ExxonMobil Chemical Company

Recipe Toolkit

User's Guide

Version 1.1

January 13, 2022

Prepared by:

ILS Automation Inc.

Table Of Contents

1 Introduction

The Recipe Toolkit facilitates the transfer or download of product or process specific data from a SQL database to the DCS. The toolkit is implemented using Ignition from Inductive Automation and is one of several toolkits that combine to make a full featured supervisory control system.

The majority of recipe data are process setpoints written through OPC to the DCS. The OPC item-ids of the DCS tags are stored with the recipe data in the database. The Recipe Toolkit will automatically manage the OPC tags within Ignition. The Recipe Toolkit reads the recipe from the SQL*Server database and automatically creates all of the necessary OPC tags. Local recipe tags must be created manually or by a site startup script outside the scope of the recipe toolkit.

The two main users of the system are a contact engineer who is responsible for maintaining the recipe within the database and console operators and Application Engineers (AE) that are responsible for selecting, reviewing, and downloading the recipe to the DCS.

The database design is described in the *Database Design Specification*. The user interface for the contact engineer is described in the *DB Manager User's Guide*. This document describes the design and use of the recipe toolkit as it pertains to the operator and application engineer.

2 Recipe Toolkit User Interface

This section provides a description of the operator console and user interface.

2.1 Overview

The general steps to using the recipe system are as follows.

1. Select a recipe to review values for changes before downloading the values.
2. Review the recipe, making changes if necessary.
3. Save a modified version of the recipe for later review or download, if desired.
4. Download the recipe.
5. Review any download errors.
6. Correct errors from the download.

All changes made to a recipe by an operator pertain only to the current recipe (i.e., temporary changes) unless the changes are saved to the recipe database as a new version. Refer to section 2.5.3 for details on saving a new version of a recipe.

The screens involved in this workflow are shown below. The sequence begins by pressing the "Recipe" button on the operator console.

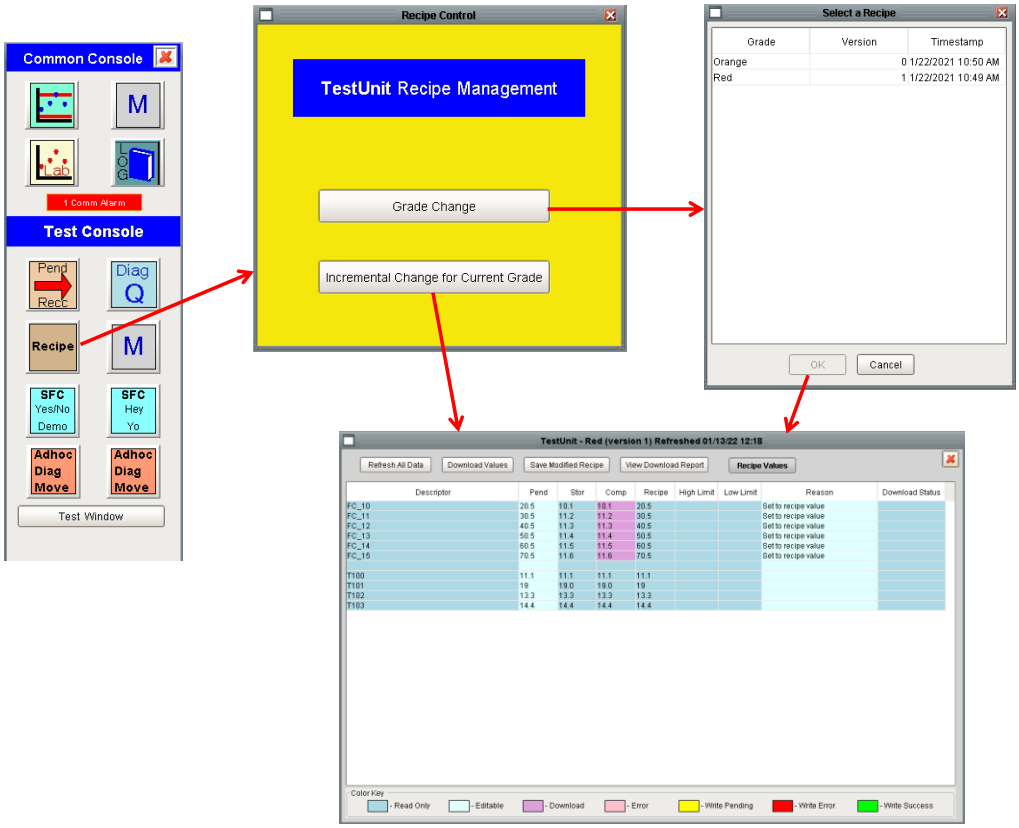

Figure 1 - Recipe Toolkit Overview

2.2 Accessing Recipes from the Operator Console

The operator can access the recipe workspace by clicking on the recipe button on the unit specific operator console window which is a site specific window. The recipe button is configured as shown below to open the generic recipe control window (the yellow window). The recipe button is configured as shown with the recipe family and the generic recipe control window (the yellow window).

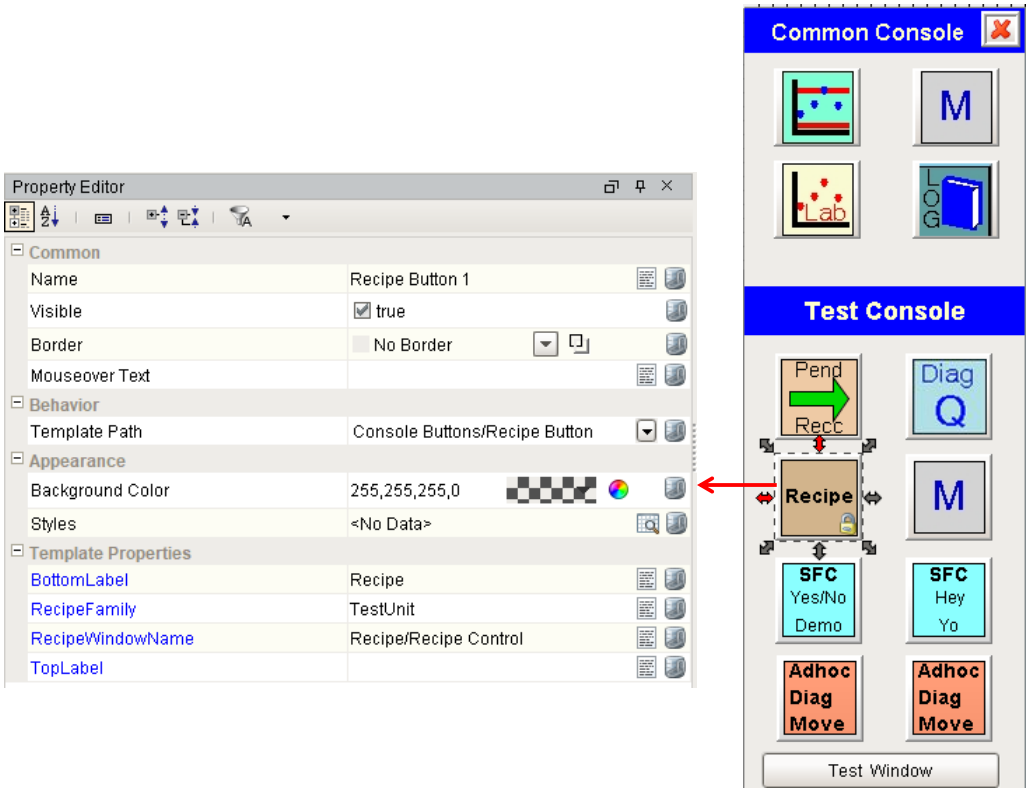

Figure 2 - Operator Console and Recipe Button

2.3 Recipe Control Window

The Recipe Control window allows the operator to choose whether to perform a grade change or make an incremental process change. The window shown below is generic. It expects that the recipe family is passed to the window and is then shown in the title and passed by both of the buttons.

