Few light years away but bright and shiny STARS ☺

NewCo - Claims QA Test Strategy

Definition of Ready & Done

Program phases	Sprints			Stabilization phase		Deploy
Testing Phase	Dev Testing	Sprints	System	E2E	UAT	Production

Definition of Ready

It is a working agreement between the team and product owner on what readiness means. It is an input criteria to test a story in a sprint/system. It is a way for a SDET/QA to indicate an item is ready to work in sprint and system testing

Expectations:

- **Quality is everybody's accountability**
- Applicable User stories follow BDD standards to enable In-sprint automation.
- Anything which is not in Rally/User story will not be tested
- Clear and detailed acceptance criteria is documented in Rally
- CI/CD Pipeline should be established and working
- Smoke test script identified with proper access and test data
- Up to 80% test coverage performed in Unit testing
- Sonar Unit Testing criteria has been met: >80 Code Coverage, 0 Blocker and 0 critical open defects
- System testing - E2E Test cases ready and reviewed for execution

Expectations:

- All new development should be completed
- Sprint and System test is 100% completed
- No unresolved critical and high defects
- Test cases are ready with test data and user access

Expectations:

- EQA, UAT and Performance signoff received
- All Critical and High defects are resolved or accepted
- No Critical or High performance defects should be outstanding
- All outstanding data defects should be accepted with path to closure

Definition of Done

Expectations:

- Every task under the User Story has been completed
- Any work created is attached to the User Story so the Product Owner can review it and make sure it meets program expectations

Expectations

- Deployment completed
- Production checkout test cases passed

Rally Quality Best Practice

- Definition of Done
  - Make sure no US are closed with open defects
- Use Test Plan in rally
  - Track Newco work effort under the Newco folder
  - Set up of regression
- User Stories BDD leverage for when applicable
- Agreement on observation – see the observation document
- Regression testing – support will be provided by the System as we move forward and share the test scenarios/ test cases
- Agreement on the testing support from Developers when applicable

Rally – User Story

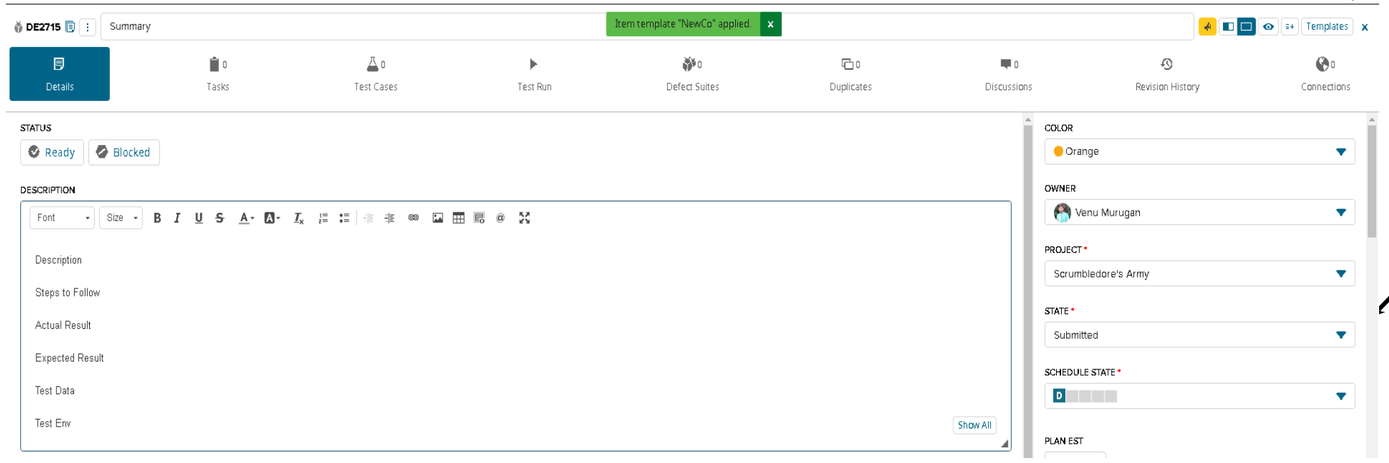
- User Story
  - Applicable User stories follow BDD standards to enable In-sprint automation.
  - All the User stories should be mapped with their respective feature
  - User Story assigned to the sprint must be the Definition Of Ready
    - This includes observation tasks being closed or converted to defects/ US when the team agrees
- Observations
  - Follow the observation path outlined in the attached documentation
  - Discussions for observations can be tracked in the discussion section
  - Email notification and ownership for observation tasks can be used for notification
  - Attach all supporting documents for the observation to the task
  - Observation tasks can be viewed in a report and filter on to provided a centralized way of the efforts.
  - Observation task converted to defect/ user stories must have the defect number logged in the notes

Microsoft Word Document

- Defects
  - Defects will be reviewed with the Claims Newco team and evaluated internal prior to tagging it for External Newco Support
  - Defects escalated can be viewed in a report and if support or additional information is needed plan to attend the external defect triage meeting.
- All Regression Test Scenarios will be automated and merged into the regression suite for support

Rally Quality - Defect

- All the defect will be traced under defect module
- Each defect should be mapped to a particular User Story
- Each defect should be mapped to its respective Test case

