**Master Script Upgrade to EMSE 3.0**

**Clearwater, FL**

**Aug 28, 2018**

Version 1.1

# TABLE OF CONTENTS

[TABLE OF CONTENTS 2](#_Toc523220601)

[Document Control 3](#_Toc523220602)

[Summary 3](#_Toc523220603)

[Analysis 3](#_Toc523220604)

[Conversion Details 4](#_Toc523220605)

[Branch execution order inconsistent with the 3.0 architecture 4](#_Toc523220606)

[Business-Specific code located in base control strings 5](#_Toc523220607)

[Consolidations 6](#_Toc523220608)

[new custom functions created 7](#_Toc523220609)

[Special Event Considerations 7](#_Toc523220610)

[Intentional Omissions 8](#_Toc523220611)

[No Active Business-Specific code outside of variable branch structure (Base Control Strings) 8](#_Toc523220612)

[Unused enabled events 8](#_Toc523220613)

[Abandoned approaches, disabled, or not called from existing Active code 9](#_Toc523220614)

[Recommendations 10](#_Toc523220615)

[repository contents 13](#_Toc523220616)

[Unit Test Suggestions 13](#_Toc523220617)

[test issues – additional discovery 16](#_Toc523220618)

[ApplicationConditionAddBefore 16](#_Toc523220619)

[ApplicationConditionUpdateBefore 16](#_Toc523220620)

[ApplicationSpecificInfoUpdateAfter 16](#_Toc523220621)

[ApplicationSpecificInfoUpdateBefore 16](#_Toc523220622)

[ApplicationStatusUpdateBefore 16](#_Toc523220623)

[ApplicationSubmitAfter 18](#_Toc523220624)

[ApplicationSubmitBefore 18](#_Toc523220625)

[ContactAddAfter 18](#_Toc523220626)

[ContactEditAfter 18](#_Toc523220627)

[ConvertToRealCapAfter 19](#_Toc523220628)

[ConvertToRealCapBefore 19](#_Toc523220629)

[DOCUMENTUPLOADAFTER 19](#_Toc523220630)

[DocumentUploadBefore 19](#_Toc523220631)

[FeeAssessAfter 19](#_Toc523220632)

[InspectionResultSubmitAfter 19](#_Toc523220633)

[InspectionScheduleAfter 19](#_Toc523220634)

[InspectionScheduleBefore 19](#_Toc523220635)

[PaymentReceiveAfter 19](#_Toc523220636)

[WorkflowAdhocTaskAddAfter 19](#_Toc523220637)

[WorkflowAdhocTaskAddBefore 19](#_Toc523220638)

[WORKFLOWADHOCTASKUPDATEAFTER 19](#_Toc523220639)

[WorkflowAdhocTaskUpdateBefore 19](#_Toc523220640)

[WorkflowTaskUpdateAfter 19](#_Toc523220641)

[WorkflowTaskUpdateBefore 19](#_Toc523220642)

[Batch Testing 19](#_Toc523220643)

[Pageflow testing 19](#_Toc523220644)

[UAT Issues 19](#_Toc523220645)

# Document Control

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Name** | **Version** | **Change Reference** |
| 7/24/2017 | Jeanne E. Chalk | 1.0 | Base Draft – Before dev deployment |
| 8/22/2018 | Jeremiah C. Johnson | 1.1 | Revisions before testing |
|  |  |  |  |

# Summary

Accela has upgraded the agency scripts from 1.x format to the newest scripting 3.0 format. The scripts have been moved out of standard choices and into JavaScript files, using the Accela best practice directory structure. The standard choices, which were a matrix of Control String Standard Choices and nested branch calls, have been converted to a structure that mirrors the record type structure for the agency. This document describes the details of the conversion.

# Analysis

The current scripts in use are a hybrid approach of 1.x event script methodologies.

A review of the scripting shows that much of the scripting lies within the base Control String Standard Choices and must be migrated out into appropriate variable branching. Efforts have been made to observe and preserve execution order already in place.

# Conversion Details

* To assist with testing, the original “Branch” standard choice names have been added as comments to the converted scripts. Once testing is complete these comments won’t be needed. Also included are specific notation indicating date and author of conversion. Once tested, these may be removed at the agency’s leisure.
* All references to showDebug (ES\_CW\_DEBUGGROUPS) and unnecessary activation of showMessage have been deleted as this is bad practice. The 3.0 architecture executes the INCLUDES\_CUSTOM\_GLOBALS script at the beginning of each event and debug control is user specific. Users who wish to/need to see debug messaging should update the code in the INCLUDES\_CUSTOM\_GLOBALS script to the appropriate debug level.
* Standard Choice lines that were disabled have not been converted. Branches and sub-branches which were not referenced in the code or were fully disabled are noted in the Intentional Omissions section. These were not converted.
* All script files have been formatted to JavaScript standards for indentation.

