests Lab Data processing using the PHD simulation capabilities described earlier.

| Item | Test | Expected Results | Compliance |
| --- | --- | --- | --- |
|  | **In Designer, set up the test environment** |  |  |
|  | Ensure that tag Configuration/LabData/pollingEnabled is True. |  |  |
|  | Ensure that tag Configuration/Common/simulateHDA is True |  |  |
|  | **In a client logged in as phassler, set up the test environment** |  |  |
|  | Open the “Test Console” console, open the lab data chooser. | An empty lab data table is displayed | Pass |
|  | Select the “Test” post from the combo box. | List of pages scroll area is updated with a single page: Nutrients | Pass |
|  | Press the “Nutrients” button | Empty Lab Data table is displayed. | Pass |
|  | **Simulate PHD Lab Data:** |  |  |
|  | In SQL\*Server, in the PHD database, simulate a valid lab value by editing the history table so that the *val* = 4.6 and the *dateTime* is roughly four hour old where the *itemId* = “ils.lab.glucose” | Within one minute the new value will be displayed at the top of the glucose table. In addition, the “Nutrients” button on the “Lab Data Table Chooser” window will turn red. | Pass |
|  | Simulate an invalid lab value by editing the history table so that the *val* = 29.8 and the *dateTime* is roughly three hour old where the *itemId* = “ils.lab.glucose” | Within one minute the new value will be displayed at the top of the glucose table. In addition, the “Nutrients” button on the “Lab Data Table Chooser” window will turn red. | Pass |
|  | Open the Lab Data Queue | Observe that the top message indicates that a value was received that failed validity limit checks but was accepted because the ‘Test’ Console was not found. | Pass |
|  | Open a second client and log in as TEST / TEST. | The Test Console will be automatically displayed. | Pass |
|  | Simulate another invalid lab value by editing the history table so that the *val* = 30.1 and the *dateTime* is roughly two hour old where the *itemId* = “ils.lab.glucose” | Within one minute the loud workspace will be displayed only on the client logged in as TEST with a message indicating that lab value failed validity testing. Press new value will be displayed at the top of the glucose table. In addition, the “Nutrients” button on the “Lab Data Table Chooser” window will turn red. | Pass |
|  | Press Acknowledge. | A detailed dialog will be displayed with details of the value that failed including the new value and the limits. | Pass |
|  | Press Accept | Detail dialog is dismissed, Lab data table UI is updated, | Pass |
|  | Open the Lab Data Queue | Observe that the top message indicates that the operator accepted the value. | Pass |
|  | **Simulate DCS Lab Data:** |  |  |
|  | Using the Matricon OPC Explorer, select ils.lab.glucose. Set its value to 18.6 | Works as expected. | Pass |
|  | In Designer, observe the new value in the OPC tag named: LabData/Test/DCS-Lab-Values/Glucose-DCS | Tag value is 18.6 | Pass |
|  |  | Within one minute the new value will be displayed at the top of the “Glucose online value” table. In addition, the “Nutrients” button on the “Lab Data Table Chooser” window will turn red. | Pass |

## Lab Data Feedback

This tests the Lab Data Feedback module.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In Designer:** |  |  |
|  |  |  |  |
|  |  |  |  |

## SFCs

Test the SFC module, database, external Python, and windows.