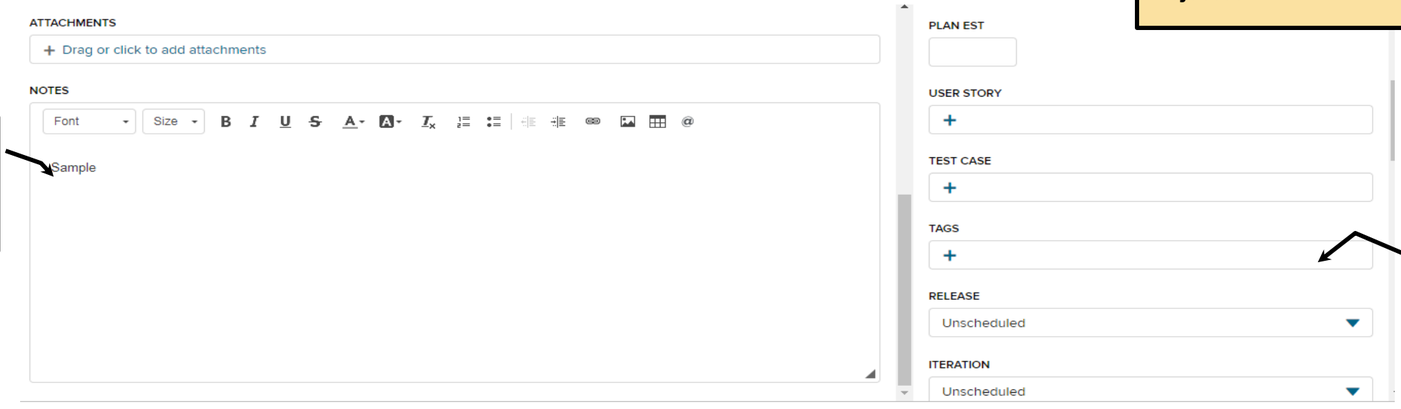


A predefined defect template will be used for the consistency and to prepopulate the Description

Other mandatory fields for defect reporting

- Owner
- Schedule State
- Environment
- User Story
- Test Case
- Iteration
- Resolution
- Severity
- Priority

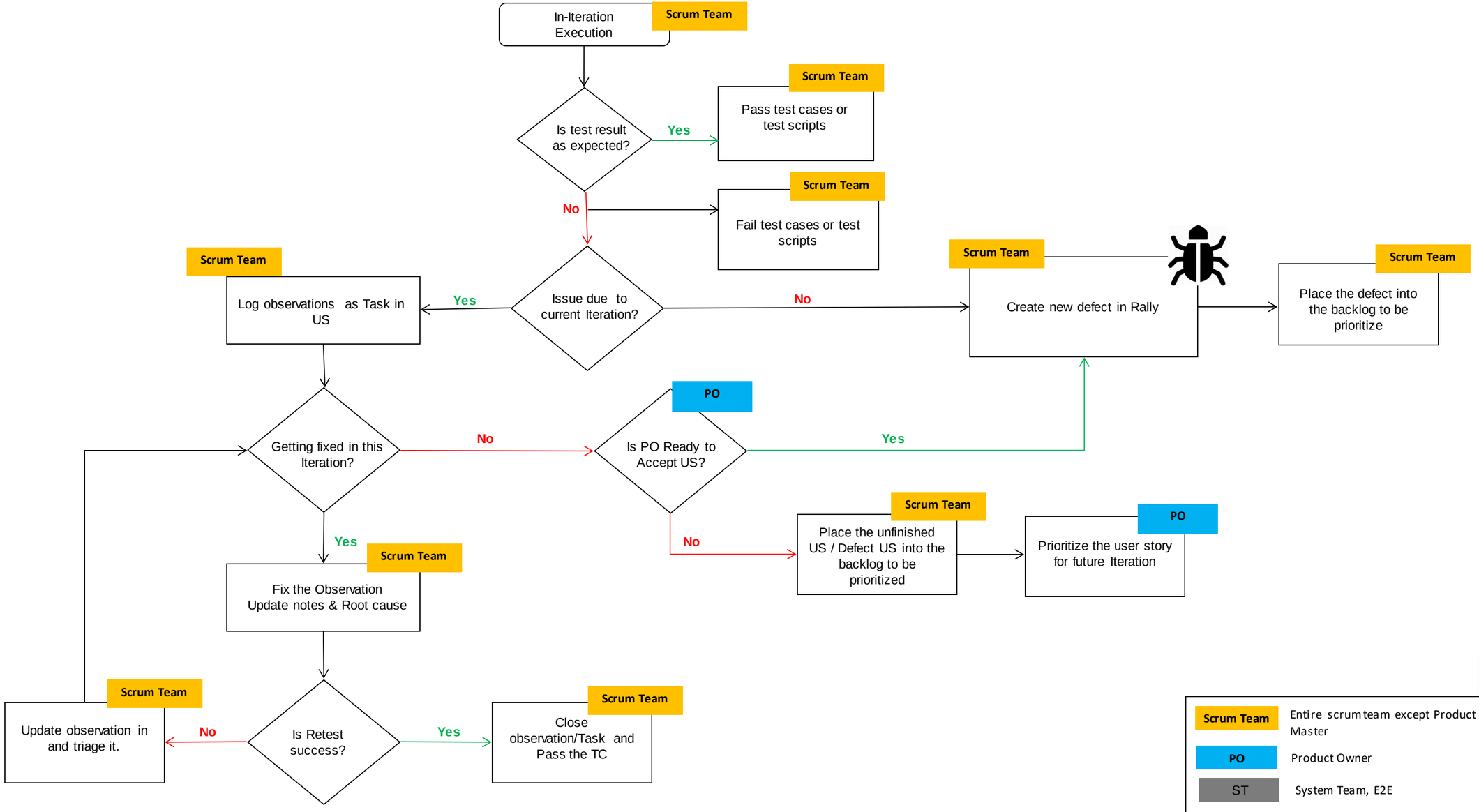
Below are the different defect statuses
Submitted – defect was initially created
Open –defect work is in progress
Fixed –code related issue and ready for re-testing
Closed
Duplicate
Rejected