The following section details specific areas of conflict between the 1.x standard choices and the 3.0 architecture, how they have been resolved, and potential impacts to future development.

## Branch execution order inconsistent with the 3.0 architecture

As this agency was not upgrading from 2.0 there was little risk of branch order conflict.

It should be noted for future development that the 3.0 execution order is as follows:

Prefix:\*/\*/\*/\* - These should be limited in use for code that absolutely must be executed first and/or is global across the enterprise or applies to multiple modules.

Prefix:Group/\*/\*/\* - Module specific code that should be executed early.

Prefix:Group/Type/\*/\*

Prefix:Group/Type/Subtype/\*

Prefix:Group/\*/Subtype/\*

Prefix:Group/\*/Subtype/Category

Prefix:Group/\*/\*/Category

Prefix:Group/Type/\*/Category

Prefix:Group/Type/Subtype/Category – Note that explicit specific record type rules are executed last.

This order should be evaluated when deploying rules within an event that have a specific firing order dependency.

## Business-Specific code located in base control strings

The following events had module-specific code nested in the base control string (consistent with 1.x methodology). This required relocating the code to an appropriate module-specific branch and additional analysis was required to ensure consistent execution order.

A minor reduction in processing time may be noted as irrelevant code is no longer evaluated for a specific module.

* ApplicationSpecificInfoUpdateAfter
* ApplicationStatusUpdateBefore
* ApplicationSubmitAfter
* ConvertToRealCapAfter
* ConvertToRealCapBefore
* DocumentUploadAfter
* DocumentUploadBefore
* FeeAssessAfter
* InspectionResultSubmitAfter
* InspectionScheduleAfter
* InspectionScheduleBefore
* WorkflowAdhocTaskAddAfter
* WorkflowAdhocTaskAddBefore
* WorkflowAdhocTaskUpdateAfter
* WorkflowAdhocTaskUpdateBefore
* WorkflowTaskUpdateAfter
* WorkflowTaskUpdateBefore

## Consolidations

Any standard choices that were only called once have been consolidated with their calling branch in lieu of making them custom functions. Any code that iteratively loops (previously required calling a separate branch) has also been consolidated into a single function (addCodeConditions).

* ES\_ADD\_CODE\_CONDITIONS/ES\_ADD\_CODE\_CONDITIONS\_LOOP – addCodeConditions.js
* ES\_CW\_BTR\_CATEGORY\_ASIADDFICTICIOUSNOTICE/

ES\_CW\_BTR\_CATEGORY\_ASIADDFICTICIOUSNOTICE\_LOOP – btrCategoryASIAddFicticiousNotice.js

* ACA\_LP\_BEFORE\_V3/ACA\_LP\_Before
* Email\_FireInspReqd - WTUA:BUILDING/CONSTRUCTION PERMIT/\*/\*
* ES\_ACA\_NEW\_BLDG\_\_APP - CTRCA:BUILDING/CONSTRUCTION PERMIT/\*/\*
* ES\_ADD\_AMENDMENT\_FEES - ASA:BUILDING/AMENDMENT/\*/\*
* ES\_ADD\_ASI\_DATES - WTUA:BUILDING/\*/\*/\*
* ES\_ADD\_NOC - ASA:BUILDING/\*/\*/\*
* ES\_ADD\_PARCEL\_CONDITION - ASA:BUILDING/\*/\*/UNS
* ES\_BCP\_ELEV\_CERT\_INSP - ISB:BUILDING/CONSTRUCTION PERMIT/\*/\*
* ES\_BCP\_TIEIN\_INSP - ISB:BUILDING/CONSTRUCTION PERMIT/\*/\*
* ES\_CHECK\_LICTYPE - CTRCB:BUILDING/OVER THE COUNTER/\*/\*
* ES\_CHECK\_LP\_ATT\_EXP\_ACA - CTRCB:BUILDING/OVER THE COUNTER/\*/\*
* ES\_CHECK\_PLAN\_REVIEW\_STATUS - WTUB:BUILDING/CONSTRUCTION PERMIT/\*/\*
* ES\_CW\_BTR\_CATEGORY\_ASIADDFICTICIOUSNOTICE - ASA:BUSINESSTAX/RECEIPT/\*/\*
* ES\_CW\_CONVERT\_TO\_REAL\_CAP\_OTC\_CHCKBOXES - CTRCA:BUILDING/OVER THE COUNTER/\*/\*
* ES\_CW\_PLN\_CALENDAR\_NEXTMEETINGDATES - WTUA:PLANNING/\*/\*/\*
* ES\_CW\_TMP\_ADD\_INSPECTION - ASIUA:PLANNING/TEMPORARY USE/TMP/\*
* ES\_CW\_VERIFY\_TRADE\_FOR\_INSPECTION - ISB:BUILDING/\*/\*/\*
* ES\_EMAIL\_TREE\_PERMIT - WTUA:PLANNING/TREE/\*/\*
* ES\_GET\_LP\_SEQNO - ES\_CHECK\_LP\_ATT\_EXP, ES\_CHECK\_LP\_ATT\_EXP\_ACA
* ES\_ONLINE\_CUSTOMER\_REQUEST - DUA:BUILDING/\*/\*/\*
* ES\_PNU\_MOVE\_WF - WTUA:CODECOMPLIANCE/\*/PNU/\*
* ES\_REMOVE\_PARCEL\_CONDITION - WTUB:BUILDING/\*/\*/UNS
* ES\_RENBR\_CAP - ASA:PLANNING/DEVELOPMENT AGREEMENT/\*/\*
* ES\_STATE\_FEE\_UPDATE - FAA:\*/\*/\*/\*
* ES\_UPDATE\_BLDG\_WF\_TASKS - IRSA:BUILDING/\*/\*/\*
* ES\_UPDATE\_CODE\_WF\_TASKS - IRSA:CODECOMPLIANCE/\*/\*
* ES\_VERIFY\_BCP\_PERMIT\_FEES - WTUB:BUILDING/\*/\*/\*

