Lesson 4 Configuring Policy Transactions After Issuance

4.1 Configure a new return premium allocation

When BillingCenter receives a credit, Succeed Insurance wants to allocate the credit to the same policy period as the credit charge. Any remaining funds should be distributed first to last.

4.1.1 Requirements

Spec 1 Credits should be allocated to the same policy period first to last.

Spec 2 Remaining funds should be distributed to other policy periods first to last.

4.1.2 Tasks

1. **Add a typecode to the ReturnPremiumAllocateMethod typelist.**
   a) Extend the ReturnPremiumAllocateMethod.tti typelist.
   b) Suggested typecode name: PolicyPeriodFirst_Ext
   c) Do not restart the server until all configuration is done, otherwise, the system will throw an exception.
2. **Create a new class based on ProportionalReturnPremiumAllocationStrategy.gs.**
   a) Suggested package name: si.bc.classes.creditallocation
   b) Suggested class name: PolicyPeriodFirstReturnPremiumAllocationStrategy_Ext
3. **Register the new class in LinkedImplementationLoaderImpl.gs**
4. **Restart the server.**

4.1.3 Testing procedure

1. **Create a new Return Premium Plan called RPP CH Lab**
   a) Clone the Default Return Premium Plan
   b) Set the name field to RPP CH Lab
   c) Change the Eligible Items dropdown to Same Payer
   d) Add a Policy Change context, select Policy Period First for the method, and move it to top priority
   e) Click the Update button to save the new plan
   f) Navigate to the Return Premium Plans (Administration → Return Premium Plans) and move the RPP CH Lab plan to top priority

2. **Create an account.**
    a) QuickJump: Run Account
3. **Add a policy to the account with these details:**
    a) Actions → Add Policy
    b) Policy Number: CH-A
    c) Payment Plan: PP04
    d) Premium: $600
4. **Add a second policy to the new account with these details:**
    a) Actions → Add Policy
    b) Policy Number: CH-B
    c) Payment Plan: PP04
    d) Premium: $600
5. **Make a Direct Bill Payment and override the allocation to include any invoice.**
    a) Actions → New Payment → New Direct Bill Payment
    b) Amount: $360
    c) Click Override Distribution
    d) Include Only: Up to amount under contract
    e) Click Execute
6. **Look at the Charges screen. Note that the payment was applied to both policies.**
7. **Navigate to CH-A policy and create a policy change with a $300 credit**
    a) Actions → Change Policy
    b) No changes are needed in step 1 of the Policy Change Wizard, so just click Next
    c) On step 2, click the add button, select Premium as the type, and enter -300 in the amount field
    d) Click Finish to complete the wizard
8. **Navigate to the account and look at the Charges screen. Credit should only be allocated to policy CH-A First to Last.**
    a) Make sure CH-B did not receive any credit.
9. **Navigate to CH-A policy and create a policy change with a $200 credit**
    a) Actions → Change Policy
    b) No changes are needed in step 1 of the Policy Change Wizard, so just click Next
    c) On step 2, click the add button, select Premium as the type, and enter -200 in the amount field
    d) Click Finish to complete the wizard

10. **Navigate to the account and look at the Charges screen. CH-A should be paid in full and the remaining $80 credit to policy CH-B First to Last.**

4.1.4 Solution

1. **Add a typecode to the ReturnPremiumAllocateMethod typelist.**
   a) Extend the ReturnPremiumAllocateMethod.tti typelist.
   b) Suggested typecode name: PolicyPeriodFirst_Ext
   c) Do not restart the server until all configuration is done, otherwise, the system will throw an exception.
2. **Create a new class based on ProportionalReturnPremiumAllocationStrategy.gs.**
   a) Suggested package name: si.bc.classes.creditallocation
   b) Suggested class name: PolicyPeriodFirstReturnPremiumAllocationStrategy_Ext

```
       package si.bc.classes.creditallocation

uses gw.api.web.payment.AllocationPool
uses gw.api.web.payment.ReturnPremiumAllocationStrategy
uses gw.payment.FirstToLastAllocationStrategy

/**
 * Credit Allocation - Configure a new Return Premium Allocation lab
 */
class PolicyPeriodFirstReturnPremiumAllocationStrategy_Ext implements ReturnPremiumAllocationStrategy{

  override function allocate(list: List<BaseDistItem>, allocationPool: AllocationPool) {
    // determine if any positive distribution items exist
    var posDistItems = list.where(\item -> item.InvoiceItem.Amount.IsPositive)
    if(posDistItems.Empty) {
      return
    }
    // determine if any negative distribution items exist
    var negDistItems = list.where(\item -> item.InvoiceItem.Amount.IsNegative)
    if(negDistItems.Empty) {
      return
    }

    // set up variables
    var negPolicyPeriod = negDistItems[0].PolicyPeriod
    var currency = allocationPool.Currency
    var amountAvailable = allocationPool.GrossAmount
    // Get all positive distribution items that have the same policy period as a negative endorsement
    var preferredPosDistItems = posDistItems.where(\item -> item.PolicyPeriod == negPolicyPeriod)
    // Get remaining positive distribution items
    var remainingPosDistItems = posDistItems.where(\item -> item.PolicyPeriod != negPolicyPeriod)

    // allocate credit to preferred positive distribution items if some exist
    if(preferredPosDistItems.HasElements) {
      // the allocate method will only use the portion needed
      var preferredAmount = preferredPosDistItems.sum(currency, \item -> item.GrossAmountOwed)
      new FirstToLastAllocationStrategy().allocate(preferredPosDistItems,
AllocationPool.withGross(amountAvailable))
      amountAvailable = amountAvailable - preferredAmount

    }
    // allocate credit to remaining distribution items if some exist and if money remains
    if(amountAvailable.IsPositive and remainingPosDistItems.HasElements) {
```

```
      new FirstToLastAllocationStrategy().allocate(remainingPosDistItems,
AllocationPool.withGross(amountAvailable))
    }
  }

  override property get TypeKey(): ReturnPremiumAllocateMethod {
    return ReturnPremiumAllocateMethod.TC_POLICYPERIODFIRST_EXT
  }
}
```

3. Register the new class in LinkedImplementationLoaderImpl.gs

```
         override function returnPremiumAllocationStrategies() :
Collection<ReturnPremiumAllocationStrategy> {
            return {
                new FirstToLastReturnPremiumAllocationStrategy(),
                new LastToFirstReturnPremiumAllocationStrategy(),
                new ProportionalReturnPremiumAllocationStrategy(),
                new PolicyPeriodFirstReturnPremiumAllocationStrategy_Ext()
            }
         }
```

4. Restart the server.