Open a designer and a client.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In the client:** |  |  |
|  | Open SFCs->Recipe Data Browser | An empty browser and no errors. Press all of the buttons and look for errors. | Pass |
|  | Open SFCs->SFC Recipe Data Array Keys | An empty list and no errors | Pass |
|  | Open SFCs->SFC Run Log | An empty table and no errors | Pass |
|  | Open SFCs->SFC Viewer | Observe the new SFC viewer with the power table at the top. The table and viewer is empty. | Pass |
|  | Open SFCs->SFC Runner | An empty viewer, empty combo box, and no errors | Pass |
|  | **In Designer:** create a SFC using the unit procedure operation framework with custom blocks and recipe data |  |  |
|  | Right-click on “Sequential Function Charts” and select “New Folder”. Name the folder “Test”. | Works as expected | Pass |
|  | Right-Click on the Test Folder and select “New Sequential Function Chart”, name the chart “TestUP” | Works as expected | Pass |
|  | Drag a unit procedure and an end step and connect them. Name the unit procedure “UP” | Works as expected | Pass |
|  | Create another chart named “TestOP” and drag an operation and an end step and connect them. Name the operation “OP” | Works as expected | Pass |
|  | Create another chart and name it “Tasks” | Works as expected | Pass |
|  | Drag a Yes/No step, two transitions, two action steps, and two end steps from the palettes and connect them as shown in Figure 2. | Works as expected | Pass |
|  | File -> Save, select Global Only, check the remember checkbox. | Works as expected | Pass |
|  | **In the client:** |  |  |
|  | Open the recipe data browser. | Observe the 3 charts with steps as shown in Figure 3. Note that the browser does NOT indicate a hierarchy because the unit procedure and operation were not fully configured. | Fail – The folder “Test” is shown as a chart. |
|  | **In Designer:** Configure the hierarchy. |  |  |
|  | Configure the unit procedure to call the TestOP. | Works as expected | Pass |
|  | Configure the operation to call Tasks | Works as expected | Pass |
|  | File -> Save | Works as expected | Pass |
|  | **In the client:** |  |  |
|  | Open the recipe data browser. | Observe the chart hierarchy as shown in Figure 4 | Pass |
|  | Select Test \ TestOP in the left pane and OP in the middle pane | Recipe Data “+” button lights up. | Pass |
|  | Press the “+” button | Recipe Data Type Selector popup is displayed | Pass |
|  | Select “Simple Value” and press OK | Recipe Data Editor Popup Window is displayed | Pass |
|  | Enter the key “ans” and set the datatype to “String”, press Save | ans appears in the right pane of the browser. | Pass |
|  | **In Designer:** Configure the Tasks. |  |  |
|  | Open the Tasks chart and configure the Yes/No step as shown in Figure 5 | Works as expected | Pass |
|  | Configure the transitions as shown in Figure 5 | Works as expected | Pass |
|  | Configure the action steps as shown in Figure 5 | Works as expected | Pass |
|  | File -> Save | Works as expected | Pass |
|  | **In SQLServer:** |  |  |
|  | Add “Test/TestUP” to the SfcNames table | Works as expected | Pass |
|  | **In the client:** Run the chart |  |  |
|  | Open Admin->SFCs->SFC Runner | Observe Test/TestUP in the combo list | Pass |
|  | Select Test/TestUP and press “Open Control Panel” button | Observe “Test” control panel in the lower left | Pass |
|  | Press the “V” button on the control panel. | Observe the enhanced SFC Viewer w/ the power table at the top. | Pass |
|  | Press “Start” on the control panel. | Observe popup window as shown in Figure 6 | Pass |
|  | Press “Yes” | In the SFC Viewer, select the Test/Tasks chart and observe that the step “Yes” executed as shown in Figure 7 | Pass |
|  | Press “Reset” on the control panel | “Start” button is enabled | Pass |
|  | Press “Start” on the control panel | Observe popup window as shown in Figure 6 | Pass |
|  | Press “No” | In the SFC Viewer, select the Test/Tasks chart and observe that the step “No” executed as shown in Figure 8 | Pass |
|  | Open Admin->SFCs->SFC Run Log | Observe two entries for each of the unit procedure and operation executions. | Pass |
|  | Test Context sensitive help |  |  |
|  | In Designer, open chart Test/Tasks |  |  |
|  | Right-click on the Yes/No step and select “Help” | Observe that your default web browser is opened (chrome) and displays the section of the User’s Guide for the Yes/No Step | Pass |



Figure 2 - Test Chart Hierarchy



Figure 3 - SFC Hierarchy in Recipe Browser



Figure 4 - Updated Chart Hierarchy



Figure 5 - Task Configuration

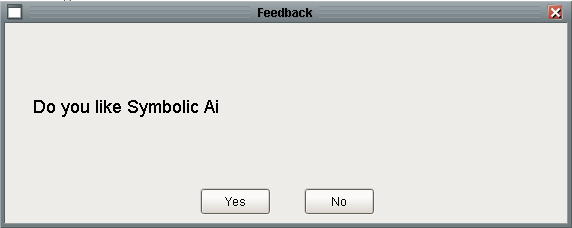


Figure 6 - Yes/No Popup



Figure 7 – “Yes” Execution Path



Figure 8 - "No" Execution Path

## Symbolic Ai

Test the Symbolic Ai module, database, external Python, and windows.