## new custom functions created

Some branches were converted into new custom functions as they were called more than once within the branches or could potentially lend themselves to in the future due to the nature of the code (i.e. Email Notifications or fee additions):

* ES\_ACTIVATE\_REVIEWS - activateReviews
* ES\_ADD\_BLDG\_REQ\_FEES - addBldgReqFees
* ES\_ADD\_CODE\_CONDITIONS - addCodeConditions
* ES\_ADD\_PLN\_TREE\_FEES - addPlnTreeFees
* ES\_BCP\_ADD\_TRADE\_FEE - bcpAddTradeFee
* ES\_BCP\_CHECK\_FEMA - bcpCheckFEMA
* ES\_BTR\_CATEGORY\_FEES - btrCategoryFees
* ES\_BTR\_CATEGORY\_FEES\_LOOP - btrCategoryFeesLoop
* ES\_CHECK\_LP\_ATT\_EXP - checkLPAttExp
* ES\_CHECK\_LP\_COND - checkLPCond
* ES\_CK\_SUBMITTAL\_DATE - checkSubmittalDate
* ES\_CW\_DEBUGGROUPS – Moved to Accela architecture standard INCLUDES\_CUSTOM\_GLOBALS
* ES\_PARCEL\_TO\_ASI - parcelToASI

## Special Event Considerations

The following events have items of special note relevant to testing during this project and future development:

* ApplicationSubmitAfter – Caution! Code calling ES\_RENBR\_CAP was called by es\_renbr\_cap. Due to case sensitivity, this was likely not executing. It will be now, Test Carefully!
* ConvertTorRealCapAfter – getArch\_ENG\_Email may not have worked on this event before because it was not declared in that event’s master script. It will now. Test Carefully!
* VoidFeeAfter – The master script currently associated to this event is custom for the agency. The script in place will not execute variable branches as written. Should additional VFA automations be desired in the future, it is recommended that a case be logged with customer support requesting the creation of a new productized master script and this custom code should be moved to VFA:\*/\*/\*/\*.

# Intentional Omissions

The following standard choices were intentionally omitted for the following reasons:

## No Active Business-Specific code outside of variable branch structure (Base Control Strings)

The following standard choices contained debug control only or disabled lines

* ApplicationConditionAddBefore
* ApplicationConditionUpdateBefore
* ApplicationSpecificInfoUpdateBefore
* ApplicationSubmitBefore
* ContactAddAfter
* ContactEditAfter
* FeeAssessBefore
* FeeEstimateAfter
* InspectionResultModifyAfter
* InspectionResultModifyBefore
* InspectionResultSubmitBefore
* PaymentReceiveAfter

## Unused enabled events

The following events were enabled and have no specific automations or validations associated to them and/or had no master script associated. They have been disabled to prevent error and speed event execution.

