# Scotiabank Mortgages in Clafer

*Defining the concept of mortgage and its variants using Clafer*

*Draft v. 3*
*April 24, 2011*

*By Michał Antkiewicz*
http://gsd.uwaterloo.ca/mantkiew
http://softwarecontinuum.com

## Overview

The purpose of this document is to demonstrate how a complex family of concepts can be modeled using Clafer. The document presents an exact record of the actual analysis performed by the author and shows a typical evolution of the concept definitions when using the "concrete-to-abstract" domain analysis approach.

Basic knowledge of Clafer is assumed. Refer to Concept Modeling Using Clafer - Tutorial for a thorough introduction.

## Concept: Mortgage

| | |
|---|---|
| Contributor | |
| Coverage | Mortgage Centre section of the source |
| Creator | Michał Antkiewicz |
| Date | As of Apr 19, 2011 |
| Description | Conceptual model of a family of mortgages offered by Scotiabank |
| Format | Clafer |
| Language | en |
| Rights | Bank of Nova Scotia |
| Source | http://www.scotiabank.com/cda/content/0,1608,CID8216_LIDen,00.html |
| Subject | Mortgages for individuals |

### Initial Analysis

This section presents how different variants of mortgages can be written down directly while reading the source. Each concept below represents "notes" taken during the initial read. The intention is to enumerate all variants and their properties as they are presented before abstracting the concepts.

```
abstract Mortgage
    valueProposition -> string
```

abstract FixedRate : Mortgage
    [ valueProposition = “Play it safe by knowing exactly what your mortgage rate and payment will be for the full term.” ]

abstract OpenTerm : FixedRate
    [ valueProposition = “Get fixed payments, the flexibility to pay off your mortgage faster, and the security of locking into another term at any time.” ]
    open
    xor term
        6months ; 1year
    interestRate
        fixedForTheFullTerm
    xor financingAvailable
        conventional
        insured
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly
    prepaymentOptions
        anyAmountUpToFullAmount
            atAnyTime
            withoutPenalty

abstract Flexible/Closed Mortgage : FixedRate
    [ valueProposition = “Lock into a competitive rate for 6 months with the option to convert to a longer term without penalty.” ]
    closed
    term
        6months
   interestRate
        fixedFor6months      -- which is, effectively, the same as fixedForFullTerm
    xor financingAvailable
        conventional
        insured
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly
    prepaymentOptions
        prepayUpTo15%OfOriginalAmount
        increasePaymentsByUpTo15%OfCurrentTermPayment

abstract 1,2YearClosedTermMortgages : FixedRate
    [ valueProposition = “Get the stability of a fixed rate and payment over the short-term at a very competitive rate.” ]
    closed
    xor term
        1year ; 2years
   interestRate
        fixedForFullTerm
    xor financingAvailable
        conventional

insured
xor paymentFrequency
weekly ; biweekly ; semi-monthly ; monthly
prepaymentOptions
prepayUpTo15%OfOriginalAmount
increasePaymentsByUpTo15%OfCurrentTermPayment

abstract 3,4,5,7YearClosedTermMortgages : FixedRate
[ valueProposition = “Get the stability of a fixed rate and payment over the long-term at a very competitive rate.” ]
closed
xor term
3years ; 4years ; 5years ; 7years
interestRate
fixedForFullTerm
xor financingAvailable
conventional
insured
xor paymentFrequency
weekly ; biweekly ; semi-monthly ; monthly
prepaymentOptions
prepayUpTo15%OfOriginalAmount
increasePaymentsByUpTo15%OfCurrentTermPayment
cashBackOption
cashBackUpTo5%ofMortgagePrinciple
upfront

abstract 5YearClosedTermMortgages : FixedRate
[ valueProposition = “By locking into a longer term mortgage, especially while current interest rates are so low, you can have the security of knowing your payment won't change.” ]
closed
term
5years
interestRate
fixedForFullTerm
xor financingAvailable
conventional
insured
xor paymentFrequency
weekly ; biweekly ; semi-monthly ; monthly
prepaymentOptions
prepayUpTo15%OfOriginalAmount
increasePaymentsByUpTo15%OfCurrentTermPayment