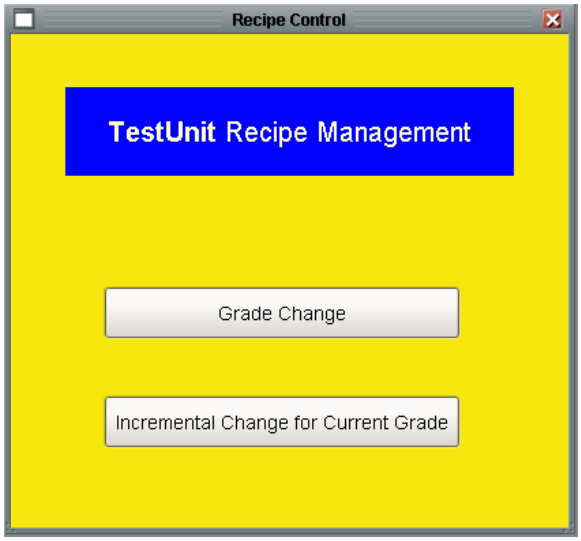

Figure 3 - Recipe Select / Display Screen

2.4 Recipe Type & Grade Selection Screens

Operators can choose a recipe from a list of active recipes. The system does not restrict the number of versions of a recipe, however, the operator can only download versions that the contact engineer has designated as active.

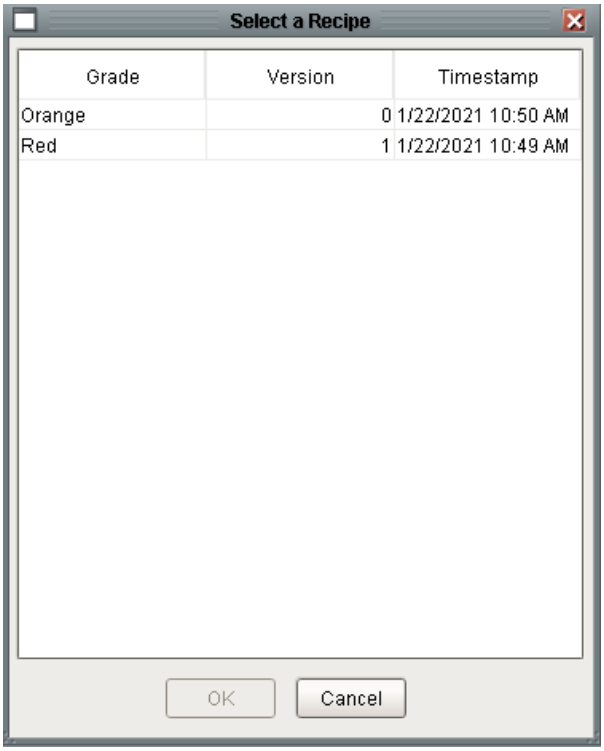

Figure 4 – Grade/Version Selection Screen

2.5 Recipe Review Screen

The "Recipe Review" screen is the centerpiece of the module. The features of the screen are described below.

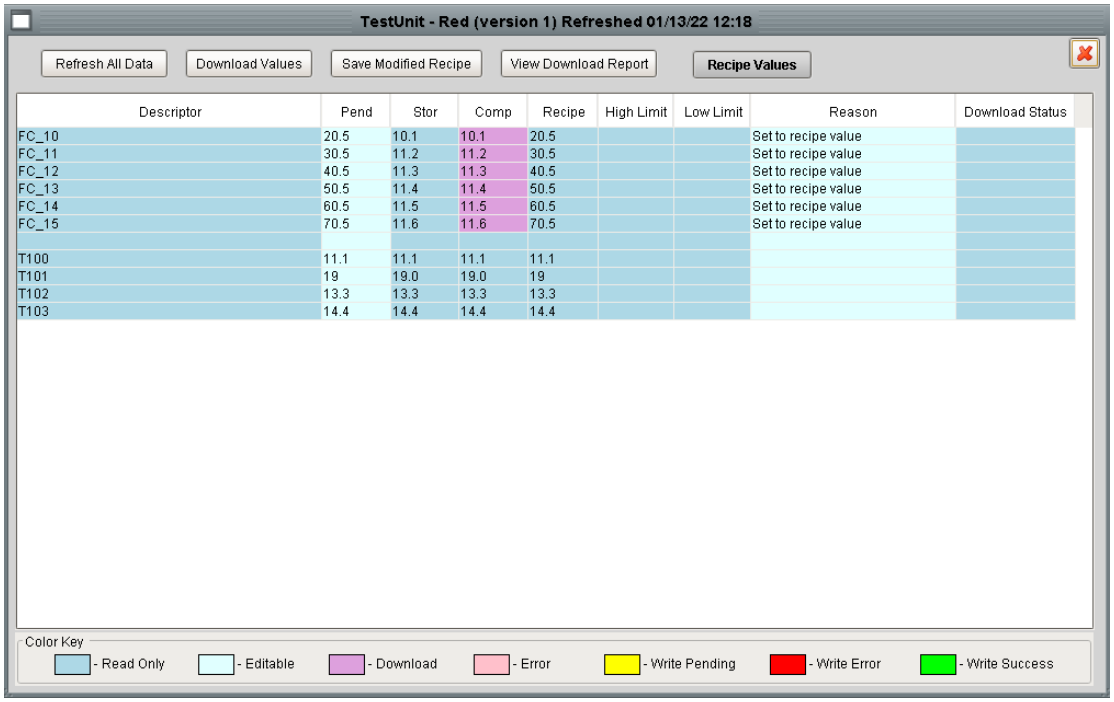

Figure 5 - Recipe Review Screen

2.5.1 Color Codes

Colors are used to indicate the success or failure of individual recipe value downloads. These color codes are summarized below. Note that the default color values are used in the descriptions below. As each Ignition site is able to customize the colors, users should review the Ignition help for the site recipes to see the actual colors used for information.

Tag Color (defaults)	Description
Light Blue	Pre-Download: Manual entry of data in the cell allowed. Post-Download: Values (row) download was confirmed.
Red	At least one value (row) download was not confirmed
Pink	The value in the DCS does not equal the recommended value to download.
Purple	A manual entry or value change requires the value to be downloaded.

Workspace Color (defaults)	Description
Green	All values downloaded successfully
Red	Some PEND values not downloaded
White	New recipe with no download attempted
Purple	One of the value (row) downloads could not be confirmed in a reasonable amount of time.

2.5.2 Table Description

The columns that are used in the recipe tables are described below:

Column	Visible	Description
Descriptor	Yes	Describes in plain English the specific component of the recipe. Units for the value are usually provided in brackets at the end of the description.
Pend	Yes	Contains the pending (new) value that will be downloaded to the tag in the DCS or Ignition tag when the download button is pressed. In recipe download mode, this column initially contains the recommended value. In Mid-Grade download, this column initially contains the current value.
Stor	Yes	The current value of the Stor tag.
Comp	Yes	The current value of the Comp tag.
Recc	Yes	The recommended value that should be downloaded.
High Limit	Yes	The upper limit for a value entered into the PEND field. If the value in the PEND field is larger than this limit, a warning is issued and the new value is rejected. Use the scroll arrows at the bottom of the spreadsheet to view the values in this column
Low Limit	Yes	The lower limit for a value entered into the PEND field. If the value in the PEND field is smaller than this limit, a warning is issued and the new value is rejected. Use the scroll arrows at the bottom of the spreadsheet to view the values in this column
Reason	Yes	Provide a reason for downloading a PEND value that is different from the RECC value. After the operator enters a reason, the operator can then change the PEND value