### Setup

The following figure shows the tags used in this and subsequent tests:



Figure 9 - Tags used to test Symbolic Ai

### Application / Family / Diagram / Final Diagnosis / Text Recommendation Test

This tests the entire process of creating an application, family, diagram, and final diagnosis including triggering the final diagnosis to become True and making a text recommendation and posting an Operator Alert.

Open a new designer and a new client.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In the Designer:** |  |  |
|  | In the tag browser, create a folder DiagToolkit and inside of it create two PKS controllers: FC100 and FC101 and configure them as shown in Figure 9. |  |  |
|  | **In the Client:** |  |  |
|  | Open the Application Hierarchy Configuration window by selecting “Application Hierarchy” under the “Symbolic Ai” menu. | Window is opened. Both trees contain the single Symbolic Ai root node. | Pass |
|  | On the application hierarchy tree, select the Symbolic Ai node, then right-click on it and select “Add Application” | A blank “Application Editor” window is opened. | Pass |
|  | Verify Combo boxes | Observe that the Queue, Ramp Method, and Unit combo boxes all have choices and nothing has been selected. | Pass |
|  | Enter an application name “App1”, description, check the Managed checkbox, select “XOTest”, “Longest”, and “Test” from the three combo boxes | Works as expected | Pass |
|  | Press the Outputs button | Observe an empty list of outputs | Pass |
|  | Press the “Add Row” button which is the only enabled button | An empty Quant Output Editor window slides on top | Pass |
|  | Press the “Search” button to the right of the Tag field. | The tag browser window slides on top | Pass |
|  | Select DiagnosticToolkit/ TESTAPP1/ Outputs/ FC100 and press the green back arrow | Observe that the tagpath is updated into the Tag Field. | Pass |
|  | Enter FC100 into the name field, press tab | Works as expected | Pass |
|  | Select Most Positive from the feedback method combo box, unselect the Incremental Output checkbox, Enter 20, -20, 0.02, 200, 0 into the fields. Press the green back button. | The Outputs pane slides back on top with FC100 in the list. | Pass |
|  | With FC100 selected press the “Edit” button. | Output Editor window slides on top with all of the information entered previously | Pass |
|  | Press the green back button | The Outputs pane slides back on top with FC100 in the list. | Pass |
|  | Press the green “+” button | Observe an empty Quant Output Editor window slides on top | Pass |
|  | Press the green Tags button | Observe that the tag browser window slides on top | Pass |
|  | Select DiagnosticToolkit/ TESTAPP1/ Outputs/ FC101 and press the green back arrow | Observe that the tagpath is updated into the Tag Field. | Pass |
|  | Enter FC101 into the name field, press tab | Works as expected | Pass |
|  | Use the default values and press the green back button. | The Outputs pane slides back on top with FC100 and FC101 in the list. | Pass |
|  | Press the green back button | The home page for the application editor slides back on top. | Pass |
|  | Press “Save” | Windows is dismissed, the “Application Hierarchy” tree in the “Application Hierarchy Window” is updated to reflect the application “App1” | Pass |
|  | Select “App1”, then right-click on the “App1” node and select “Add Family” | A blank “Family Editor” window is opened | Pass |
|  | Enter “Fam1” as the family name, enter a description and change the priority of the family to 5.6 | Works as expected | Pass |
|  | Press Save. | Application editor is dismissed. A triangle appears to the left of the application in the tree. | Pass |
|  | Press the triangle to the left of the application node to expand the children | Observe a family node labelled “Fam1” | Pass |
|  | **In SQLServer:** |  |  |
|  | View rows in DtApplication | Table contains a single row with values consistent with entered data. | Pass |
|  | View rows in DtFamily | Table contains a single row with values consistent with entered data. | Pass |
|  | View rows in DtQuantOutput | Table contains a two rows with values consistent with entered data. | Pass |
|  | **In the Designer:** |  |  |
|  | Right click on the “Symbolic Ai” node and select “Create Folder” | A “New Folder” popup is displayed with suggested value “New Folder” | Pass |
|  | Enter “App1” and press return | Popup is dismissed and a folder is created under the Symbolic Ai node. | Pass |
|  | Right-click on App1 and select “Create Diagram” | A “New Diagram” popup is displayed with suggested value “New Diagram” | Pass |