abstract 10YearClosedTermMortgages : FixedRate
[ valueProposition = “A competitive low rate and payment that are guaranteed for 10 years.” ]
closed
term
10years

interestRate
fixedForFullTerm
xor financingAvailable
conventional
insured
xor paymentFrequency
weekly ; biweekly ; semi-monthly ; monthly
prepaymentOptions
prepayUpTo15%OfOriginalAmount
increasePaymentsByUpTo15%OfCurrentTermPayment
cashBackOption
cashBackUpTo5%ofMortgagePrinciple
upFront

abstract VariableRate : Mortgage
[ valueProposition = “Consider a variable rate mortgage. Where the rate you pay fluctuates with Scotiabank Prime Rate.” ]

abstract ScotiaFlexValueMortgageClosedTerm : VariableRate
[ valueProposition = “Take advantage of a mortgage with a low rate and low payment.” ]
closed
term
5years
interestRate
resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
xor financingAvailable
conventional
insured
xor paymentFrequency
weekly ; biweekly ; semi-monthly ; monthly
prepaymentOptions
prepayUpTo15%OfOriginalAmount
increasePaymentsByUpTo15%OfCurrentTermPayment
addedBenefits
convertAnytimeToFixedTermProductWithTermGreaterThanTheRemainingTerm
withoutPenalty

abstract ScotiaFlexValueMortgageOpenTerm : VariableRate
[ valueProposition = “Flexibility means greater mortgage value: you get a low rate and low payments, and a guaranteed rate discount when you lock into Scotiabank's 5-year fixed rate.” ]
open
term
5years
interestRate
resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
xor financingAvailable
conventional
insured
xor paymentFrequency

```
        weekly ; biweekly ; semi-monthly ; monthly
    prepaymentOptions
        anyAmountUpToFullAmountAtAnyTimeWithoutPenalty
    addedBenefits
        convertAnytimeToFixedTermProductWithTermGreaterThanTheRemainingTerm
            withoutPenalty
```

These definitions can now be used to instantiate a mortgage for a client. For example

```
ExampleMortgage : ScotiaFlexValueMortgageOpenTerm
    [ conventional
      weekly ]
```

Note, that only two parameters needed to be specified as all other parameters are predetermined by the used mortgage variant (actually, other parameters such as principal amount are missing. We'll add them in the Section Context Analysis.

## Abstraction by commonality extraction

As can be easily seen, the variants listed in the previous section share many common properties: financingAvailable and paymentFrequency repeat in each variant and can be easily moved to the main concept Mortgage. Doing so allows us to remove these properties from each variant.

```
abstract Mortgage
    valueProposition -> string
    xor financingAvailable
        conventional
        insured
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly
```

Additionally, it can be seen that there is a fixed set of available term options that can be modeled as an enumeration:

```
enum MortgageTerm = 6months | 1year | 2years | 3years | 5years | 7years | 10years
```

Now we can say that every mortgage has a term that can be (in general) chosen from these values as follows:

```
abstract Mortgage
    ...
    term -> MortgageTerm
    ...
```

Also, every mortgage can be open or closed.

```
abstract Mortgage
    ...
    xor kind        -- this name does not come from the source. Need to confirm this abstract term with an SME.
        open
```

```
        closed
    ...
```

We observe that interestRate is the same among all variants of FixedRate mortgage with one exception: the variant Flexible/ClosedMortgage has interestRate.fixedFor6months. However, since the term for this variant is 6months, we can consider the interest rate to be fixedForFullTerm as well.

```
abstract FixedRate : Mortgage
    interestRate
        fixedForTheFullTerm
```

We observe the following commonalities among all variants of VariableRate mortgage:

- addedBenefits are the same
- interestRate is the same
- [ term = 5years ]

These commonalities can be moved to the definition of VariableRate as follows:

```
abstract VariableRate : Mortgage
    [ term = 5years ]
    interestRate
        resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
    addedBenefits
        convertAnytimeToFixedTermProductWithTermGreaterThanTheRemainingTerm
            withoutPenalty
```