Column	Visible	Description
Store Tag	No	The tag that the PEND value will be downloaded to when the download button is pressed. Normally, the STOR and COMP tags (and values) are the same. But, the STOR and COMP tags can be different. For example, different tags would be used if the user needed to download a value to an intermediate DCS tag (STOR TAG) that is then processed at some point in time (i.e., via BPL) to move the value to the final destination (COMP TAG).
Comp Tag	No	The tag that is the ultimate destination of the download of the Pend value.
Access	No	The type of access is required to change manually the values in the PEND column. **CC** – Comment lines used to separate or clarify entry information. **AE –** Editable only by an AE **EO** – Operator cannot see or edit Pend value, visible and editable by an AE. **OC** – Operator can edit the Pend value
Write Location	No	The alias of the write location as defined via the RtWriteLocation table. This is used during the building of a recipe within the Recipe Toolkit. Essentially, the Recipe Toolkit building of a recipe will associate each write location with its unique OPC data location. This process enables writes to several OPC locations from a single installation of Recipe Toolkit. Additionally, this set-up will allow a single grade recipe to span OPC locations. For example, one can set-up the aliases of LCN1 and LCN2 in the WriteLocation table and then equate those locations with their respective TPNServers. Each line of the recipe with LCN1 in is master database will then go to the LCN1 TPNServer while the others with LCN2 will go to the other TPNServer. **NOTE: there is no default write location – the developer MUST specify all locations desired even if it is only one location.**
Step	No	Defines the display order of the recipe, automatically assigned by the DB Manager application.
Mode Attribute	No	Certain settings in a DCS require that a controller be put into a mode attribute setting so that an application external to the DCS can download a value to the controller. If the controller requires such a setup, the tag attribute to set is filled in this field. For example, a DCS could require that a controller mode attribute be put to PROGRAM to accept values. The Recipe Toolkit should in an ideal scenario place the controller to the correct setting and return the controller to the previous setting when the value is downloaded. Note: At this time, the operator is required to have the controller in the proper control mode (computer, cascade, etc.) so that a value such as a setpoint can be successfully downloaded. For example, if a SP is downloaded to a controller in MAN with PV tracking enabled, the SP store would fail.

Column	Visible	Description
Mode Attribute Value	No	Defines the value (text or number) to be downloaded to *MODATTR*.
Download Type	No	This column is for internal use only. Values are Immediate, Deferred Value, or Skip. Immediate is used for simple tags, either internal Ignition tags or simple outputs. Deferred Value is used when writing to several related tags (setpoints and limits) where the order of download is important). Skip indicates that the operator has entered a blank pend value which means that nothing will be written to the tag.
Download	No	This column is for internal use only. Checkbox that indicates if the current pend value will be downloaded. It is based in comparing the pend value to the comp value.
Data Type	No	This column is for internal use only. The data type of the tag.
Plan Status	No	This column is for internal use only. This is used to drive the animation of the Comp column.
Download Status	No	This column is for internal use only. Once the download button is pressed, this column will indicate the status of each attempted write.
ValueId	No	This column is for internal use only. The id of the tag in the recipe database.

2.5.3 Button Descriptions

The buttons that are at the top of the window are described below:

Button	Description
Refresh All Data	Updates the STOR and COMP columns with the most current data from the DCS or Ignition. It will also update the status of the download as indicated by the animation in the Comp column.
Download Values	Writes the PEND values to the DCS. PEND values that are different from the STOR/COMP values will be written. See section 2.6 for details.
Save Modified Recipe	Save the values shown in the PEND column to the recipe database creating a new version of the grade. Note that a new version will be saved even if no changes to the recipe were made. The new version will not be active and therefore will not be available for download unless made active by an engineer with write access to the recipe database (i.e., contact engineer).
View Download Report	Opens a window that will show the results of the current download along with every attempted download. See section 3.7 fort details
Recipe Values / Process Values	This is a two state button and is described below in section 2.5.4.

2.5.4 Recipe Values / Process Values

This is a two state button. The state of the button indicates the source of the values in the "Pend" column. If the state is "Recipe Values" then the recipe values will be used. If the state is "Process Values" then the comp value will be used. The initial state of the button is set based on route that the user took to open the window. If the user navigated via the grade change route then the initial state will be "Recipe Values". If the operator navigated via the incremental change route then the state will be "Process Values".

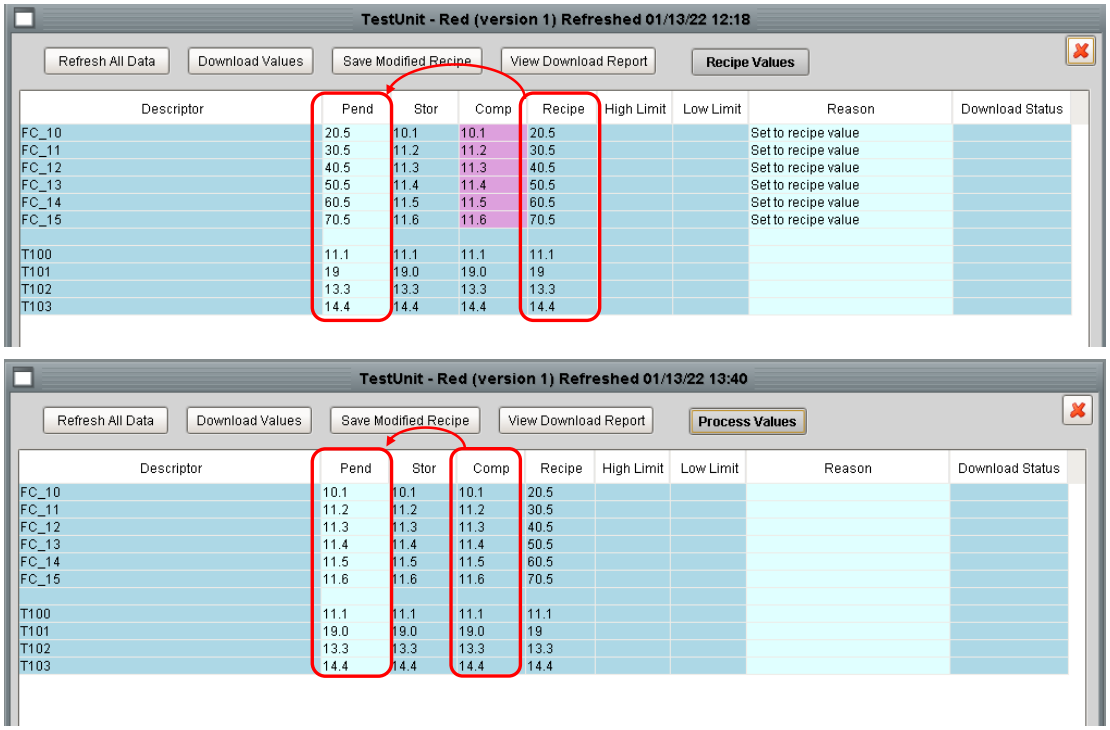

Figure 6 - Recipe Values vs Process Values

2.5.5 Operator / AE Specialization

The data displayed in the recipe table and the ability to edit individual Pend values is based on the role of the user. All ExxonMobil sites use Microsoft Active Directory security settings. The security settings do not provide a global AE role. This is intentional, because AEs from one site should not be able to access a different site unless specifically granted access to it. Therefore the roles that determine whether an individual is an AE is site specific. The specific roles that equate to an AE for a site should be entered in the *RoleTranslation* table. Refer to the *Common Facilities User's Guide* for further information

2.5.6 Modifying the Recipe

The operator can modify the PEND and the REASON values for rows where the **Access** column contains **OC** (far right side of spreadsheet). These values are typically shown in light blue (default) whereas most of the other non-editable values are shown dark blue (default).

NOTE: If the PEND value is changed to the RECC value, no comment is required since the default comment of "Set to recommended value" will be automatically placed into the REASON field. If the PEND value and COMP field are different, the COMP value be shown in pink (default).

Any value entered into the PEND field will be checked against the high and low limits for that entry. If the entry is outside of the limits, an error box will be displayed. The operator should try a different value within the limits or notify the appropriate engineer of the need to enter a value outside of the normal operating limits. The engineer will need to alter the limits in the database and the recipe will have to be fetched again. Not all rows have high and low limits. For these cases, the operator should make sure that the value entered into the PEND field makes sense before downloading the value. After modifying any PEND fields, the operator should review the changes to make sure that the new values are correct before downloading the recipe values to the DCS.