|  | Enter “Diagram1” and press “Create Diagram” | Popup is dismissed and a new diagram is added as a child of the App1 folder. Both the folder and diagram’s names are in italics. | Pass |
|  | File -> Save | Folder and diagram names are shown ion normal font. | Pass |
|  | Open Diagram1 | An empty canvas is displayed. | Pass |
|  | Drag a high limit observation block from the Observation palette onto the canvas. | Block is on canvas, background becomes dirty, Diagram1 is in italics | Pass |
|  | Using the property editor, change the limit to 10. | Works as expected | Pass |
|  | Drag a low limit observation block from the Observation palette onto the canvas. | Works as expected | Pass |
|  | Using the property editor, change the limit to -10.0. | Works as expected | Pass |
|  | From the tag browser, select FC100.value and drag it onto the left side of the canvas. | An input block is created, named FC100-value and is configured with FC100. | Pass |
|  |  | A property display box is automatically created below the input displaying “FC100-value” | Pass |
|  | Connect the output of the input block to the input of the high limit observation block and low limit blocks. Press Save | Background becomes clean, diagram name in normal font. | Pass |
|  |  | Connections change from dashed to solid grey |  |
|  | Drag an OR block from the Logic tab onto the canvas | Background becomes dirty | Pass |
|  | Connect the outputs of the two observation blocks to the input of the OR block. | Works as expected | Pass |
|  | Drag a Final Diagnosis from the Conclusion tab to the canvas. | Works as expected | Pass |
|  | Drag the output of the OR block to the input of the final diagnosis. Select File -> Save. | Works as expected | Pass |
|  | Select the Final Diagnosis | Property editor is displayed with just an “Activity Buffer Size” property in addition to the required 3 (Name, Class, UUID) | Pass |
|  | Change the name of the final diagnosis to FD1, select File -> Save. | Works as expected | Pass |
|  | **In SQLServer:** |  |  |
|  | View rows in DtFinalDiagnosis | Table contains a single row for FD1 with most fields NULL or the default. | Pass |
|  | View rows in DtRecommendationDefinition | Table is empty. | Pass |
|  | **In the Client:** |  |  |
|  | Open the Test Console | Works as expected | Pass |
|  | Select “Symbolic Ai -> Application Hierarchy | The Diagram tree displays the App1 folder. | Pass |
|  | Expand the App1 folder | Diagram1 is displayed as a child of App1 | Pass |
|  | Select Diagram1 | Left and right arrows are grey and disabled | Pass |
|  | In the Hierarchy Tree, expand App1 | Family “Fam1” is displayed as a child of App1. | Pass |
|  | Select Fam1 | Right arrow becomes green | Pass |
|  | Press the right Arrow | Diagram1 icon in the Diagram Tree becomes grey | Pass |
|  |  | Diagram1 is shown as a child of Fam1 | Pass |
|  | Expand Diagram1 | Final Diagnosis FD1 is shown as a child of Diagram1 | Pass |
|  | Press the two-state button labelled “All Diagrams” | Button text changes to “Unreferenced Diagrams” | Pass |
|  |  | Diagram1 is removed from the tree | Pass |
|  | Select FD1 in the hierarchy tree | Edit button is enabled | Pass |
|  | Press the Edit button | Final Diagnosis editor is displayed | Pass |
|  |  | FC100 and FC101 are displayed in the list of available Outputs | Pass |
|  | Configure the Final Diagnosis as shown in Figure 11. Press Save. | Works as expected. |  |
|  | **In the Designer:** |  |  |
|  | Using the tag browser, set the value of tag FC100.value to 2.3 | Observe the outputs of both observations and the OR block turn red | Pass |
|  | Set the value of tag FC100.value to 15.6 | Observe the output of the High Limit observation turn Green, the output of the Low observation remains Red, the output of the OR block becomes Green. | Pass |
|  | **In the Client:** |  |  |
|  | Wait for the OC alert | Observe the loud workspace indicating that a new text recommendation has arrived. | Pass |
|  | Press Acknowledge | Observe a modal Text Recommendation with the text: “FC100 is out of Spec” | Pass |
|  | Press OK on the modal dialog | Text recommendation is closed | Pass |
|  | **In the Designer:** |  |  |
|  | Diagram and Final diagnosis are reset. | Observe that Diagram1 has been reset: all 3 of the Boolean connections are grey, view the internal state of both observations, the or block and the final diagnosis. The port: out of all 4 should be UNKNOWN. | FAIL. |
|  | Close the designer and client | Works as expected. | Pass |