Notes section for capturing defect communications and root cause

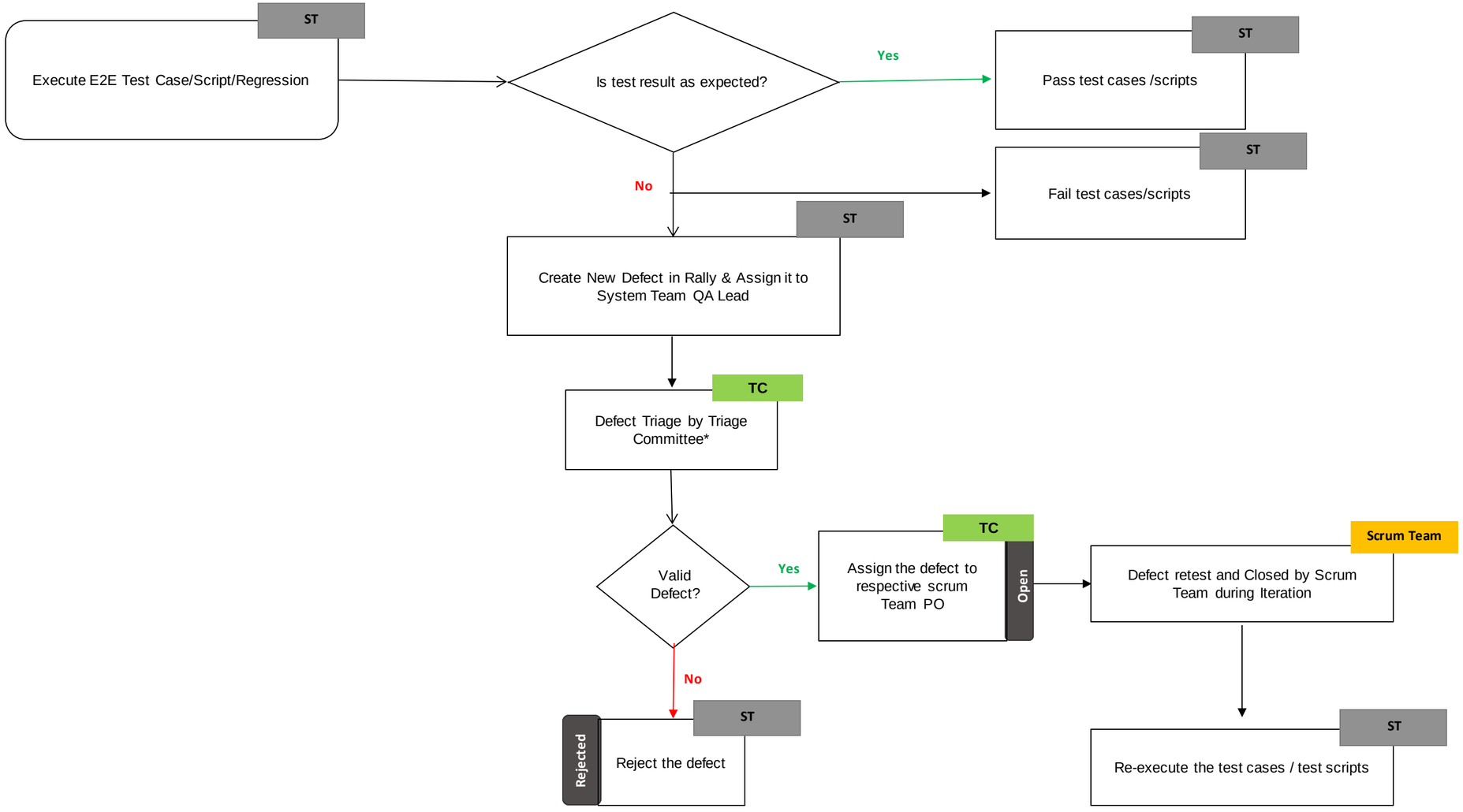
Tag for defect
Newco – this will allow us to identify Newco related
External Support - when the defect requires support from the Newco teams to resolve

Observation/ Defect Management - Sprint Execution Phase



All the issues related to scope within the current Iteration should be logged as "Task in US" and should neither be considered as defects nor be accounted in any of the metrics.

Defect Management - System Test Execution Phase (Regression)



Legend

TC	Triage Committee- Product Owner, TLL, Bus. Arch, SME's, System QA Lead

Defect Severity & Priority

Severity -

Severity	Definitions
Critical	The defect causes system level errors that prevent test execution. For example, server is not available, application is throwing error. Fata defects must be fixed immediately so the testing can continue
High	The defect adversely affects data consistency or high priority application functionality as listed in the business requirement and has no workaround. For example, Not able to create Event, Exposure. High defects will be prioritized and responded to after Critical defects such that testing can continue
Medium	The defect does not affect major functionalities, but only affects less important area of the application functionality. For example, Invalid Coverage limits are prefilled, Aggregate and CSL functionality not working as expected. Medium defect may or may not be fixed before the software is released
Low	The defect is superficial and has minimal impact on the user experience. For example, cosmetic changes such as small display issue or non critical information, poor wording or error message. Low defect may or may not be fixed before the software is released

Priority –

Priority	Definitions
High	High priority indicates that these defect should be fixed first within the severity category
Medium	Medium priority indicates that these defect should only be fixed after the high priority defects have been fixed within the severity
Low	Low priority indicates that these defect should be fixed last

Rally Quality – Test Plan

The **Test Plan Module** in Rally provides visibility into the Test Scenarios being executed by the team. It also allows the team to view the status for each Test Scenario with in the User Story. The Basic folder structure is outline below

Level 1 Test Folder (TF) – Year_Team Name
- **Level 2 Test Folder (TF)** - Release month_ Year
  - **Level 3 Test Folder (TF)** - User Story number 1
    - **Level 1 Test Case (TC)** - TC number _ User story Feature ?
    - **Level 1 Test Case (TC)** - TC number _ User story Feature ?
  - **Level 3 Test Folder (TF)** - User Story number 2
    - **Level 1 Test Case (TC)** - TC number _ User story Feature ?