If a blank value is entered in the PEND column then nothing will be written to the DCS regardless of the value in the DCS. This is new behavior!

NOTE: If the value of **NaN** is shown in the PEND cell, a value of float("NaN") will be downloaded to the tag. If a value of **NaN** is shown in a high or low limit cell, then no limit checking on the PEND value occurs. **NaN** literally means "Not a Number" and is defined as in Python as float("NaN").

2.6 Recipe Download

The following sections describe what happens when the user presses the "Download Values" button. The details about how the recipe download is processed in discussed in section 3.

2.6.1 Timeout

Each tag write has its own timeout that is individually determined based on the class of the tag and should consider the communication scan rate. There is also an overall timeout that serves as a safeguard for the overall download. Because the download are processed in parallel, the timeout is independent of the number of tags to be written. The time is site configurable via the tag named *downloadTime* in the *Recipe/Constants* folder. An initial setting of two minutes is reasonable.

2.6.2 Feedback

Once a download starts the operator will receive feedback as described in the following sections.

2.6.2.1 Success

As each value is successfully downloaded from Ignition to the DCS tag or Ignition tag, the STOR and COMP values will update to be the PEND value. The background of the COMP field will change from pink to light blue if the download is successfully confirmed.

If all PEND values downloaded were successfully confirmed, the background of the recipe spreadsheet will turn green.

2.6.2.2 Failure

If the download of a PEND value is not successfully confirmed, the STOR and/or COMP value will not equal the PEND value. The background of the COMP field will change from pink to red. The background of the recipe spreadsheet will turn red.

A download can fail for a number of reasons; some of which are summarized below. The operator should check each row where the PEND value does not equal the CURR value (look for red backgrounds in cells). Then for each row, the operator should determine how best to resolve the download failure.

1. The recipe value being downloaded was below or above a setpoint clamp of the tag. The operator should adjust the control setpoint limit, decide to keep the current value, or notify an engineer of the discrepancy in the setpoint limits and recipe value. For example if a value of 20 psig were being written to the setpoint of a controller with a low setpoint limit of 25 psig, the new value of 20 would be rejected. The operator could move the lower setpoint limit and then try to either enter the new recipe setpoint value manually or re-download the recipe value from Ignition.
2. The controller in the DCS is output clamped (high or low) even though a setpoint limit has not been reached. The new value from the recipe cannot be entered until the constrained controller is moved away from the output constraint.
3. The recipe value being downloaded is outside the range of the tag even though the value passed the high and low limit checks in the Ignition recipe spreadsheet. The operator should verify that the recipe value is correct. If so, notify an engineer of the differences in ranges in the recipe and the DCS tag.
4. The download of the value took an exceptionally long time and could not be confirmed in timely manner. The background of the workspace will be shown in purple if this error occurs. The operator can try to re-download the recipe to check for confirmation or review the DCS tag to see if the value was actually downloaded.
5. The OPC to DCS communication has failed which prevents the recipe values from being downloaded to the DCS or verified as downloaded. Contact an engineer if this error occurs.

NOTE: The "Review Download Errors" opens the Download Log user interface which displays all of the tag downloads, both successes and failures. See section 3.7 for additional details.

3 Tag Downloads, Monitoring/Auditing, Automatic Download

This section outlines the tasks that are performed by the recipe 'engine' automatically to download, monitor and audit tag values.

3.1 Creating and Deleting Tags

OPC tags are created automatically by the recipe system as they are needed. Shared tags that are used by the recipe, and other toolkits, are implemented via memory tags and are created programmatically on startup.

3.1.1 OPC Tags

OPC tags are created whenever a new recipe is selected and fetched from the database. Tags are defined in the *RtValueDefinition* table where the *WriteLocation* names an alias in the *RtWriteLocation* table. OPC tags are implemented using the UDTs shown below. Refer to the **I/O Facilities Design Specification** for details.

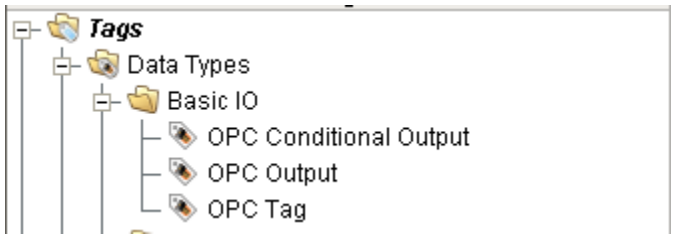

Figure 7 - Basic I/O User Defined Types Used by Recipe Toolkit

3.1.2 Local Memory Tags

The Recipe toolkit writes values to OPC tags and to memory tags. By definition, memory tags are local to the Ignition system and are only used by other applications in the Ignition platform. These tags are specified by the "Write Location" value: *Local*. These tags are expected to exist if they are specified in the recipe. The recipe toolkit will not create or delete these tags.

These tags may be created manually or programmatically in a site specific script. For Vistalon, this script is in the external Python library named: *xom.vistalon.startup.createTags()* which is called by *xom.vistalon.startup.gateway()* which is called on startup.

3.1.3 Recipe Detail User Defined Type

Recipe Detail UDTs may be created automatically whenever a new recipe is selected and fetched from the database. A recipe detail object is created, if one does not already exist,

whenever a tag with certain suffixes is encountered. The suffixes and recipe detail type are described below.

Suffixes	Recipe Detail Type
SP, SPCH, SPCL, SPHILM, SPLOLIM	Recipe SP Details
PV, PVHILM, PVLOLM	Recipe PV Details
OP, OPCH, OPCL, OPHILM, OPLOLM	Recipe OP Details

The purpose of these objects is to define the relationship between otherwise independent tags. The relationship is used to coordinate writes to the same controller where we may write a SP and limits around that SP. Although the tags may actually be part of a single controller in the DCS, in Ignition they have been defined as stand-alone tags. It is important that the proper order is used so that a transient alert is not triggered. The coordination is described in section 3.2 It is possible that a PV, SP, or OP tag is encountered without any corresponding limit object. In that case, the detail tag is not really needed, but it will be created anyway.

The UDT for implementing recipe details is:

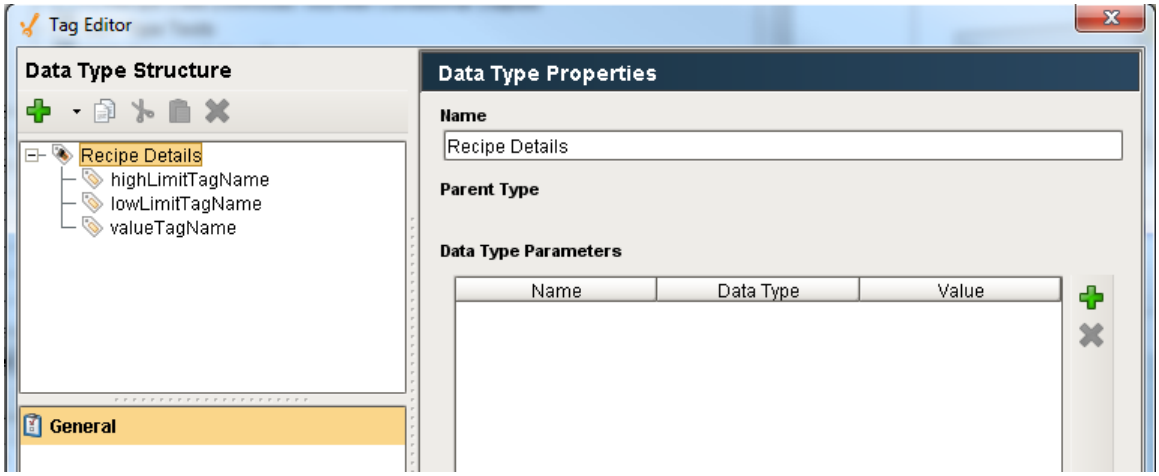

Figure 8 - Recipe Detail User Defined Type

3.2 Clamps – Order of Download

When it is determined that there are clamps for a download, there are as many as three tags whose writes need to be correctly ordered: a high limit, the value, and a low limit. The order must be determined such that value never violates the limits. For example, if the current value of the high limit, value, and low limit are 60, 50, and 40 respectively, and the new values are 90, 80, and 70. Then the order of writes must be high limit, value, and then low limit. Conversely, if the current values are 90, 80, and 70 and the new

values are 60, 50, and 40; then the download order must be low limit, value, and then high limit. Whereas all simple write proceed in parallel, these write must occur serially. This is implemented in *ils.io.recipedetail.py* and is accessed by the API *writeRecipeDetail* in *ils.io.api*.

