Lesson 9 Configuring Transaction Approval

9.1 Configure transaction approval

Succeed Insurance want to implement aggregate approval limits for account disbursements. For example, suppose there is a user who has the authority to create disbursements of up to $100. The user wants to create a disbursement for $120 without waiting for approval. To circumvent their authority limit, the user creates two $60 disbursements for an account. Both disbursements are below the user's authority limit, so both are automatically approved.

Succeed wants to prevent the user from doing this by enforcing a user's aggregate approval limit for disbursements. If a user has this limit, then the first $60 disbursement would be approved automatically. But the second disbursement would exceed the aggregate approval limit and therefore require approval.

9.1.1 Requirements

Spec 1 Implement aggregate approval limits that limit account disbursements for any given account.

Spec 2 Do not enforce aggregate approval limits for automatic disbursements.

Spec 3 Enforce aggregate approval limits per user

9.1.2 Tasks

1. **Find all AuthorityEvent instances linked to the account associated with the current transaction.**
2. **Total the amounts of the transaction objects linked to the relevant AuthorityEvent instances.**
3. **Retrieve the user's aggregate authority limit.**
4. **Check to see if the total of the previous transactions plus the new transaction is less than or equal to the aggregate authority limit.**

9.1.3 Testing procedure

1. **Add a test policy**
   a) Create a test account by running the Run Account quick jump command
   a) Add a new policy with $240 premium. Fill in any required fields
2. **Fully pay the policy**
   a) Click Actions → New Payment → New Direct Bill Payment
   b) Enter 240 in the amount field
   c) Click the Execute Without Distribution button
3. **Create a policy change to reduce the premium for the policy**
   a) Navigate to the policy

b) Click Actions → Change Policy

c) On step 1 of the wizard, no additional changes are needed so click Next.

d) In step 2, click the add button and enter -120 in the amount field and then click finish.

4. **Verify that Aaron Applegate has an authority limit of $100 for disbursements**

   a) Navigate to Administration → Users & Security → Users

   b) Click on Aaron Applegate

   c) On the Authority Limits tab, confirm that his Approve Disbursement authority limit is set to $100.

5. **Log in as a limited user and access test account**

   a) Log in as Aaron Applegate (aapplegate)

   b) Use the account search to locate the test account

6. **Create the first $60 disbursement**

   a) Click Actions → New Transaction → Disbursement to start the Account Disbursement Wizard.

   b) Enter 60 in the amount field

   c) Select the current billing system date as the payment date

   d) Select Return Premium PolicyChange as the reason

   e) Click next to move to step 2 of the wizard and then click finish

   f) Click on Disbursements in the side panel

   g) Notice that the disbursement is approved

7. **Create the second $60 disbursement**

   a) Click Actions → New Transaction → Disbursement to start the Account Disbursement Wizard.

   b) Enter 60 in the amount field.

   c) Select the current billing system date as the payment date.

   d) Select Return Premium PolicyChange as the reason.

   e) Click next to move to step 2 of the wizard.

   f) Notice that an alert message is displayed at the top of the screen. This message tells the user that disbursement will require approval because this user does not have sufficient authority.

   g) Click finish.

   h) Click on Disbursements in the side panel.

   i) Notice that the second disbursement is awaiting approval.

9.1.4 Solution

1. **Comment out the existing code**

   a) Open the ActivityApprovablePlugin.gs in Studio

b) Comment out the existing version of the isCanApproveDisbursement method

2. Add code to the method as shown below

```
         override function isCanApproveDisbursement(disbursement: Disbursement,
isCurrentUserCanApproveAction: Boolean): Boolean {
            if (disbursement typeis AccountDisbursement) {
              var totalAccountDisbursementAmountForUser = disbursement.Amount
              var disbUtil = new DisbursementUtil()
              foreach (disb in disbUtil.getAccountAndCollateralDisbursements(disbursement.Account)) {
                if(disb typeis AccountDisbursement) {
                  if(disb.AuthorityEvent.CreateUser == User.util.CurrentUser) {
                    totalAccountDisbursementAmountForUser += disb.Amount
                  }
                }
              }
              var max = User.util.CurrentUser.getAuthorityProfile().
                  getMaximumAuthorizedAmount(AuthorityLimitType.TC_APPROVEDISBURSEMENT,
disbursement.Account.getCurrency())
              return totalAccountDisbursementAmountForUser <= max
            }
            return isCurrentUserCanApproveAction
          }
```

3. In Studio, select Run → Debugging Actions → Reload Changed Classes.