Mandatory fields for Test cases

- Owner
- Type
- Method
- Priority
- Verdict
- Associated US
- Result attachment

In addition to the Basic Test Plan Folder Structure the ***NewCo effort*** will have additional folders

Level 1 Test Folder (TF) – Year_Team Name
- **Level 2 Test Folder (TF)** - Release month_ Year
  - **Level 3 Test Folder (TF)** - User Story number 1
    - **Level 1 Test Case (TC)** - TC number _ User story Feature ?
    - **Level 1 Test Case (TC)** - TC number _ User story Feature ?
  - **Level 3 Test Folder (TF)** - User Story number 2
    - **Level 1 Test Case (TC)** - TC number _ User story Feature ?

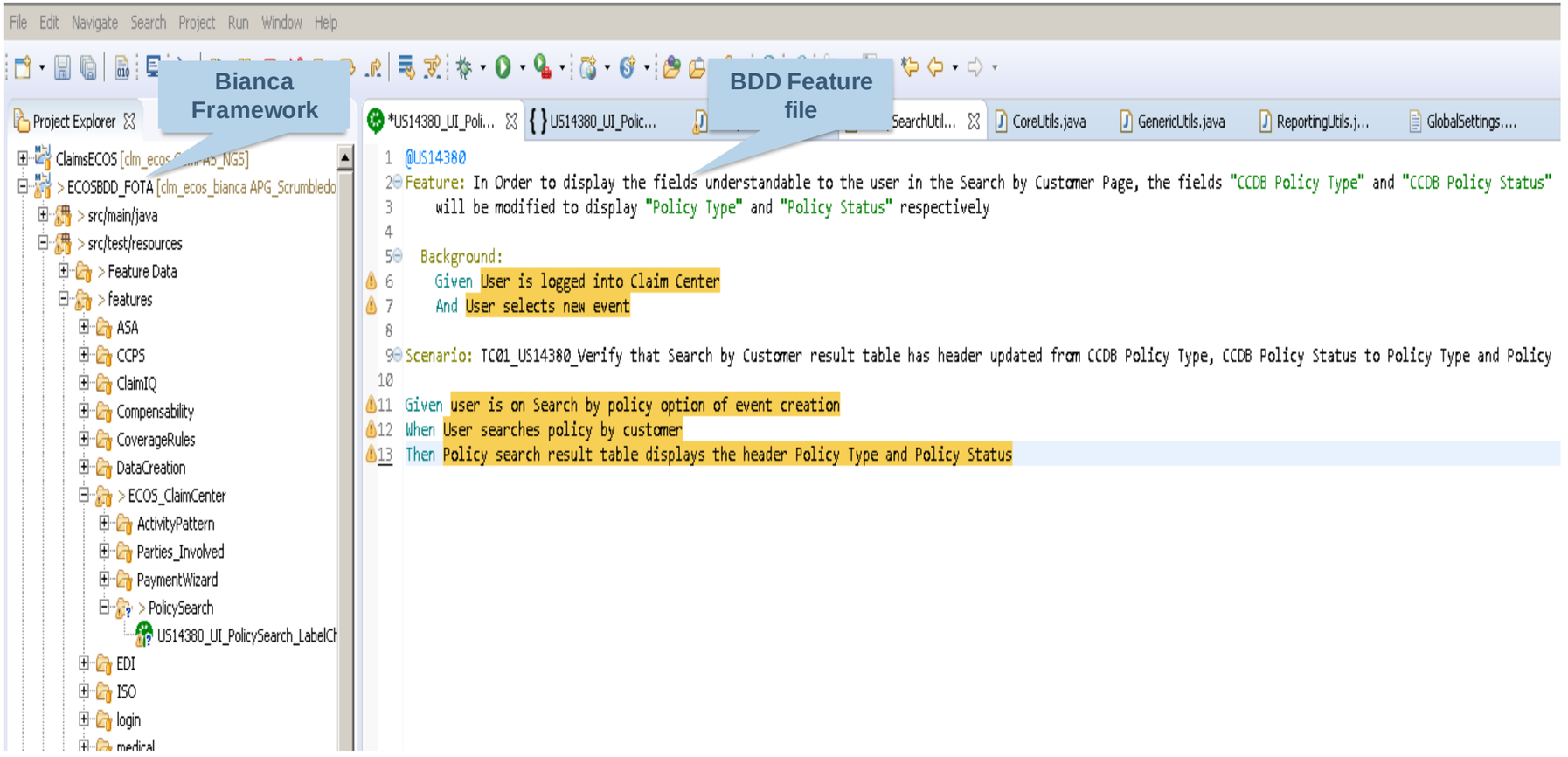
Level 1 Test Folder (TF) – 2020_Newco
- **Level 2 Test Folder (TF)** – Program Increment 1
- **Level 3 Test Folder (TF) –** Iteration 1.1
- **Level 4 Test Folder (TF) –** User Story number 1
  - **Level 1 Test Case (TC)** - TC number _ User story Feature ?

Test Automation Framework for NewCo

Automation Framework - Bianca

- **Automation Framework –** BDD Bianca framework
- **Leverage BDD for automation design** – Leveraging Hartford standard framework (Bianca) along with BDD for automation design.