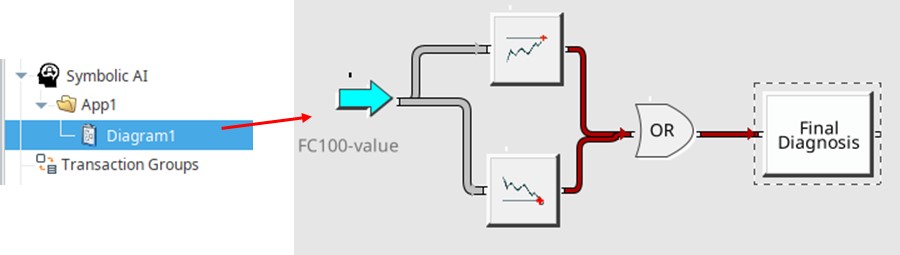


Figure 10 - Hierarchy and diagram for the first test

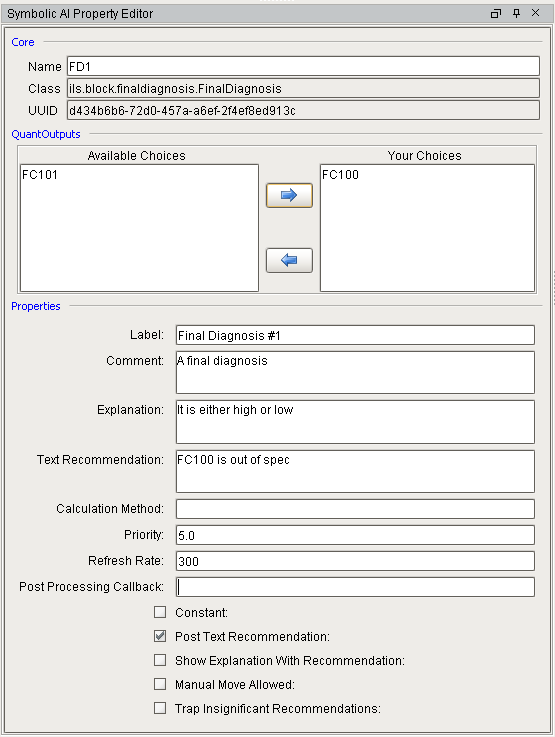


Figure 11 - Final Diagnosis Properties

### Application / State Management

A troublesome issue has been setting the state of diagrams from the project tree in Designer. This was a significant bug that was detected in 2.2r1 and fixed in 2.2r2. These tests use the diagram hierarchy created in the previous section.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **Test Setting the state from a diagram** |  |  |
|  | In the Gateway webpage: Open the Status -> Symbolic Ai page | Currently defined diagram list includes App1/Diagram1 and indicates that its state is ACTIVE | Pass |
|  | In Designer: Under Symbolic Ai in the project tree, expand the Symbolic Ai tree. | Works as expected. The diagram icon in the tree for Diagram1 is dark blue indicating that the diagram is active and is not open | Pass |
|  | Open diagram Diagram1. | The diagram is opened and the icon in the tree becomes white with a dark blue background indicating that the diagram is active and is open. | Pass |
|  | Right-click on diagram Diagram1 in the project tree and select **Set State -> Disabled**. (Do NOT select File->Save) | Diagram became dirty as reflected by the diagram name becoming italicized. A brown badge is added to the icon in the project tree. Diagram background is dirty. | Pass |
|  | In the Gateway webpage: refresh the Status -> Symbolic Ai page. | Diagram state is still Active | Pass |
|  | In Designer, select File -> Save | Works as expected, italicized name returns to normal font, background becomes dark grey. | Pass |
|  | In the Gateway webpage, refresh the Status -> Symbolic Ai page. | Diagram state is DISABLED | Pass |
|  | **Test Setting the state from a folder** |  |  |
|  | In Designer: Right-click on folder App1 and select **Set Diagram States in Hierarchy -> Active**. (Do NOT select File->Save). | Diagram1 becomes dirty as reflected by the diagram name in italics and dirty background. The brown badge is removed from the project tree icon. | Pass |
|  | In the Gateway webpage: refresh the Status -> Symbolic Ai page. | Diagram state is still DISABLED | Pass |
|  | In Designer, select File -> Save | Works as expected, italicized name returns to normal font, background becomes light-grey. | Pass |
|  | In the Gateway webpage, refresh the Status -> Symbolic Ai page. | Diagram state is ACTIVE | Pass |