* ApplicationSelectBefore – No associated Script
* FeeEstimateAfter4ACA – No associated Script
* InspectionResultModifyAfter – No Associated Script
* InspectionResultModifyBefore – No Associated Script
* InspectionResultSubmitBefore – No Associated Script

## Abandoned approaches, disabled, or not called from existing Active code

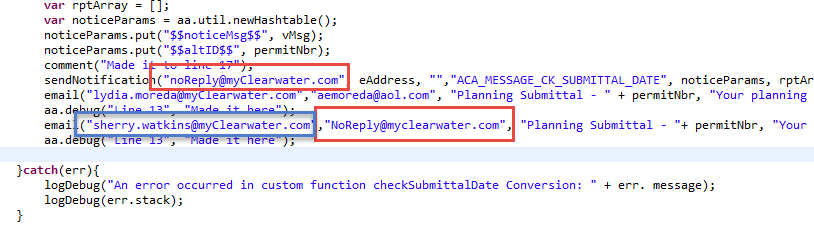
The following Standard Choices were not converted and should be retained outside of Accela for reference and re-evaluated for business need at a future date by the scripting resources and SMEs. The rough conversion of active standard choices and lines is contained in the file *“unconverted code.txt”* in the Delivery Package. It should be noted that portions of the code contained within the file is not syntactically correct and should not be deployed without additional development and testing effort.

* ES\_ADD\_PLAN\_REVIEW\_COND – Standard Choice disabled
* ES\_CHECK\_ADHOC\_TASK – not called in any standard choices or scripts
* ES\_CHECK\_BLDG\_REV – mostly disabled lines, not called in standard choices or scripts
* ES\_CW\_ASSIGN\_BLD\_INSP – not called in any standard choices or scripts
* ES\_CW\_ASSIGN\_DOC – only call disabled
* ES\_CW\_BTR\_BATCH\_RENEWALFEES – Standard Choice disabled
* ES\_CW\_PLN\_Resubmittal\_Date – Standard Choice disabled
* ES\_CW\_TESTSTANDARDCHOICE – Standard Choice disabled
* ES\_GET\_ASIT\_PLAN\_REV – Standard Choice disabled
* ES\_LOOKUP\_EDIT\_PLN\_DATES – Standard Choice disabled
* ES\_MOVE\_WORKDESC\_SHORTNOTES – Standard Choice disabled
* ES\_RPT\_CaseConditions – Standard Choice disabled
* ES\_WALK\_THRU\_REVIEW – Standard Choice disabled
* UPDATE\_TRADES – Standard Choice disabled

# Recommendations

Some recommendations on the existing code are as follows:

* Any references to the “getAppSpecific” function should be replaced with invoking the Global Variable “AInfo[]”. The getAppSpecific function will make a call to the database to get the ASI field requested. However, the master scripts natively load these values when the script begins, so a subsequent call isn’t necessary unless the value of that field is modified within the execution of the current event. This may result in a minute reduction in script execution time. Only one instance noted: btrCategoryFees, line 18
* branchTask and loopTask have largely been phased out of use as they have displayed incompatible behavior with workflows designed with the newer workflow designer. Workflows initially created in the Classic Workflow Designer should still function appropriately. For newer workflows created in the new visual Workflow designer, it is recommended to use the closeTask or setTask master script functions as appropriate. This should be considered in future development.
* The script conversion converted each standard choice line as is with the exception of CapType/Variable Branch qualification statements, look for opportunities for consolidation where repetition is noted.
  + activateReviews
* Line Numbers included in the code for debug messaging and troubleshooting have been updated to reflect new function names and line numbers. The old entries have been left in. Consider removing the old references once reviewed. Search for instances of “line” in:
  + activateReviews
  + bcpAddTradeFee
  + checkSubmittalDate
  + isNewPNU
  + ASA:Building/\*/\*/\*
  + CTRCA:Building/Construction Permit/\*/\*
  + CTRCA:Building/Over the Counter/\*/\*
  + CTRCA:Planning/\*/\*/\*
  + IRSA:Building/Fire/\*/\*
  + WTUA:Building/Construction Permit/\*/\*
  + WTUA:Planning/Tree/\*/\*
* Many instances in the code of enabled comment(), aa.print(), aa.debug(), logMessage(), and logDebug() used for function development. Any that are not indicative of critical error identification and do not normally need to be shown, consider commenting out once function development is complete.
* Camel Case convention. Two functions were found that are not using camelCase function naming convention. Renamed and calls updated: CheckForDuplicateComment (not called) and UpdateCapReviewStatus(two calls)
* Miscellaneous logic notes not previously referenced. See @TODO tags within the following functions for more suggestions:
  + countHistoryItemsForTask
  + getPlanReviewStatus
  + IRSA:CodeCompliance/\*/\*/\*
  + ISB:Building/Construction Permit/\*/\*
* Consider replacing static variables in code ([noReply@myClearwater.com](mailto:noReply@myClearwater.com) and other emails) with variable names. These variables can be declared in the INCLUDES\_CUSTOM\_GLOBALS script and maintained in one single place. Alternatively emails for specific staff work assignment notifications may be stored in a standard choice and leveraged using the lookup() master script function. This eliminates the requirement to update those values within each branch when the value needs to change. The update would be accomplished in one place and applied globally at the same time.