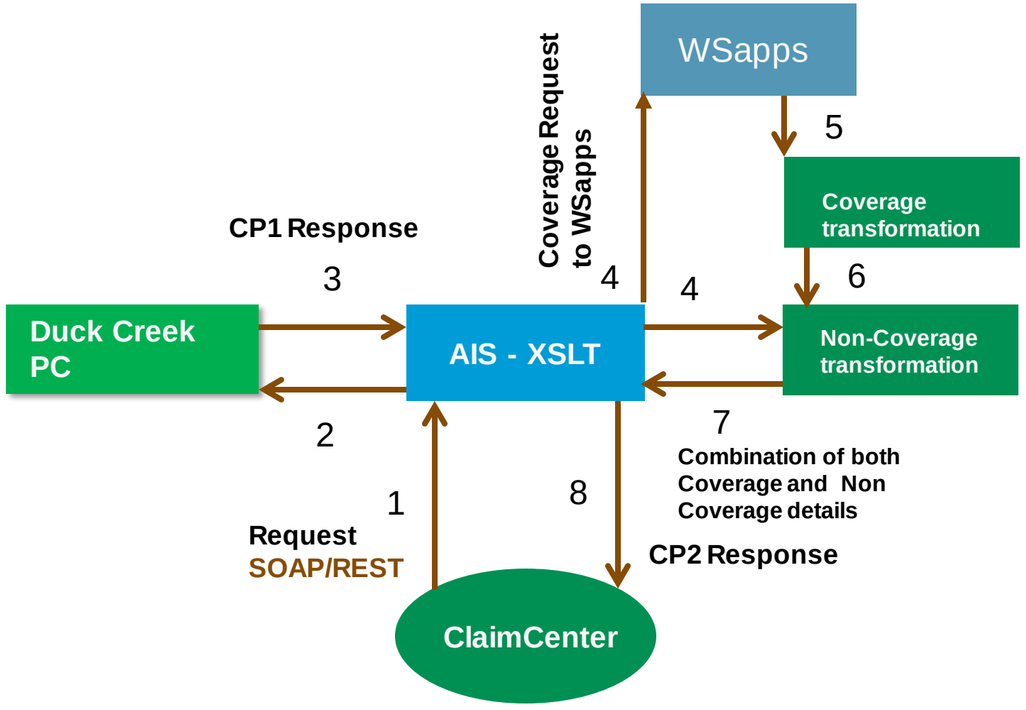
Bianca framework + BDD



Claims: Testing Strategy

Claims policy Service Validation

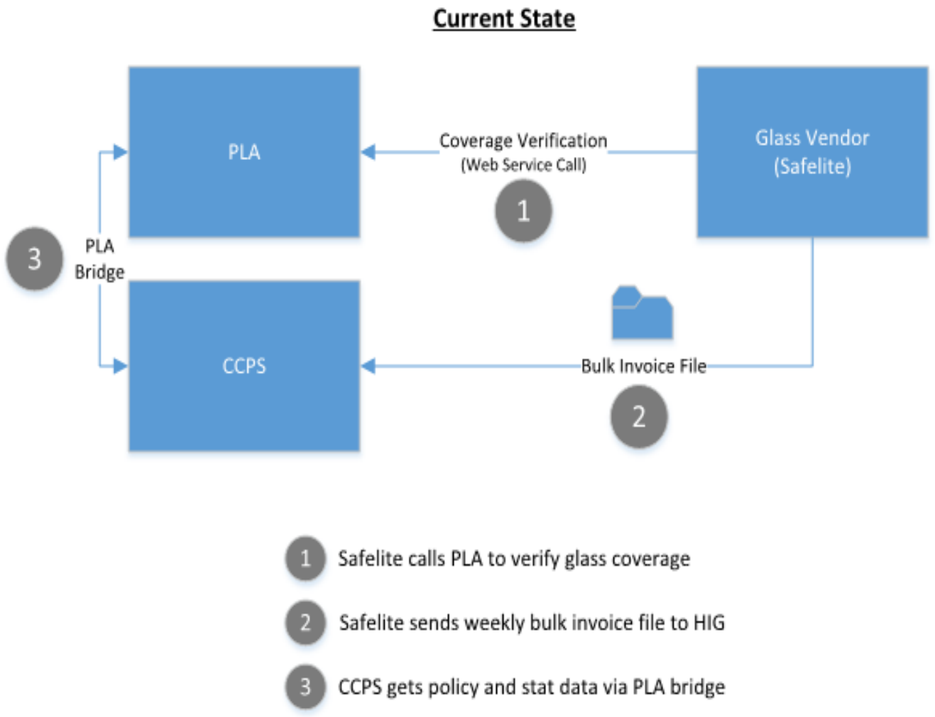
- Validate Policy coverage and Non- Coverage details in CP1 response from Duck Creek to ECOS
- Validate Policy prefill CP2 request / response in ECOS after Coverage, Non- Coverage, XSLT and WSapps changes
- Validate Policy Retrieval response for the below policy types
  - NewCo Auto
  - NewCo Home
  - NewCo Liability
  - NewCo Umbrella
- Validate Coverage transformation changes from Wsapps layer



