Spending Limit Extension Report

Saul van der Vies & Noël Keijzer

September 10th 2017

Total time spent: (combined)

Category 1: 4 hours 20 minutes

- Add methods to calculate if the spending limit will be exceeded for a transaction(both for a card and an account).
  50 minutes
- Update database to support spending limits(add spending limit to ledger table, update existing methods) and add setTransferLimit method to the Ledger Service.
  40 minutes
- Change chains in the system that process transactions to use MessageWrappers.
  20 minutes
- Add setTransferLimit method to Authentication Service
  20 minutes
- Add setTransferLimit method to Api Service
  15 minutes
- Add transferLimit processing to dayProcessing in the SystemInformation Service
  15 minutes
- Add setTransferLimit method to SystemInformation Service
  10 minutes
- Debugging transactions failing/causing timeouts.
  1 hour 20 minutes
- Fixing timeouts occurring in non-transaction requests.
  10 minutes

Category 2: 10 minutes

- Adding new tests to the test suite
  10 minutes

Category 3: negligible

We added a spending limit to the ledger table and added a variable to the ledger service to keep track of the daily spending limit for a debit card. The ledger service was changed such that it checks the weekly spending limit for each transaction that is processed, unless an admin override is used. It also checks the daily spending limit if a debit card is used in the transaction. To set the spending limit a Map was added to the systemInformation Service that maps dates to a list of TransferLimit objects. These are transfer limits that are due to be set on the date of the key. When a simulateTime request is received all transfer limits that have a key that is before or on the current date are then sent to the ledger service, which will then update all the limits. This extension was very straightforward to implement and did not require many changes to the system. The changing of the transaction communication caused some bugs in the system that caused requests to time-out, which took a bit of time to fix. Besides that the testing phase was very straightforward as well.

The fact that our transaction communications did not use MessageWrappers yet kind of worked against us because in the ledger it was not possible to check if a transaction was a transfer or a payment with a card. The way our simulateTime method was set up inside the system made it really easy to add the updating of transfer limits at the end of this method. This meant that not being able to update spending limits when the request was received was not much of a problem as we could just add the to a list and process them at the end of the appropriate simulateTime request.