* Similar but inconsistent function between events. The same function across multiple events appear to be very similar but display different commentary and in some cases functionality. Examine the following for business use cases and consider consolidating to a single “master” version where appropriate and update the appropriate calls:
  + assignDueDateToTaskLevel/….WATAA, …..WTUA
  + getPrimaryEmail4PlanReview/….WATUA
  + isActiveAdhocTask(DUA)/isAdhocTaskActive(workflow events)
  + reactivateTaskLevel(WATAA)/…WATUA
* Master Script Function Override: parcelConditionExists: Function needs logic work, Should have an else, debug & return false if Condition type and Condition Desc match are not found….
* WTUB:Building/\*/\*/UNS
  + The logic in this script should be in the same event WTUA. If any event before it cancels it will not execute.
  + There is no need for an AppMatch check here it is done by the branching.
* CONVERTTOREALCAPBEFOREV3/CTRCB:BUILDING/OVER THE COUNTER/\*/\*
  + This logic would function better if it were in ASB, it is best practices to put validation there.
* AssignDueDateToTaskLevelWATUA/WATAA.js
  + Can these two functions be combined?
* GetPrimaryEmail4PlanReviewWATUA.js
  + Consoloidation should be done to combine this function with its duplicates from other events “controlString” can be used to determine event.
* IsAdhocTaskActive.js
  + This function will not work if the Ad-Hoc task has been added more than one time.

* inspUserId
  + I would recommend a custom function to be called to establish this variable and to use that function to calculate the value every time it is needed or at least in every script it is used in. Establishing a global and re-using it like is done is error prone and can easily cause new developers to the team to get lost figuring out where it comes from. You can access the UserID directly from the API no need to keep a standard choice cross referencing these example

var inspUserId = aa.person.getUser(AInfo[‘Assigned Inspector’].split(‘ ‘)[0], ‘’, AInfo[‘Assigned Inspector’].split(‘ ‘)[1]).getOutput().getUserID();

will produce the same answer.

* WTUB:BUILDING/CONSTRUCTION PERMIT/\*/\*
  + If we compare this side by side with WATUB version there is lots of redundancy can we combine these two events

# repository contents

* Agency Scripts: These are the Clearwater scripts that have been converted from standard choices into pure JavaScript.

The master script 3.0 deployment for Clearwater FL is located in GitHub.

|  |  |
| --- | --- |
| Repository Site | <https://github.com> |
| Repository URL | <https://github.com/CityOfClearwater/AccelaAutomation> / <https://github.com/CityOfClearwater/AccelaAutomation_TEST> |

To connect to the repository, the “EMSEToolConfig” standard choice will need to be configured.

* + INCLUDES\_CUSTOM: These are custom functions that were previously branch standard choices and modified Master Script Functions discovered within the master scripts. Only branch standard choices that were called from multiple locations have been converted to functions.
  + Batch Scripts – Backup only
  + Event: These are the event scripts that were previously stored as standard choices
  + Miscellaneous – Master Scripts and Non-Productized INCLUDES files. Non-productized copies of Event Master Scripts available in the 8.0.0 product version have been included. Versions 8.0.1 contain productized versions of 3.0 master scripts and those may be leveraged in lieu of the non-productized copies in a future product upgrade.
  + Pageflow Script – Rewritten to 3.0 standard.
  + Docs: This document, and deployment checklist

# Unit Test Suggestions

In addition to the unit tests used in previous upgrade exercises, the following unit tests should be completed successfully in the dev/test environment before script development is considered complete.

Proceed with testing through all relevant workflows defined and noted in each branch and for each record type. Positive and negative case scenarios (criteria = true vs criteria = false) should be considered. For standard choices that previously read as “true ^ someAction()”…. there will be no negative case apart from specific record type for which the rule was designed.

When testing, testing in the existing environment should also be performed to ensure consistency between the 1.x and 3.0 methodology. Project scope to include only conversion of scripts, not corrections. Failing logic may be noted within this document in Suggestions area but corrections will not be performed.