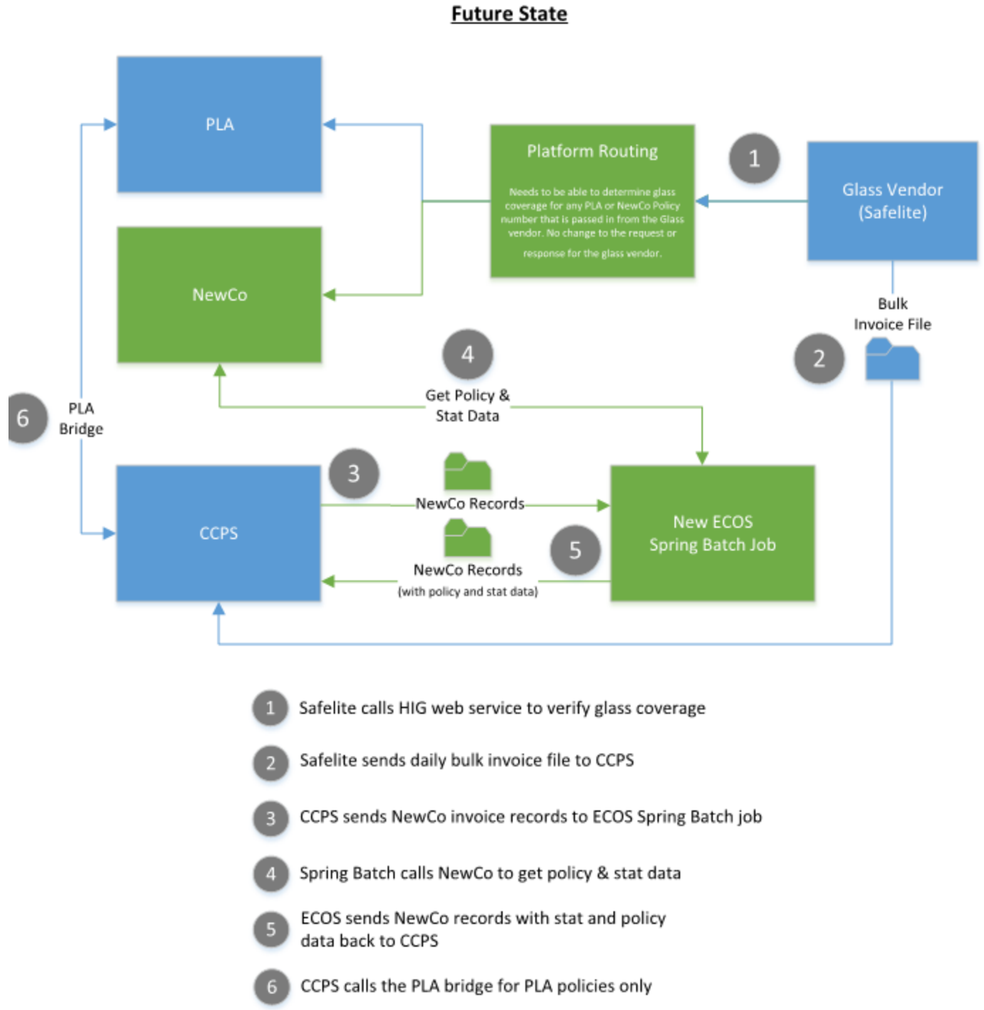
Claims Service Test Considerations

- Services/WSDL/XML
- Connectivity
- Authentication / Security
- Transformation Logic
- Data Combination (Valid/Invalid)

Glass vendor – (Safelite)



CCPS works on mainframe technology. So it's unable to connect with DC system in cloud. Because of that reason ECOS will act as an intermediate system to share policy and stat details for NewCo policies to CCPS from DC .



Testing

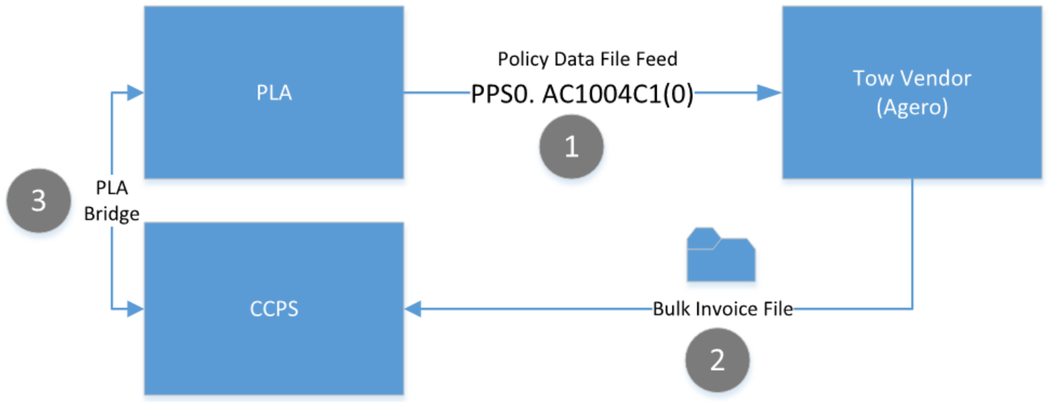
Validated ECOS to CCPS outbound file layout as per expected layout designed by CCPS

Completed field level validation for all fields that are generated in outbound file based on invoice file

Tow vendor – (Agero)

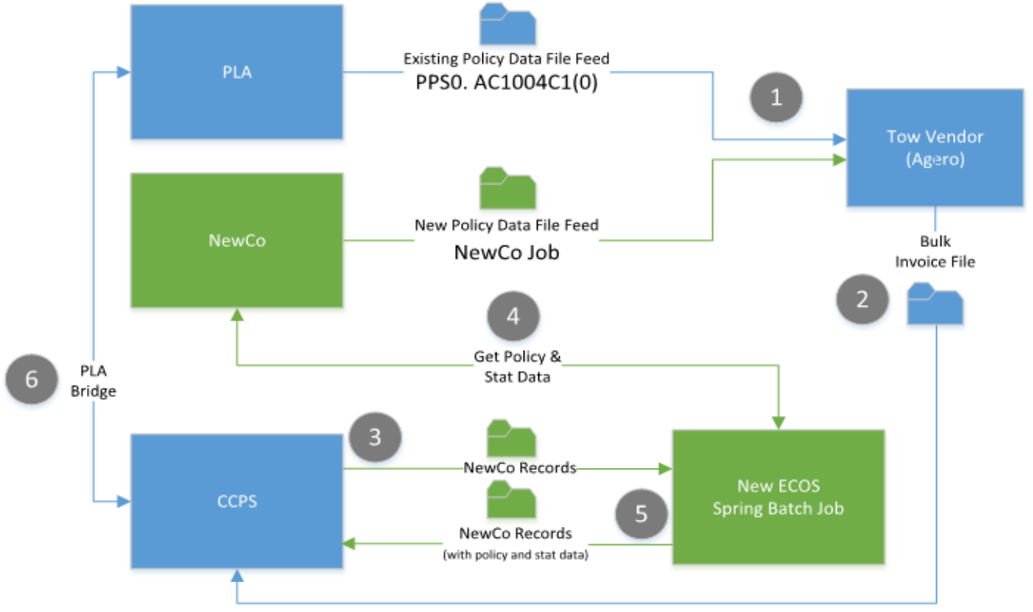
CCPS works on mainframe technology. So it's unable to connect with DC system in cloud. Solution - ECOS will act as an intermediate system to share policy and stat details for NewCo policies to CCPS from DC .