3.3 Parallel Download

Parallel download means that recipe values are downloaded in a fashion where successive tag writes are not dependent on waiting for a successful previous tag write. A success is determined by reading the tag value after the write and comparing it to the target value.

3.4 Monitoring & Audit

The Recipe Toolkit monitors the progress of the download. The background color of comp column is animated based on the status of each individual download. Additionally, the status of each write is also logged to the *RtDownloadDetail* table for a permanent record.

An example of a situation where a download would fail is the case where a contact engineer does not specify specific clamps within a recipe but such clamps exist in the DCS (the operator manually set some clamp values). A SP, OP or PV download is attempted where the level is outside the clamp range. The DCS may either reject the value or set the level at the clamp boundary. This discrepancy is automatically detected by the Recipe Toolkit by a simple comparison of the desired download value and read back value.

3.5 Automatic Download

There are two types of automatic downloads that are supported. The first type, hereafter referred to as fully automatic, is completely automatic without any user interface or interaction. The second type, hereafter referred to as semi-automatic, automatically selects the grade and displays the recipe download user interface on the appropriate client. The recipe is only downloaded when the user presses OK.

3.5.1 Trigger Mechanism

Regardless of the type of automatic download, it is triggered from the DCS via an OPC tag embedded in a *Download Trigger* UDT as shown below.

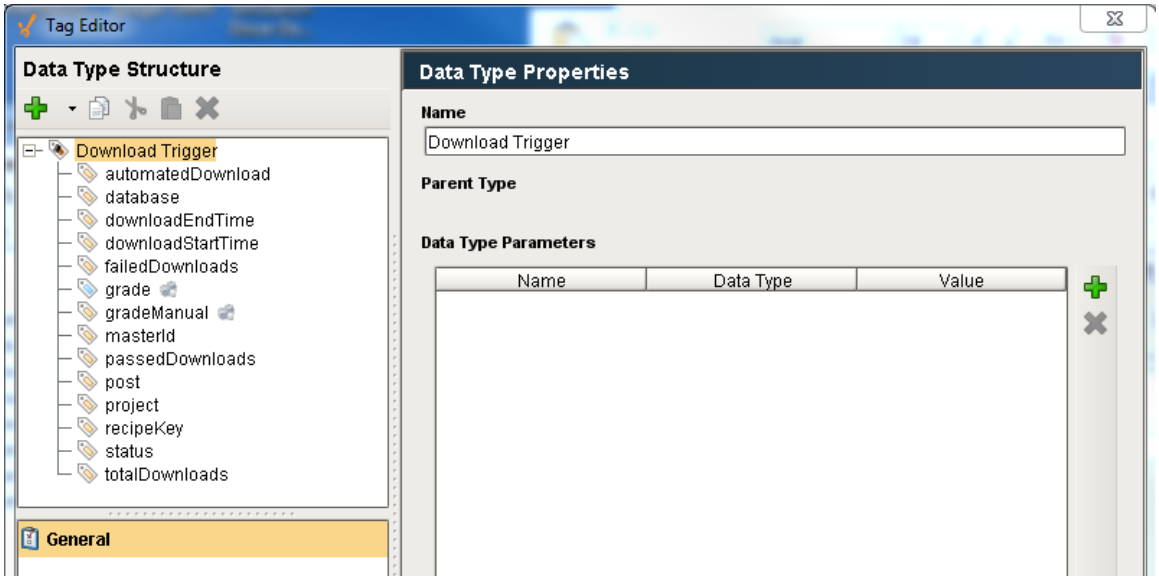

Figure 9 - Download Trigger User Defined Type

The OPC server and item-id of the grade tag need to be manually configured. The other tags in the UDT are all memory tags. The memory tags that need to be manually configured to customize the behavior of the UDT are:

Tag	Description
automatedDownload	If True, then the download will be fully automatic without any user interface. If False, then the user interface will be automatically displayed, but the download will not proceed until the Download button is pressed.
Database	Name of the database connection to use for fetching the recipe and logging the download results. This is only relevant for fully automatic downloads which run in the gateway and do not have a database context.
Post	If automatedDownload is false, then this is used to find the client where the user interface will be displayed.
Project	If automatedDownload is false, then this defines the clients that will receive the message. If automatedDownload is True then project is used to specify the project since tags are not part of a project.
recipeKey	Specifies the recipeKey / unit to use when fetching the recipe from the database. This is crucial for systems where multiple consoles are in the same project.

The *grade* and *gradeManual* tags both trigger the download to proceed via tag event script. The *grade* tag in an OPC tag which allows the download to be triggered from the

DCS. The *gradeManual* tag allows the download to be triggered from some logic elsewhere in the Ignition toolkit such as a SFC. Both tags have an identical handler.

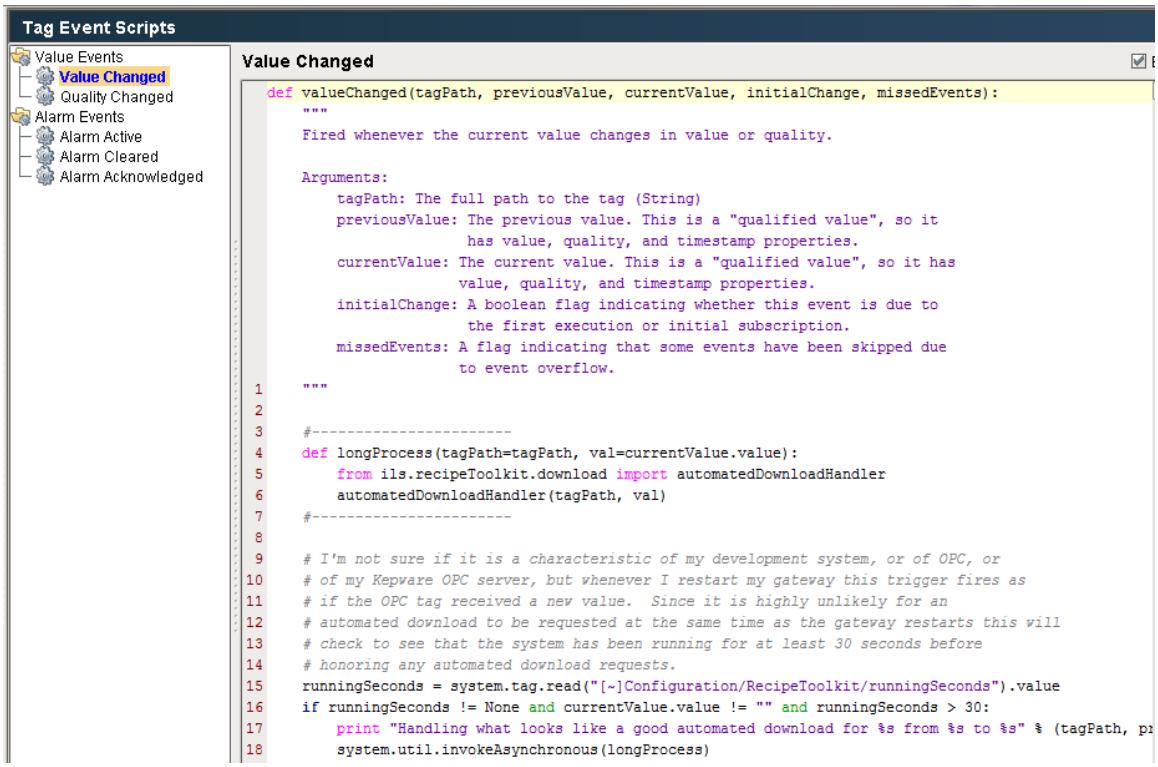

Figure 10 - Download Trigger Logic

3.5.2 Creating Trigger Instances

Triggers are created manually and configured for every recipe key / unit that supports automated download. The instances should be created in an appropriate place in the Site folder. They should NOT be placed in the recipe key / unit specific folders. The download trigger for the Escorez application is shown below. This trigger is driven by a SFC which uses the *gradeManual* tag; therefore the *grade* OPC tag is not configured.