After applying these modifications, the entire set of definitions is as follows:

```
enum MortgageTerm = 6months | 1year | 2years | 3years | 5years | 7years | 10years

abstract Mortgage
    valueProposition -> string
    term -> MortgageTerm
    xor kind
        open
        closed
    xor financingAvailable
        conventional
        insured
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly

abstract FixedRate : Mortgage
    [ valueProposition = “Play it safe by knowing exactly what your mortgage rate and payment will be for the full
term.” ]
    interestRate
        fixedForTheFullTerm
```

```
abstract OpenTerm : FixedRate
   [ valueProposition = “Get fixed payments, the flexibility to pay off your mortgage faster, and the security of
locking into another term at any time.”
      open
      term in 6months + 1year ]
   prepaymentOptions
      anyAmountUpToFullAmount
         atAnyTime
         withoutPenalty

abstract Flexible/Closed Mortgage : FixedRate
   [ valueProposition = “Lock into a competitive rate for 6 months with the option to convert to a longer term
without penalty.”
      closed
      term = 6months ]
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment

abstract 1,2YearClosedTermMortgages : FixedRate
   [ valueProposition = “Get the stability of a fixed rate and payment over the short-term at a very competitive rate.”
   closed
   term in 1year + 2years ]
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment

abstract 3,4,5,7YearClosedTermMortgages : FixedRate
   [ valueProposition = “Get the stability of a fixed rate and payment over the long-term at a very competitive rate.”
      closed
      term in 3years + 4years + 5years + 7years ]
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment
   cashBackOption
      cashBackUpTo5%ofMortgagePrinciple
         upfront

abstract 5YearClosedTermMortgages : FixedRate
   [ valueProposition = “By locking into a longer term mortgage, especially while current interest rates are so low,
you can have the security of knowing your payment won't change.”
      closed
      term = 5years ]
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment

abstract 10YearClosedTermMortgages : FixedRate
   [ valueProposition = “A competitive low rate and payment that are guaranteed for 10 years.”
```

```
      closed
      term = 10years ]
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment
   cashBackOption
      cashBackUpTo5%ofMortgagePrinciple
         upFront

abstract VariableRate : Mortgage
   [ valueProposition = "Consider a variable rate mortgage. Where the rate you pay fluctuates with Scotiabank
Prime Rate."
      term = 5years ]
   interestRate
      resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
   addedBenefits
      convertAnytimeToFixedTermProductWithTermGreaterThanTheRemainingTerm
         withoutPenalty

abstract ScotiaFlexValueMortgageClosedTerm : VariableRate
   [ valueProposition = "Take advantage of a mortgage with a low rate and low payment."
     closed ]
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment

abstract ScotiaFlexValueMortgageOpenTerm : VariableRate
   [ valueProposition = "Flexibility means greater mortgage value: you get a low rate and low payments, and a
guaranteed rate discount when you lock into Scotiabank's 5-year fixed rate."
      open ]
   prepaymentOptions
      anyAmountUpToFullAmount
         atAnyTime
         withoutPenalty
```

At this point, we clearly see that all fixed rate mortgages have interest rate fixedForTheFullTerm and all variable rate ones have resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges. We can make this choice explicit and move interestRate to Mortgage.

```
abstract Mortgage
   valueProposition -> string
   term -> MortgageTerm
   xor kind
      open
      closed
   xor interestRate
      fixedForTheFullTerm
      resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
   xor financingAvailable
```

```
        conventional
        insured
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly

abstract FixedRate : Mortgage
    [ valueProposition = "Play it safe by knowing exactly what your mortgage rate and payment will be for the full
term."
      fixedForTheFullTerm ]

abstract VariableRate : Mortgage
    [ valueProposition = [ "Consider a variable rate mortgage. Where the rate you pay fluctuates with Scotiabank
Prime Rate."
      term = 5years
      resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges ]
    addedBenefits
        convertAnytimeToFixedTermProductWithTermGreaterThanTheRemainingTerm
            withoutPenalty
```

All other definitions remain unchanged.

With that we can conclude conceptual analysis of the concept mortgage and its fixed and variable rate variants. At this point, we have a clear understanding of the common properties of every mortgage, distinguishing properties of fixed and variable rate variants, and a clear understanding of the particular distinguishing characteristics each concrete variant brings. Enriched with this understanding, we can proceed with a deeper analysis of the domain.