### Property Editing Test

A troublesome issue has been reliable saving of block properties. These tests use the diagram created in the previous section.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In the Designer:** |  |  |
|  | Under Symbolic Ai in the project tree, open the diagram App1.Diagram1 | Diagram looks the same as left above. | Pass |
|  | Select the top High Limit Observation and change the Limit to 10.5. Tab to another field. | Diagram became dirty | Pass |
|  | Press File -> Save | Diagram is reset | Pass |
|  | Close the diagram, open the diagram. | Works as expected | Pass |
|  | Select the top High Limit Observation | The Limit is 10.5 | Pass |
|  | Close the Designer, restart the gateway, open Designer, and select the same diagram. |  |  |
|  | Select the top High Limit Observation | The Limit is 10.5 | Pass |

### Initial Value Propagation Test

A recurring issue that is of vital importance is the propagation of initial values into diagrams. This is especially important of inputs to diagrams that are tags whose values do not change frequently such as setpoints or equipment configuration OPC tags. This tests building two nearly identical diagrams. The difference between them is that the first is built left to right and the second is built right to left. After the diagrams are built and data values set, the gateway is restarted. The state of the diagrams after the restart should be identical to the state before the restart.

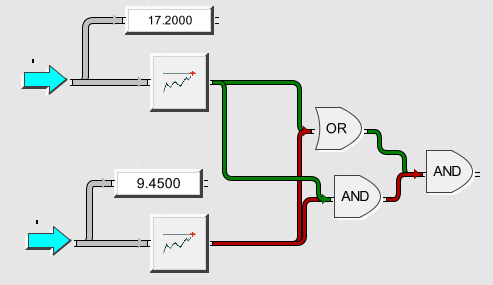


Figure 12 - Initial Value Propagation Test Diagram

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In Designer:** |  |  |
|  | In the tag browser, set DiagToolkit/T1 to 9.45; set DiagToolkit/FC101.value to 17.2 | Works as expected | Pass |
|  | Right-Click on the “Symbolic Ai” node in the project tree. Select “Create Folder”, name the folder “Startup” | Works as expected | Pass |
|  | Right-Click on the “Startup” folder created above. Select “Create Diagram”. Name the new diagram “Startup 1” | Works as expected | Pass |
|  | Drag the FC101.value onto the upper left of the diagram | An input block is created | Pass |
|  | Drag T1 onto the lower-left of the diagram. | An input block is created | Pass |
|  | Drag a high limit block from the observation palette across from the top input. Configure the limit to 10.0. | Observation block is created | Pass |
|  | Drag a high limit block from the observation palette across from the bottom input. Configure the limit to 10.0. | Observation block is created | Pass |
|  | Connect the output of the top input to the input of the top high limit. | Connection is made | Pass |
|  | Connect the output of the bottom input to the input of the bottom high limit. | Connection is made | Pass |
|  | Drag two readouts onto the diagram, one above each high limit connection back to each of the inputs | Readouts are created. | Pass |
|  | Drag an OR block onto the diagram to the right of the top high limit. Connect the outputs of each high limit to the input of the OR. | Block and connections are made. | Pass |
|  | Drag an AND block onto the diagram to the right of the bottom high limit, directly below the OR block created above. Connect the outputs of each high limit to the input of the AND. | Block and connections are made. | Pass |
|  | Drag an AND block onto the diagram to the right of two blocks created above. Connect the outputs of OR and the AND to the input of the new AND. | Block and connections are made. | Pass |
|  | Select File -> Save | Works as expected | Pass |
|  | Close the diagram and reopen the diagram. | Works as expected | Pass |