***Special Note! If emails are sent from your pre-production environments, Please be sure to change the email addresses on all test records to ensure that your customers are not needlessly confused by emails resulting from testing.***

| **Test Case** | **Steps** | **Expected Results** | **Record IDs** |
| --- | --- | --- | --- |
| Example:  Script: ASA:Building/\*/\*/UNS | Parcel Associated to Record does not have the Unsafe Building Parcel Condition Applied  Upon creation of Building/\*/\*/UNS record: | The Unsafe Building Standard Condition will be applied to the record | *[Fill in with record ID(s) from successful unit test]* |
|  | Parcel Associated to Record ***does*** have the Unsafe Building Parcel Condition Applied  Upon creation of Building/\*/\*/UNS record: | No additional condition will be added  ***Consider other rules that may also be firing within that branch for the combination you are submitting.*** | *[Fill in with record ID(s) from successful unit test]* |
|  | Other potential tests based on applicability of configuration:  Parcel Associated to Record does not have the Unsafe Building Parcel Condition Applied  Upon creation of OTHER record types, the condition should not be added.  Etc….. |  |  |

This conversion project was put on hold and during that hold period changes were made to scripts by the Clearwater team. The following scripts were changed, and those changes have been merged with the conversion that was started.

* + INCLUDES\_CUSTOM/checkSurchargeFee.js

SCRIPTS/EVENT/PRA;BUILDING!~!~!~.js

SCRIPTS/PAGEFLOW/ACA\_BEFORE\_JOB\_VALUE\_CHECK.js

SCRIPTS/PAGEFLOW/ACA\_DEBUG\_MSG\_EXAMPLE.js

INCLUDES\_CUSTOM/activateReviews.js

INCLUDES\_CUSTOM/addBldgReqFees.js

INCLUDES\_CUSTOM/assignInsp4AMO\_CLW.js

INCLUDES\_CUSTOM/checkSubmittalDate.js

INCLUDES\_CUSTOM/closeActiveTask.js

INCLUDES\_CUSTOM/getArch\_ENG\_Email.js

INCLUDES\_CUSTOM/getPrimaryEmail4PlanReviewWATUA.js

INCLUDES\_CUSTOM/isActiveAdhocTask.js

INCLUDES\_CUSTOM/isAdhocTaskActive.js

INCLUDES\_CUSTOM/unpaidIssFeesDue.js

SCRIPTS/BATCH/ADD\_MULTIPLE\_PARCELS2CAP.js

SCRIPTS/BATCH/BTR\_NIGHTLY\_BATCH4.js

SCRIPTS/BATCH/OCL\_LATE\_CASES.js

SCRIPTS/BATCH/REG\_RENEW\_START.js

SCRIPTS/EVENT/ASA;BUILDING!~!~!~.js

SCRIPTS/EVENT/ASA;PLANNING!~!~!~.js

SCRIPTS/EVENT/ASA;~!~!~!~.js

SCRIPTS/EVENT/ASIUA;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/ASIUA;BUILDING!~!~!~.js

SCRIPTS/EVENT/ASIUA;PLANNING!~!~!~.js

SCRIPTS/EVENT/CTRCA;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/CTRCA;BUILDING!~!~!~.js

SCRIPTS/EVENT/DUA;BUILDING!~!~!~.js

SCRIPTS/EVENT/DUA;PLANNING!~!~!~.js

SCRIPTS/EVENT/FAA;~!~!~!~.js

SCRIPTS/EVENT/IRSA;BUILDING!~!~!~.js

SCRIPTS/EVENT/ISA;BUILDING!~!~!~.js

SCRIPTS/EVENT/ISB;BUILDING!~!~!~.js

SCRIPTS/EVENT/WATAB;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/WATUA;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/WATUA;BUILDING!~!~!~.js

SCRIPTS/EVENT/WATUB;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/WTUA;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/WTUB;BUILDING!CONSTRUCTION PERMIT!~!~.js

SCRIPTS/EVENT/WTUB;BUSINESSTAX!RECEIPT!~!~.js

SCRIPTS/MASTERSCRIPTS/INCLUDES\_CUSTOM\_GLOBALS.js

SCRIPTS/PAGEFLOW/ACA\_EXP\_LIC\_MSG.js

* + Ultimately you will want to test the submission of every record type and its path through workflow along with any inspections.
  + Added else statement to countHistoryItem4Task calls should be tested for accuracy
    - WATAA:BUILDING/\*/\*/\*
    - WATAA:BUILDING/FIRE/\*/\*
    - WATUA:BUILDING/\*/\*/\*
    - WATUA:BUILDING/CONSTRUCTION PERMIT/\*/\*
    - WTUA:BUILDING/CONSTRUCTION PERMIT/\*/\*

# test issues – additional discovery

The following information contain any specific abnormalities observed during spot-checking the events for proper execution. Not every specific branch was fully evaluated but issues with the branches tested have been noted here.