## Context Analysis

The definitions we obtained so far originate from the web site, which is meant to highlight main characteristics of the different mortgage variants from the marketing point of view. There are, however, other essential properties of mortgage that were not mentioned so far. To further enrich our definitions, we examine Mortgage Comparison Calculator.

We discover mortgageAmount, interestRate, amortization, and payment. The mortgageAmount must be between 5000$ and 9,999,999$, the interestRate must be between 0.5% and 25% and amortization between 1 and 30 years. We add these properties to the definition of Mortgage as follows. We however rename interestRate and payment to currentInterestRate and currentPayment as these values change over time.

```
abstract Mortgage
    valueProposition -> string
    term -> MortgageTerm
    xor kind
        open
        closed
    mortgageAmount -> Currency
    [ 5000 <= mortgageAmount <= 9999999 ]
    amortization -> integer
```

```
    [ 1 <= amortization <= 30 ]
    xor interestRate
        fixedForTheFullTerm
        resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
    currentInterestRate -> Percentage
    [ 0.5 <= currentInterestRate <= 25 ]
    currentPayment -> Currency
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly
    xor financingAvailable
        conventional
        insured
```

Given the following definitions of Currency and Percentage:

```
abstract Currency extends float

abstract Percentage extends float
    [ val >= 0 && val <= 100 ]
```

Now we examine a payment chart. We discover two more properties: principal and balance. The principal refers to the initial mortgage amount and balance refers to the outstanding mortgage amount which changes over time. We modify the definition by changing mortgageAmount to principalMortgageAmount and we add balance as balance.

```
abstract Mortgage
    ...
    principalMortgageAmount -> Currency
    [ 5000 <= principalMortgageAmount <= 9999999 ]
    balance -> Currency
    [ balance <= principalMortgageAmount ]
    ...
```

At this point, our definition covers both product information and mortgage comparison calculator contexts. A brief analysis of other on-line tools, such as, Build Your Mortgage Plan, What Can I Afford, Mortgage Payment Calculator**,** and Mortgage Selector, reveals that our definitions are adequate to these contexts as well.

In general, the definitions need to be revised and adapted to all relevant contexts to increase their usefulness and value.

Let us now create an example instance of the extended Mortgage concept:

```
ExampleMortgage : Mortgage
    [ term = 7years
      closed
      mortgageAmount = 234000
      amortization = 15
```

resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
currentInterestRate = 2.5
weekly
conventional ]

## Special Programs

Having defined the basic types of mortgages, we proceed to analyze additional variants of mortgages presented in Special Programs section of the source.

The Save Now, Save Later mortgage is a closed fixed rate mortgage with a 1 year term and an optional renewal term for 5 years. It has an interest rate discount for the initial term (1.66%) and one for the optional renewal term (1.25%).

```
abstract SaveNow,SaveLaterMortgage : FixedRate
   [ valueProposition = "Save now with a competitive mortgage rate and save later with a guaranteed rate discount. Limited time offer."
     closed
     term = 1year ]
   optionalRenewalTerm -> MortgageTerm = 5years
   initialInterestRateDiscount -> Percentage = 1.66
   renewalInterestRateDiscount -> Percentage = 1.25
   prepaymentOptions
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment
   addedBenefits
      renewTo5-year,FixedRate,ClosedTerm
         noEarlyRenewalInterest
         noEarlyPrepaymentPenalties
```

At this point, we see that this variant reuses the commonly used "15%" prepayment options. We may want to factor out these and the other commonly used prepayment options, so that they can be referenced rather than duplicated in each definition.

```
abstract 15%Prepayment15%Increase
      prepayUpTo15%OfOriginalAmount
      increasePaymentsByUpTo15%OfCurrentTermPayment

abstract anyAmountAnytime
     anyAmountUpToFullAmount
         atAnyTime
         withoutPenalty
```

With that, these prepayment options became independent concepts and they can now be used as follows (all other definitions need to be updated accordingly):

```
abstract SaveNow,SaveLaterMortgage : FixedRate
   ...
```

```
    prepaymentOptions -> 15%Prepayment15%Increase


abstract OpenTerm : FixedRate
    ...
    prepaymentOptions -> anyAmountAnytime
```