|  | Close Designer, reopen Designer, open the diagram. (This tests a critical bug introduced in 2.2r2) | Diagram is opened and looks as it did when saved. | Pass |
|  | Set the value of FC101.value to 17.2 and T1 to 9.45. | The diagram should look like Figure 12 | Pass |
|  | Create an identical diagram in the opposite order |  |  |
|  | Right-Click on the “Startup” folder created above. Select “Create Diagram”. Name the new diagram “Startup 2” | Works as expected | Pass |
|  | Drag an AND block onto the diagram on the far right in the middle. | Block is made. | Pass |
|  | Drag an AND block onto the diagram to the left and below the AND block created above. Connect the output of this AND to the input of the AND created above. | Block and connection is made. | Pass |
|  | Drag an OR block onto the diagram directly above the AND block created in the previous step. Connect the output of the OR to the input of the far right AND. | Block and connection is made. | Pass |
|  | Drag two readouts onto the diagram, to the left of the logic blocks, one near the top and the other in the middle. | Readouts are created. | Pass |
|  | Drag two high limit blocks from the observation palette onto the diagram directly beneath each of the readouts. Configure the limit of both to 10.0. | Observation block is created | Pass |
|  | Connect the outputs of each high limit observation to the inputs of the AND and OR blocks, making 4 connections. | Connections are drawn | Pass |
|  | Drag T1 onto the lower-left of the diagram. | An input block is created | Pass |
|  | Connect the output of the input block to the input of the lower High Limit block and the input of the lower readout | Connections are made | Pass |
|  | Drag the FC101.value onto the upper left of the diagram. | An input block is created | Pass |
|  | Connect the output of the input block to the input of the top High Limit block and the input of the top readout | Connections are made | Pass |
|  | Select File -> Save | Works as expected | Pass |
|  | Close the diagram and reopen the diagram. | Works as expected | Pass |
|  | Set the value of FC100.value to 0.0 and T1 to 0.0. | All Boolean connections should be red on both Startup 1 and startup 2 | Pass |
|  | Set the value of FC100.value to 17.2 and T1 to 9.45. | The diagram should look like Figure 12 | Pass |
|  | Select File -> Save, close Designer | Works as expected | Pass |
|  | Restart the gateway, open designer, open both diagrams. | Both diagrams should look like Figure 12 | Pass |

### Context Sensitive Help

This verifies context sensitive help on a Symbolic Ai diagram.

| # | Test | Desired Observation | Result |
| --- | --- | --- | --- |
|  | **In Designer:** |  |  |
|  | Open Diagram1 in App1 / Diagram1 |  |  |
|  | Right-click on the OR block and select Help | User manual is opened in browser and goes directly to the OR block section | Pass |
|  | Right-click on the Final Diagnosis block and select Help | User manual is opened in browser and goes directly to the Final Diagnosis block section | Pass |
|  | Right-click on the High Limit Observation block and select Help | User manual is opened in browser and goes directly to the High Limit Observation section | Pass |
|  | Right-click on the Low Limit Observation block and select Help | User manual is opened in browser and goes directly to the Low Limit Observation section | Pass |

## DB Manager

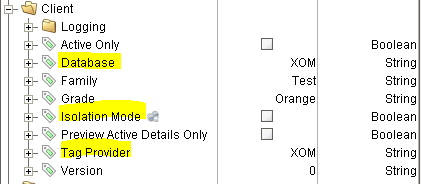
This requires that the DB Manager project was installed with the installer. Open the DB Manager project in designer and open a client.

### Verify Version

In the client, verify the version in the Help -> About This Application window.