## ApplicationConditionAddBefore

No business rules – Event executes without error

## ApplicationConditionUpdateBefore

No business rules – Event executes without error

## ApplicationSpecificInfoUpdateAfter

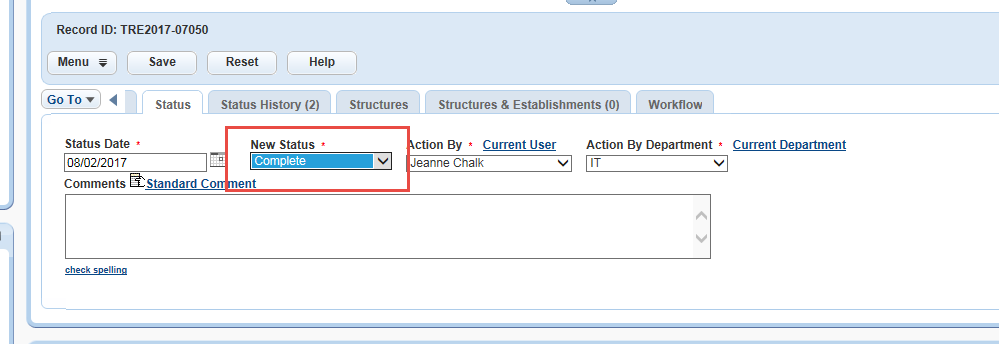
addStdCondition was slightly modified - INCLUDES\_ACCELA\_FUNCTIONS version caused errors, restored customized version for proper function.

## ApplicationSpecificInfoUpdateBefore

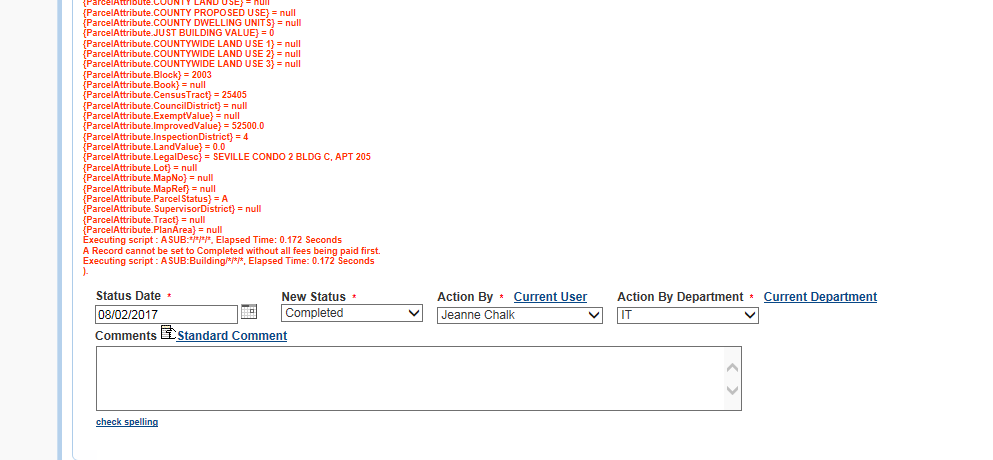
No business rules – Event executes without error

## ApplicationStatusUpdateBefore

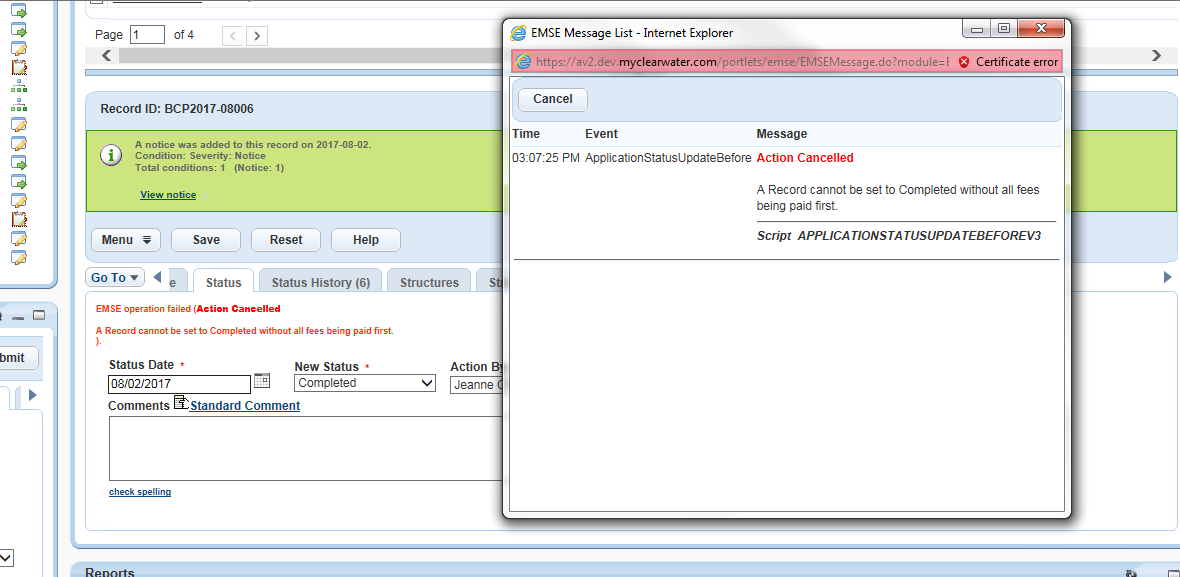
Note: ASUB:\*/\*/\*/\* (general rule for all record types) formerly ApplicationStatusUpdateBefore line 6 seeks to cancel updating the cap status to “Completed” if there is a balance due on the record. Note that not all record types that may be carrying a balance have a “Completed”. For example Building/Enforcement/Housing/HOU, Planning/Miscellaneous/MIS/NA, Planning/Temporary Use/TMP/NA, and Planning/Tree/TRE/NA all have “*Complete*” in their status groups (without the ‘d’). Many others have no status of Completed or Complete. This rule will only catch those record types with a Completed status in their Status group. Consider examining script, status groups, and/or specific record scoping criteria.