Current State



1. PLA sends policy data file to Agero
2. Agero sends monthly bulk invoice file to HIG
3. CCPS gets policy and stat data via PLA bridge

Future State



1. PLA and NewCo send policy data files to Agero
2. Agero sends monthly bulk invoice file to HIG
3. CCPS sends NewCo invoice records to ECOS Spring Batch job
4. Spring Batch calls NewCo to get policy & stat data
5. ECOS sends NewCo records with stat and policy data back to CCPS
6. CCPS calls the PLA bridge for PLA policies only

Claims – PI 1.0 Features (1/29/2020 – 4/07/2020)

Feature ID	Name	In Scope for E2E	Scope
F1353	ECOS: Search by Policy (NewCo Policy Number)	Yes	To initiate a Search from ECOS using policy number that triggers a real-time call to the NewCo and PLA to retrieve key information to allow the user to identify the correct policy within the claim
F1365	NewCo Glass Claims Management (through CCPS)	No	
F1637	NewCo Tow Claims Management (Invoice process through CCPS)	No	
F1661	Policy Retrieval of Duck creek OOTB Non-Coverage Fields	Yes	Prefilling Duck Creek OOTB non-coverage policy data in ECOS for NewCo Personal Auto
F1773	Enabler: Manual Entry of Coverage Terms for Auto	No	
F1609	Enabler: Roadside Assistance (Tow) - Capping Number of Claims	No	

Claims – PI 2.0 Features (04/08/2020 – 06/16/2020)

Feature ID	Name	In Scope for E2E	Scope
F1610	Enabler: Analysis and Design for STAT Code, both "Glass & Tow" and ECOS initiated claims	No	
F1817	Enabler: Analysis and Design for Risk Feed	Yes	ECOS to downstream system
F1352	ECOS: Search by Customer for FNOL (NewCo Policy)	No	
F1819	NewCo Glass & Tow Claims Management (through CCPS) Integration Testing with Duck Creek - STAT Virtual Service	No	
F1818	Enabler: Mapping Auto coverages for prefill from Duck Creek for OOTB retrieve	No	
F1380	CCPS Stat Coding (all ECOS initiated claim, not exclusively glass and tow)	Yes	
F1360	Manual Entry of Coverage Data in ECOS for NewCo Auto Policies - Coverage Terms and Less frequently used Coverages	No	Manual adding of coverage's and coverage terms
F1830	NewCo Glass Claims & Tow Management (through CCPS) Policy Integration and Testing with Duck Creek	Yes	

Claims policy Service Testing Conditions

Claims Policy Service Validation

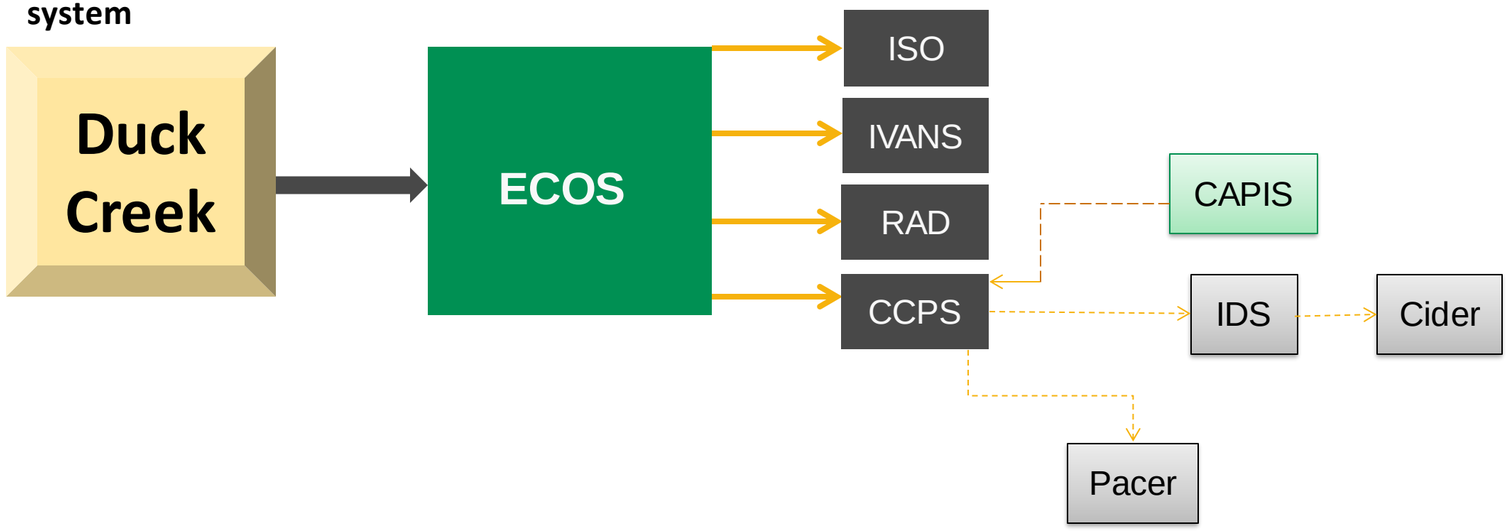
- Verify if end points of the service is **accessible**
- Data Driven Testing Conditions :
  - **Valid / Invalid Parameters in both request/response**
  - **Varying Data combinations (parameters)**
  - **Varying the elements of the service request**
- Dependency Validation
  - Validate any **dependency** for the service to be invoked.
  - Check if the service **communicates** with any other external web service for availability