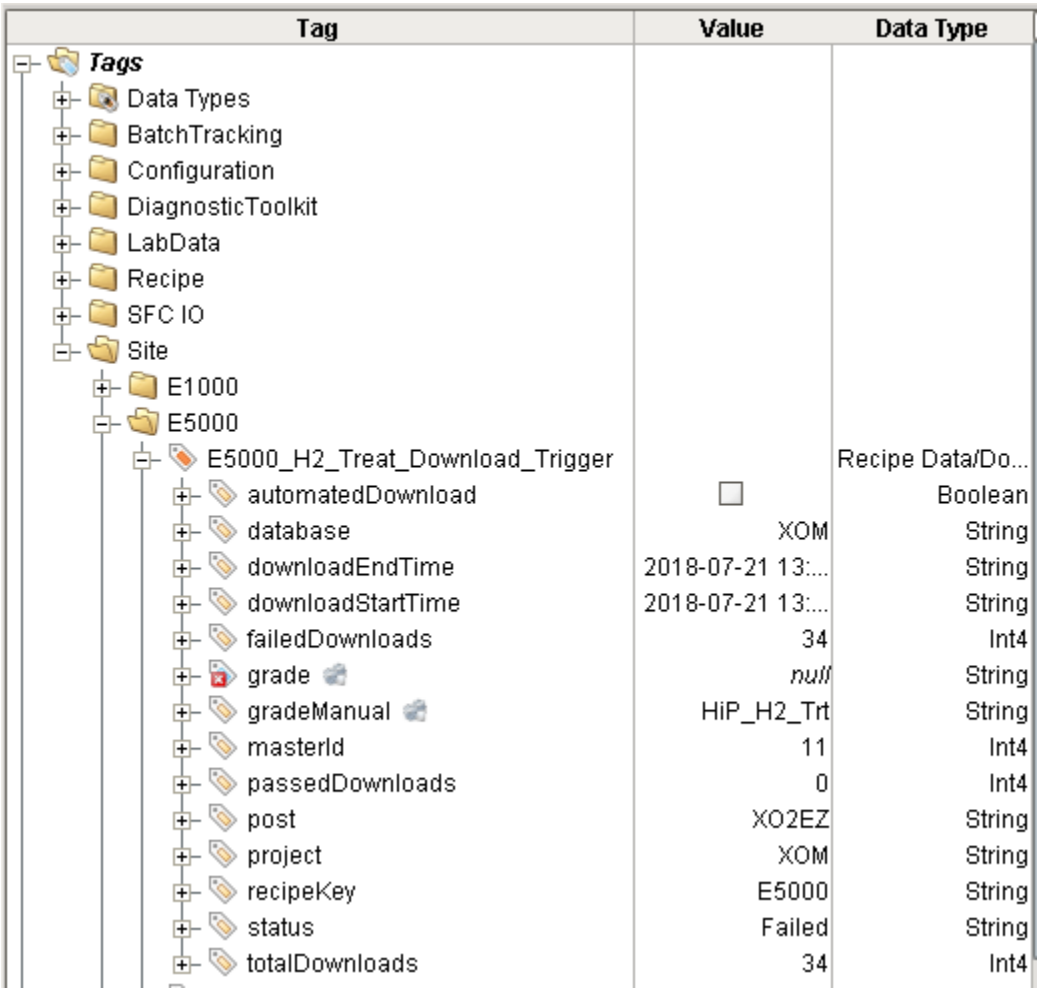

Figure 11 - Automated Download Trigger Tag for Escorez

3.5.3 Semi-Automatic Download

A semi-automatic download means that the DCS will trigger the grade selection and display of the recipe download user interface. A semi-automatic download is configured by setting the *automatedDownload* memory tag to *False*. The console is determined from the post tag.

The flow of execution begins with the tag event script on the grade or gradeManual tag:

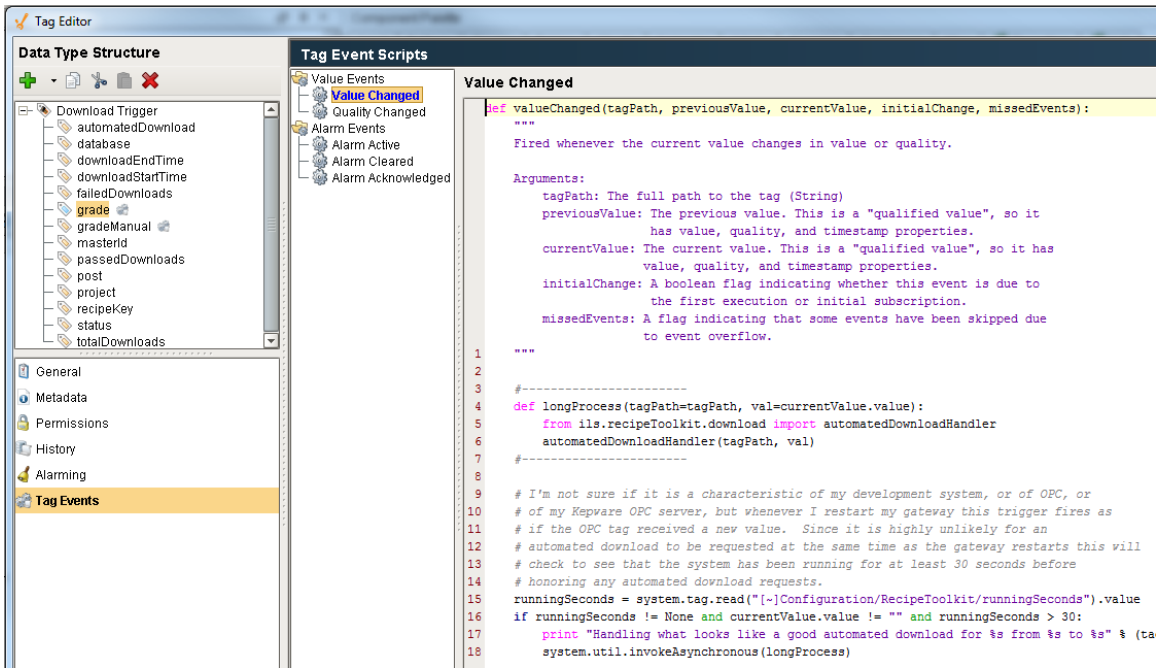

Figure 12 - Automated Download Tag Event Script

The script calls *ils.recipeToolkit.download.automatedDownloadHandler()*, which is in the external Python library. The script determines if the download is fully automatic or semi-automatic. If the download is semi-automatic, then a message is sent to every client. The client determines if it is the appropriate target for the message. This message is handled in the client message handler below:

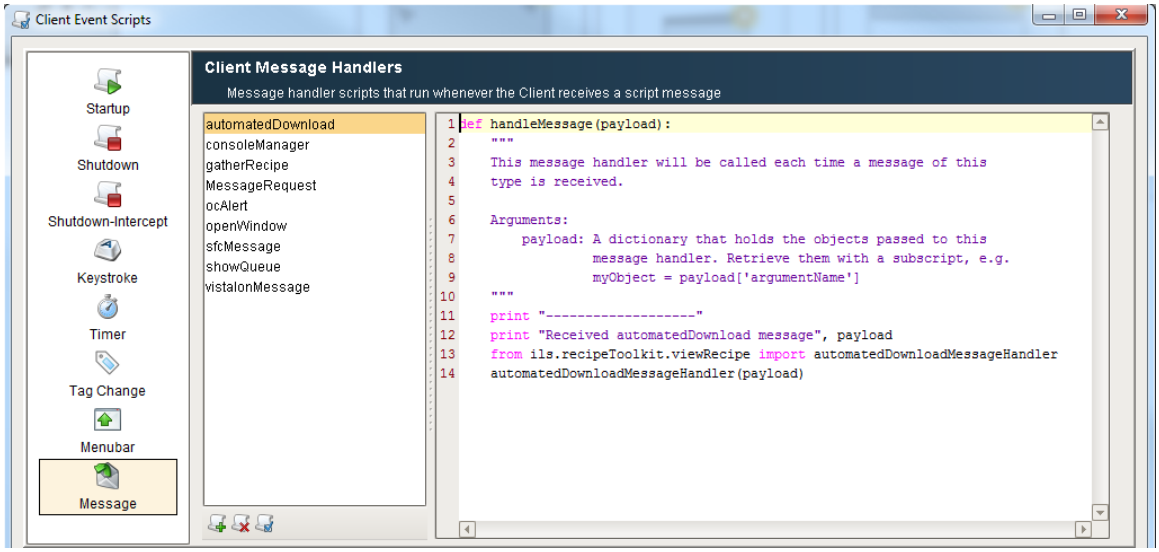

Figure 13 - Semi-Automatic Message Handler

The handler calls *ils.recipeToolkit.viewRecipe.automatedDownloadMessageHandler()*. The script determines if the message is intended for the client by comparing the post in the message to the post for the client. (The post for the client is determined when the client connects and is stored in the client tag post, see the *Common Facilities User's Guide* for details). If the message post and the client post are a match, then the Recipe Download GUI is displayed. The actual download does not happen until the operator presses the "Download" button. The operator may edit the pend values as usual.

For both fully automatic and semi-automatic downloads, the UDT that triggered the download are updated with the status of the download. This can be monitored to determine when the download is complete and if it was successful.

3.5.4 Fully Automatic Download

A fully automatic download proceeds much the same way, except that instead of sending a message to the client, the recipe is automatically fetched from the database and immediately written to DCS. A fully automatic download is configured by setting the *automatedDownload* memory tag to *True*. There is no feedback to any client that a download is taking place. Just as with a manual or semi-automatic download, the results of an automatic download are written to the *RtDownloadMaster* and *RtDownloadDetail* tables and are visible from the Download Logging screen, section 3.7.

3.6 Logbook Messages

As with all toolkit modules, generic messages can be posted to the system log and to the console message queue. Manual and automatic downloads are recorded in the system log as shown below.

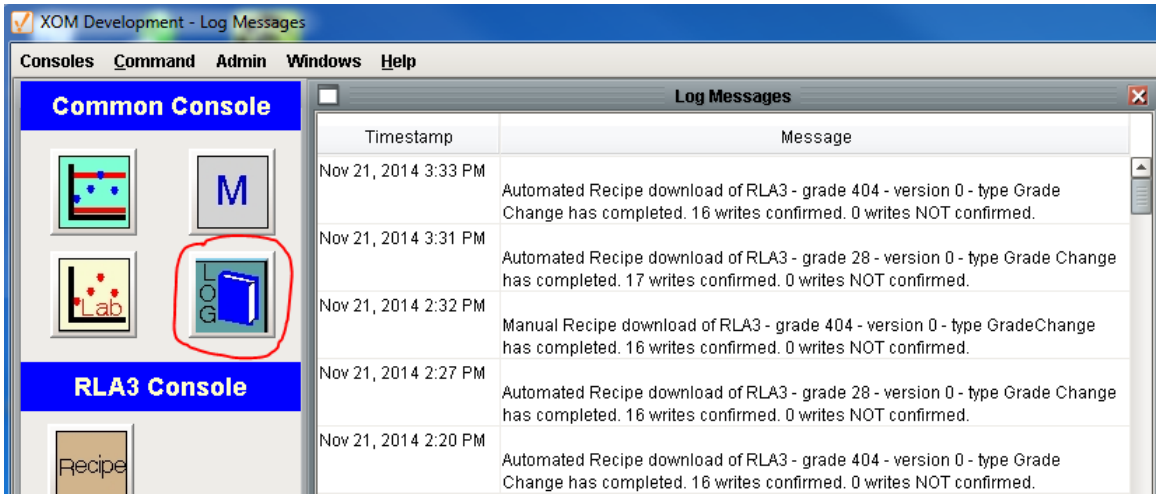

Figure 14 - Download Message in System Log

3.7 Download Logging

There are two levels of logging used. A summary message that includes the number of successful and unsuccessful writes is written to the console queue. The details of the download are also recorded to the recipe toolkit database. This screen is available from the *View Download Report* button and from the main *Admin* menu.

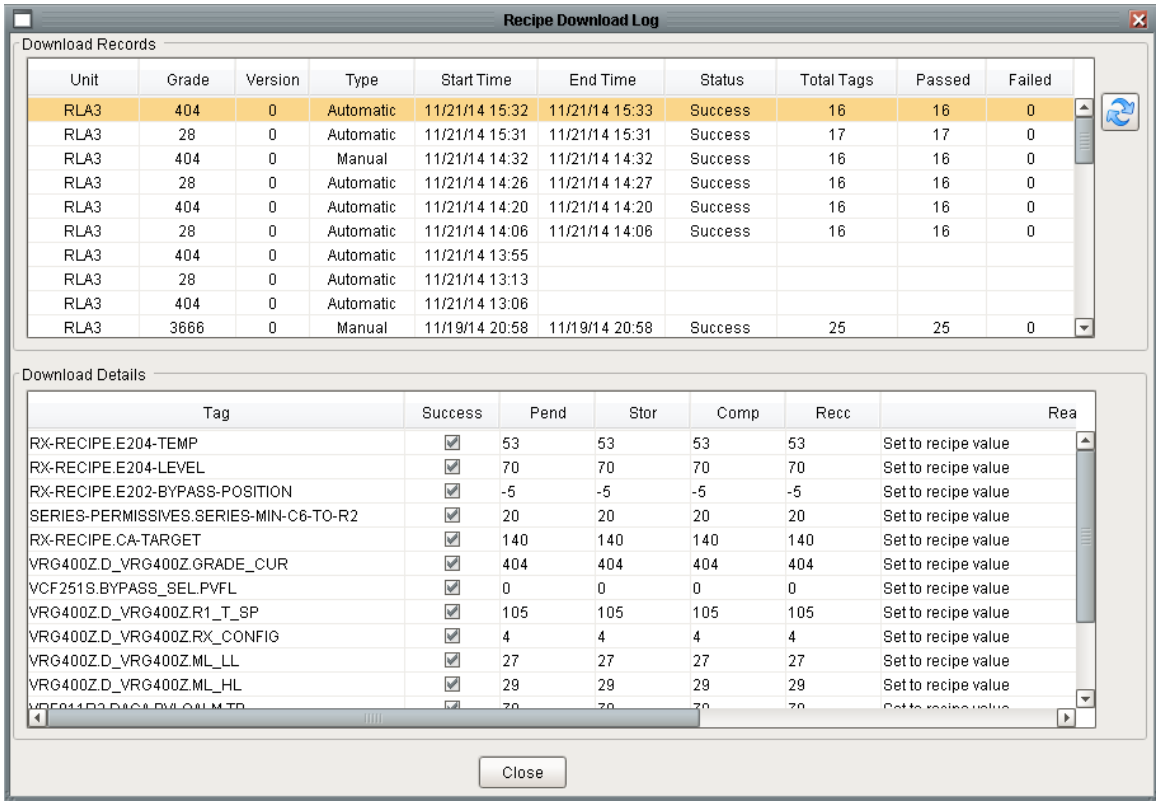

Figure 15 - Detailed Download Logging

3.8 Customization and Configuration

There are a number of settings that can be used to customize the behavior of the Recipe Toolkit. The settings are implemented as memory tags that are created automatically on startup. The default values are shown below. If tag values are changed they are never overwritten, even when new software is installed because these tags are never included in the installer.