This master script and event behaves somewhat differently from the others (If Cancel = true, the script event code is -1 where as most other events throw a 1). This is critical to stopping the commit of the status and other information. Also note that this will put the full contents of the debug into the on-screen event message for users where showDebug = true but will appear normally for end users without debug on.

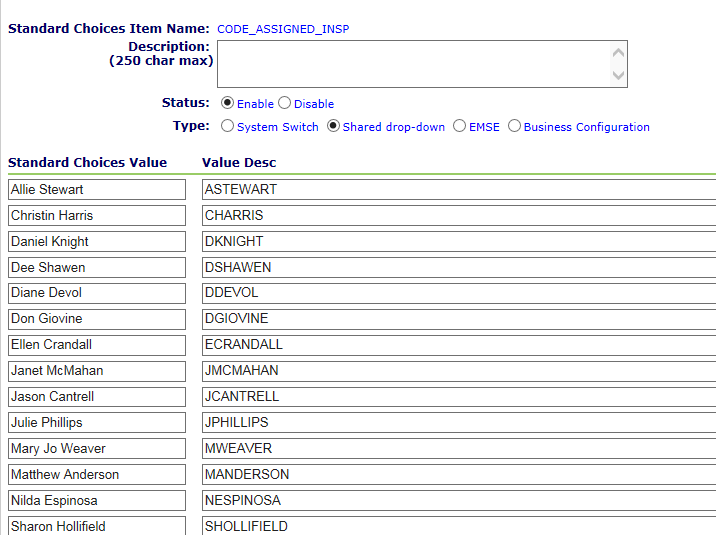
User with Debug On (system admins listed in the INCLUDES\_CUSTOM\_GLOBALS script):

Casual User with debug off:



## ApplicationSubmitAfter

Noted inconsistency in Unsafe Building record and potentially others. Inspector Danny Reep and ***many*** others in Assigned Inspector list have no corresponding entry in the CODE\_ASSIGNED\_INSP standard choice. Line 3 of ApplicationSubmitAfter, now ASA:\*/\*/\*/\* appears to be seeking parity between the two lists in order to properly auto-assign the Initial Inspection. Review Configuration of that dropdown list and this standard choice to improve performance.



## ApplicationSubmitBefore

No business rules – Event executes without error

## ContactAddAfter

No business rules – Event executes without error

## ContactEditAfter

No business rules – Event executes without error

## ConvertToRealCapAfter

## ConvertToRealCapBefore

## DOCUMENTUPLOADAFTER

## DocumentUploadBefore

## FeeAssessAfter

## InspectionResultSubmitAfter

## InspectionScheduleAfter

## InspectionScheduleBefore

## PaymentReceiveAfter

## WorkflowAdhocTaskAddAfter

## WorkflowAdhocTaskAddBefore

## WORKFLOWADHOCTASKUPDATEAFTER

## WorkflowAdhocTaskUpdateBefore

## WorkflowTaskUpdateAfter

## WorkflowTaskUpdateBefore

## Batch Testing

No changes warrant testing

Best Practice is to avoid using local copies of functions and referring to the “INCLUDES” files where appropriate. Consider revision in the future to fetch the INCLUDES files as the master scripts do.

## Pageflow testing

No changes warrant testing – Pageflow script that was referencing standard choice does not appear to be in use

# UAT Issues

The following list contains the issues and outcomes discovered during User Acceptance Testing

Please use the SharePoint site provided to log issues. <https://portal.accelaops.com/projects/CI_Clearwater/Lists/Issue%20List/AllItems.aspx>