Negative Testing

- Validate response by **removing part of elements** / fields in the request.
- Validate response by **adding an invalid header tag** in request
- Identify any areas prone to **user errors in UI** and validate response if such requests gets invoked
- Validate **Null** response

Duck Creek – ECOS Integration

Personal Lines Duck Creek platform will be integrating with ECOS (Guidewire CC) to handle claim lifecycle for the policy created in duck creek system



Application	Function
ECOS	Enterprise Claims operating system – A Guidewire Claimcenter product that will handle E2E claim life cycle activities
ISO	ISO (Insurance Services Office), a provider of premium and loss data for the property-casualty insurance industry, has created a single repository for all claim data, called the ISO ClaimSearch
IVANS	Interface to seamlessly extract and download Claims information from ECOS directly into Agents' Management Systems
RAD	Receipts and Disbursements will take of payment processing and vendor and TIN management
CCPS	CCPS is a financial interface which captures claim demographics and financial data such as reserves, payments, recoveries etc.

ECOS UI Test Scenarios

ECOS (CC):

NewCo team/Claims QA team will create policies with required coverage and Non- Coverage details

- ✓ Validate that CP1 XML from DC to ECOS will have the required coverage code and its values
- ✓ Validate that Coverages from DC are transformed in XSLT and WSapps layer and that are available in CP2 XML
- ✓ Verify that Policy details are displayed in Policy search page, by searching with valid policy number, LOB, Loss date
- ✓ Verify whether all the policy details are prefilled in ECOS by calling policy retrieval service
- ✓ Validate whether event is created successfully with the prefilled NewCo policy
- ✓ Validate whether Coverages that are available in that policy are prefilled in ECOS based on transformation rules
- ✓ Validate whether Non coverage field values are prefilled in ECOS fields as per transformation logic
- ✓ Locations, Vehicle and Property details available in the policy should be prefilled in ECOS
- ✓ Endorsement Form number and its description should be prefilled in ECOS for the coverages available in that policy
- ✓ Exposure should be created for the coverage available in that policy
- ✓ Validate whether user is able to set reserve for all cost category associated with that coverage
- ✓ Validate whether Claim symbol is generated as per requirement
- ✓ Validate whether payment is posted for the reserve created
- ✓ Validate the monetary types added to the line category of the coverage
- ✓ Validate limit overage validation for Per Person, Per Occurrence, CSL and Aggregate limit
- ✓ Validate the CSL and Aggregate codes are prefilled as expected
- ✓ Validate the Invalid Coverage for cause associated with the coverage in Administration
- ✓ Validate whether Coverage term associated with the coverage are prefilled as per transformation rules

ECOS – ISO Test Scenarios

ISO – Insurance Standard office

- Validate whether request has been send to ISO by successful creation of exposure with required details
- Validate ISO Policy Type code in request XML
- Validate ISO coverage type code and ISO loss type code with respect to the exposure of that coverage
- Validate whether upon successful submission of ISO Initial or ISO Replace, an immediate receipt will be returned to the ECOS
- Upon submission of claim to the ISO ClaimSearch system, the system searches the database and finds claims similar to the ones submitted by ECOS and validate that an response XML file and ISOMatchReport will get generated in ECOS NAS 1019 folder
- Validate whether Vehicle recovery report is generated when vehicle theft claim is submitted

ECOS – IVANS Test Scenarios

ECOS outbound to IVANS:

- Validate whether below updates are send from ECOS to IVANS after successful batch processing
  - ✓ Claim information update (CLI)
  - ✓ Payment Information (PMT)
  - ✓ Reserve Update (RES)
  - ✓ Adjuster Assignment (ADJ)
  - ✓ Notes
  - ✓ Claim Status update (CLS)
- Validate that ECOS does not send details to IVANS for un verified Policy
- Validate IVANS Coverage code and IVANS loss cause code
- Validate that if the policy producer code is not subscribed in IVANS, ECOS does not send details to it
- Validate whether Sensitive claim details are not send from ECOS to IVANS
- Validate that only approved reserve details are send to IVANS

ECOS – RAD Test Scenarios

ECOS outbound to RAD:

- Post Payment for the line category created against NewCo Coverage exposure and validate whether payment is in Requesting Status
- Validate whether Payments posted in ECOS are successfully processed in RAD and acknowledgement is send to ECOS
- Validate whether Payment Status is updated to issued by successful processing of RAD Finsynz job
- Post payment for Claimant, Insured, Vendor and validate whether it is processed successfully
- Post different type of payment Check, EFT, Manual and recurring payment and validate whether it is successfully processed in RAD and acknowledged to ECOS

ECOS – CCPS Test Scenarios

Corporate Claims Processing System:

- Validate whether claim details are successfully reached CCPS after successful completion of CCPS outbound job, once reserve is set on the claim
- The below listed items should be presented in CCPS outbound file generated in 1005 NAS path folder
  - ✓ Exposure number
  - ✓ Coverage Code
  - ✓ Transaction Code
  - ✓ Policy details
  - ✓ CDC Code
- After successful processing of claim in CCPS that was created in ECOS, the below details are to be validated in CFM print
  - ✓ Exposure number
  - ✓ Coverage Code
  - ✓ Transaction Code
  - ✓ Policy details
  - ✓ CDC Code
  - ✓ Reserve details
  - ✓ Payment details
  - ✓ Stat ID
  - ✓ WCC
  - ✓ MLC
  - ✓ Segment 3 details
- Validate the change transaction(CC23) details that are send from ECOS to CCPS by modifying any claim details