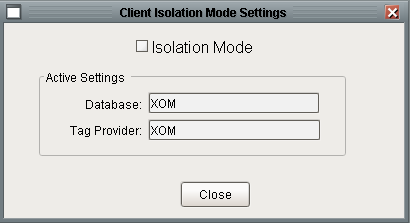
### Verify Client Tags

In designer, verify the client tags that support isolation mode.



### Verify Isolation Mode Window

In the client, verify that the “Tools -> Isolation Mode Settings” choice is present and selecting it displays the following window:



### Verify Isolation Mode State of Windows

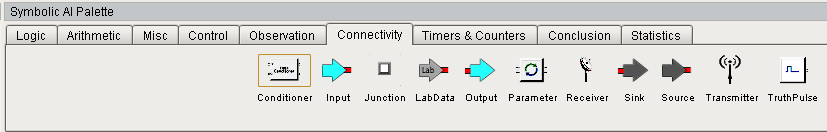
In the client, open the Isolation Mode window discussed above and check the Isolation Mode check box. Open all of the windows under the “View” menu and observe an orange background.

# Update Installer Test Procedure

This test verifies the update installer. It focusses on the process of delivering the release, not on specific capabilities of the software since individual features were tested in the previous section.

Use ILSDEV4 for this testing

Perform the following steps:

1. Start the Ignition instance for the previous version
   1. Make a gateway backup
   2. Shut down Ignition
2. In SQL\*Server:
   1. Restore the XOM\_TEST database from database of the previous version.
3. Under the E:\Program Files\Inductive Automation\Ignition7.9.12\_BLT\_Test:
   1. Delete the ILS Python, XOM Python, Database, and Projects folders under user-lib.
   2. Delete ILS libraries under lib/core.
4. Start the “Ignition Gateway 7.9.12 BLT Test” instance.
   1. Restore the gateway created in step #1.
   2. Modify the DB connection to point to XOM\_TEST
   3. Restart the gateway
5. Using SQL\*Server:
   1. inspect the DtApplication, DtFamily, DtFinalDiagnosis, DtQuantOutput, and DtQuantRecDef and make sure that each table has data.
6. Using Designer:
   1. Under Windows, delete the *Lab Data* folder and all of the windows in it.
7. Using the file manager:
   1. Delete a couple of modules under ILS Python
8. From the Gateway Webpage:
   1. Install and run the update installer.
   2. Restart the gateway.
   3. Observe that the installer is uninstalled
9. In Designer:
   1. Open the XOM project and then import the *XOM\_Update* project that was copied to the userlib/projects folder. Select overwrite for every duplicate resource.
   2. File -> Save; Exit Designer
10. Restart the gateway.
11. In the gateway web page:
    1. On the Configure / Modules page, verify that the modules have the correct build date and are running.
12. In the designer:
    1. Verify that the installer restored the deleted *windows/Lab Data*.
    2. The version window is updated.
    3. Open the Symbolic Ai folder
       1. Inspect the Symbolic Ai tree in the project tree and observe the application / family / diagram hierarchy.
       2. Open the Property Editor for several applications and make sure that they are fully configured including Quant Outputs
       3. Open the property editor for several families and ensure that they are fully configured
       4. Open the property editor for several Final Diagnosis and ensure that they are fully configured.
13. Using the File Manager:
    1. Verify that the ILS-Python, XOM-Python, Database, and Projects folders are restored.
14. Using the File Manager:
    1. Verify that the user-lib/database folder exists and that it contains all of the database update scripts.
15. Using Designer, verify:
    1. Open the Tag Browser:
       1. Verify that the configuration folder exists with subfolders for each of the toolkits.
       2. Verify that the production UDTs exist under the production tag provider
       3. Verify that the isolation UDTs exist under the isolation tag provider (isolation UDTs should not contain any OPC tags).
       4. Verify that the scan classes exist
    2. Open the Symbolic AI context:
       1. Observe the following palette including the following icons for Input and Output. 
       2. Verify that there is a Final Diagnosis on the Conclusion tab.
16. Open a client and verify:
    1. On the splash screen verify the version a release date information.
    2. In the View menu, verify that "Vistalon Product Mooney" has been removed.
17. Open SQL\*Server and verify:
    1. Select \* from TkSite - observe a single record with gateway and site startup scripts.
    2. Verify that the Version table has been updated.