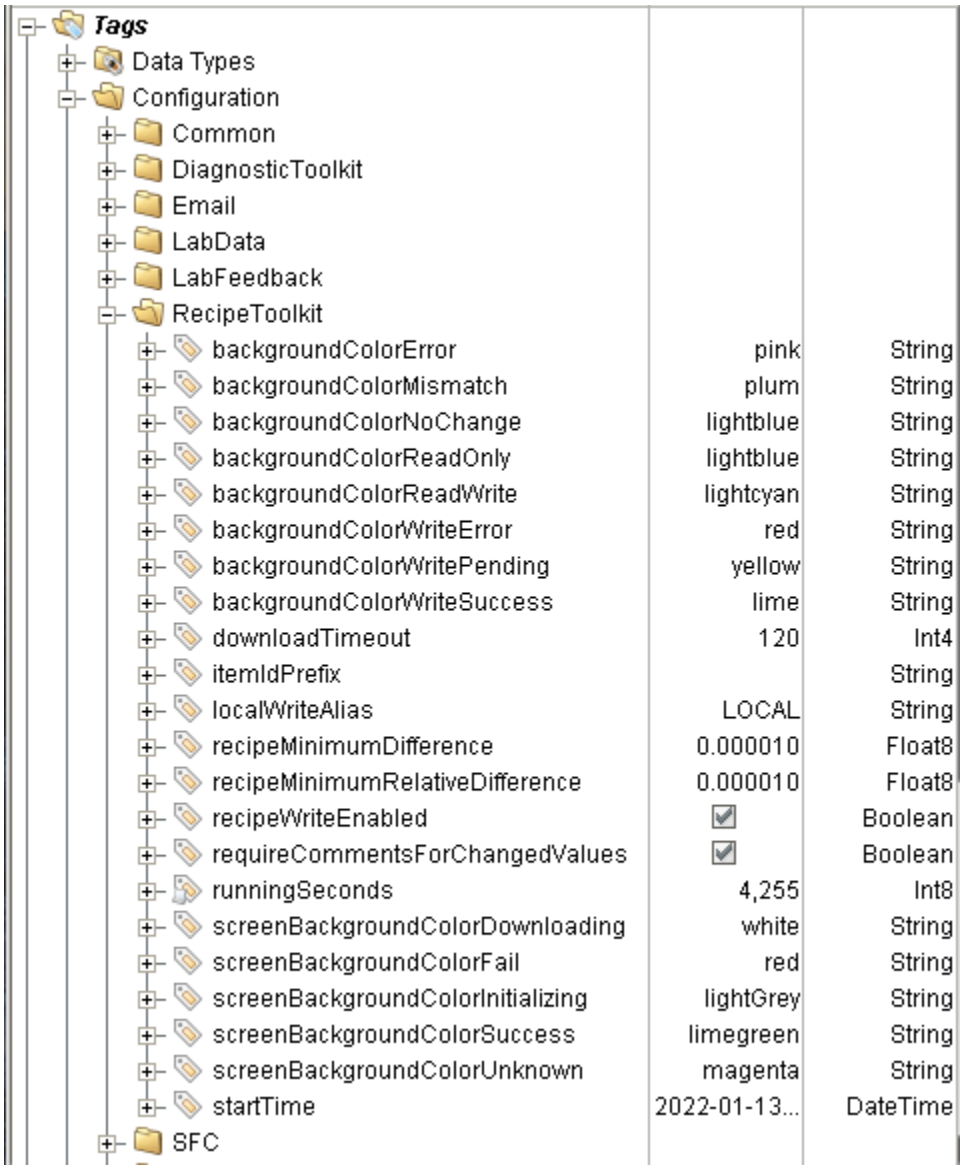

Figure 16 - Recipe Toolkit Configuration Tags

There are several settings that deserve explanation.

Setting	Description
downloadTimeout	Time in seconds for the download monitoring to process after starting a download. The results will be summarized when the time expires.
itemIdPrefix	Normally blank, this is used for sites where a fixed prefix precedes the item id of OPC tags. This was added to facilitate testing.
localWriteAlias	Value of the *WriteLocation* specified in the recipe that equates to a local memory tag. The default value is "LocalG2"
recipeMinimumDifference	Used to compare the download value with the actual value. See the algorithm below.
recipeMinimumRelativeDifference	Used to compare the download value with the actual value. See the algorithm below.
recipeWriteEnabled	Used to disable tag writes from the Recipe Toolkit. This will not have any effect on writes from other toolkits. There is another tag, *Configuration/writeEnabled*, that *does* disable all tag writes.
requireCommentsForChangedValues	If True, then the operator will be required to enter a comment describing why a new value was entered. If changing values is an expected part of a normal workflow then this should probably be set to False.

To determine is a tag write is successful, the value is read back from the tag and is compared to the value that is written. Because the value passes through a number of systems (Ignition, OPC Server, DCS) there are a number of opportunities for round off errors to be introduced. Therefore, the algorithm used to determine if the downloaded value is the same as the written value is shown below.

```
minThreshold = abs(recipeMinimumRelativeDifference * float(val1))
if minThreshold < recipeMinimumDifference:
    minThreshold = recipeMinimumDifference

if abs(float(val1) - float(val2)) < minThreshold:
    return True
else:
    return False
```

Figure 17 - Value Comparison Algorithm

4 I/O Implementation

The I/O used by the recipe toolkit utilizes the facilities provided by the common I/O facilities, refer to the I/O Design Specification for details. This section describes extensions to the standard facilities.

4.1 Download Trigger

Refer to section 3.5.1 describes a UDT for implementing automatic recipe downloads.

4.2 Recipe Details

This UDT is used to coordinate writes to a controller in the DCS when there are high and/or low limits and an associated setpoint where the writes need to be coordinated in order to not violate the limits. The recipe toolkit creates instances of the UDT as needed based on the configuration in the recipe database.

The UDT definition is shown below:

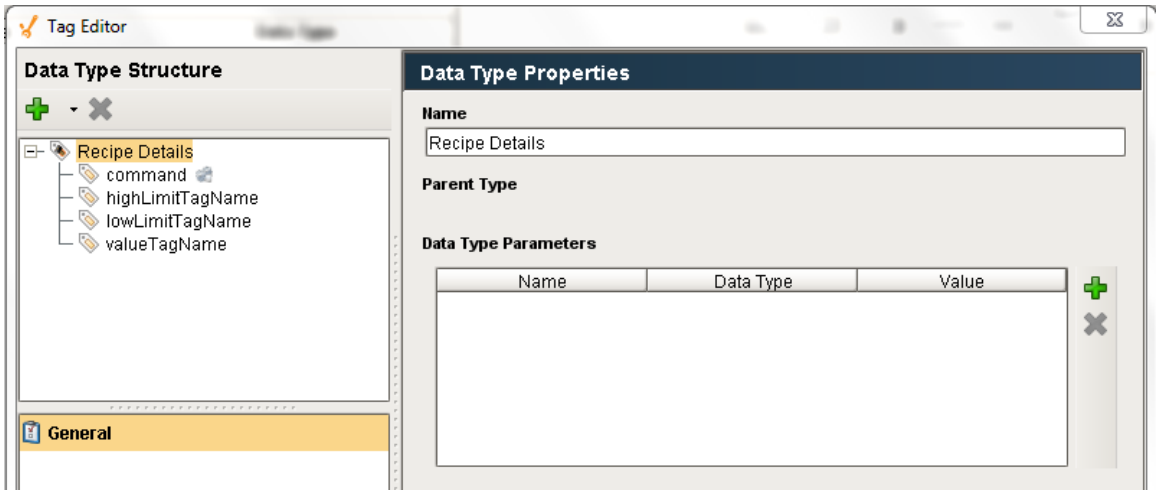

Figure 18 - Recipe Details UDT