Contents

1 UPL: Universal Processing Language

1.1 History

Pocessing systems are using manually crafted application relays to handle card processing business rules. Being defined by business analysts these rules are fallen down to engineering teams informally. Approach we provide pushes card processing to solid background in a form of domain specific language common for all card plans analytics departments.

Having compact language we can formally build various translators for particular customers and existing processing systems. At the same time we provide reference back-end Erlang system implementation for transactions processing. Also DSL gives us a natural and easy verification strategies and compactifications.

Listing 1: Deposit Program

```
program Deposit_Plus UAH
include 'PB-CASHBACK'
deposit duration range monthly 1 -> 20%
                       monthly 3 -> 22%
                       monthly 6 -> 22%
                       annual 23%
        withdraw disabled
        auto
        charge enabled monthly limit max 20000
        monthly 1% of amount to account 'bonus'
        monthly 15% name 'tax' of deposit
                    to account 'users/:client/tax'
```

This language could be easily extended to other domain areas like internet payment processing, shopping mall bonus programs, mobile operators tariff plans.

1.2 Objectives

The aim is to create small and compact language for payment transaction processing. Underlying instrumentation code should be KVS layer for storing transaction chains but naturally should be extended to different backends like Java, PL/SQL and other languages currently involved in banking card processing. We have several criteria to satisfy:

English	Self-explanatory
Fasten	Time-to-market
Optimized	Minimal Back-end Operations
Verified	No regular bugs. Only business logic.
Taxonomy	Sane structure for extensions

1.3 Operations

User Creation:

```
prepare user ':client'
    name     ':fullname'
    age      ':birth'
    phone    ':ph'
    document ':passport'
    accounts
        credit '/users/:client/credit' program 'PB-UNIVERSAL'
        account '/users/:client/:acc' program ':tariff'
```

Process Transaction:

```
prepare transaction from account 'users/:client'
                      to account ':beneficiar'
```

Notifying:

```
prepare event 'users/:client'
```

1.4 Programs

Programs are tariff programs, set of rules that we plug to transaction processing. It feels like set of filters triggered each time we fire money movements on account with a given card defenition.

Listing 2: BNF

```
Program = program Name Currency Forms

Form = limit Amount
     | grace Amount days
     | credit CreditRules
     | rate ChargeRule
     | version Amount
     | deposit DepositRules
     | accounts AccountList
```

Example:

Listing 3: credit.card

```
program PLA_DEB USD
limit 20000
version 1.0
credit monthly 10%
```

Programs are stored in its own space.

```
/programs/PB-UNIVERSAL.card
/programs/PB-DEPOSIT-PLUS.card
/programs/API.code
/programs/UA.user
```

1.5 Language Forms

Top level tariffs of billing rules are pluggable slangs that share some common part of the languages. These common part we will call language forms.

Listing 4: BNF

```
    Direction = charge
              | withdraw

   ChargeRule = Fixed + Percent
                of <amount | debt | credit | deposit | rate>
                limit <min Amount> | <max Amount>
                name Name
                to account Name

Periodically = monthly Amount
             | monthly Months -> ChargeRule
             | daily ChargeRule
             | annual ChargeRule

     Account = <credit | rate | deposit> Name
```

1.6 Accounts

Enterprise Tree handles clients, accounts, transactions, programs, events. Programs could be assigned to each node and fires atomatically on access.

```
/personal/:client
/personal/:client/bonus
/personal/:client/credit
/personal/:client/deposit
/personal/:client/rate
```

1.7 External Services

External service has its own endpoints, and could be addressed/-mounted? to local system.

```
/external/visa/:client
/external/master/:client
/external/swift/:client
/internet/paypal/:client
/bonus/:client
```

1.8 Transactions

Transactions are stored per each client's account.

```
/personal/:client/transactions
```

1.9 Deposit

Deposit program forms ususally provides such attributes of account as duration, rate, withdraw locking, charge limits, fee options and other deposit specific options. Deposit forms usually have "deposit" account name.

```
Enabled = enabled | disabled

Deposit = duration Periodically
        | duration range [Periodically]
        | withdraw Enabled
        | charge Enabled | charge Enabled Periodically
        | auto
        | final Periodically move from Id to Id
        | fee ChargeRule
        | Periodically
```

1.10 Credit

Credit programs forms mainly provide transaction filtering and other default account name ”credit”.

```
    Credit = transaction [TransRule]
           | sratus Enabled Ammount
           | Periodically

 TransRule = cashin  Amount
           | wire    ChargeRule
           | cashout Amount
```