The Long and Short Mortgage is actually a mortgage composed of a fixed-rate, closed-term mortgage for a part of the principal and the Scotia Flex Value variable-rate mortgage for the rest of the principal. For illustrative purposes (not confirmed with an SME), we assume that both mortgages need to have the same kind of financing and payment frequency as the main mortgage.

```
abstract LongAndShortMortgage : Mortgage
    [ valueProposition = “Trying to decide between short-term rates and more secure long-term borrowing options? Here's a mortgage for you.” ]
    fixedRate : FixedRate
        [ closed ]              -- only closed fixed rate mortgages are valid, otherwise this constraint will be violated
    flexValue : VariableRate

    [ principalMortgageAmount = fixedRate.principalMortgageAmount + flexValue. principalMortgageAmount]
    [ balance = fixedRate.balance + flexValue.balance ]

    [ conventional <=> fixedRate.conventional && flexValue.conventional ]
    [ insured <=> fixedRate.insured && flexValue.insured ]

    [ weekly <=> fixedRate.weekly && flexValue.weekly ]
    [ biweekly <=> fixedRate.biweekly && flexValue.biweekly ]
    [ semi-monthly <=> fixedRate. semi-monthly && flexValue. semi-monthly]
    [ monthly <=> fixedRate.monthly && flexValue.monthly]

    prepaymentOptions -> 15%Prepayment15%Increase
    addedBenefits
        partOfScotiaTotalEquityPlan
```

The Secondary Home Financing Program is designed for all types of Secondary Homes, including Type A and Type B vacation properties.

```
abstract SecondaryHomeFinancingProgram : Mortgage
    [ valueProposition = “If you're looking for a getaway or a second home to call your own, consider your many financing options.”
      term != 10years ]      -- any term other than 10years
    prepaymentOptions -> 15%Prepayment15%Increase
    conditions
        [ loanToValue <= 90
          75 <= loanToValue <= 90 <=> insured ]

abstract ScotiaMortgageForSelfemployed : Mortgage
    [ valueProposition = “Are you self-employed and looking to buy a home? See how much easier it can be.”
      term != 10years ]
    prepaymentOptions -> 15%Prepayment15%Increase
```

```
    conditions
        [ loanToValue <= 90
          75 <= loanToValue <= 90 <=> insured ]
```

The last two definitions additionally contain conditions which constrain the loanToValue ratio and specify when the mortgage has to be insured. We now have to add the loanToValue property to the definition of Mortgage. Since, loanToValue is a ratio between balance and propertyValue, we need to add the latter and constrain dependencies among the three values.

```
abstract Mortgage
    ...
    propertyValue -> Currency
    loanToValue -> Percentage
    [ loanToValue = balance /propertyValue ]
    ...
```

The StartRight mortgage program can be configured as open/closed, fixed/variable, and with any term. It has a stricter requirement for insurance for temporary residents (from 65% loanToValue) as compared to 75% for permanent residents. Minimum 5% downpayment is required.

```
abstract ScotiabankStartRightMortgageProgramForTemporaryResidents : Mortgage
    [ valueProposition = "We can help you feel at home faster if you are working and living in Canada temporarily." ]
    prepaymentOptions -> 15%Prepayment15%Increase
    conditions
        [ loanToValue <=  95
          65 <= loanToValue <=  95 <=> insured ]

abstract ScotiabankStartRightMortgageProgramForPermanentResidents : Mortgage
    [ valueProposition = "A specially designed program to meet your mortgage needs and help you in obtaining your
first home in Canada." ]
    prepaymentOptions -> 15%Prepayment15%Increase
    conditions
        [ loanToValue <=  95
          75 <= loanToValue <=  95 <=> insured ]
```

Since the two programs only differ in the lower bound of loanToValue, they can be collapsed into a single definition as follows:

```
abstract ScotiabankStartRightMortgageProgram : Mortgage
    xor residentType
        temporary
            [ valueProposition = "We can help you feel at home faster if you are working and living in Canada
temporarily." ]
        permanent
            [ valueProposition = "A specially designed program to meet your mortgage needs and help you in obtaining
your first home in Canada." ]
    prepaymentOptions -> 15%Prepayment15%Increase
    conditions
```

```
[ loanToValue <=  95
  (temporary => ( 65 <= loanToValue <=> insured) ) | |
  (permanent => ( 75 <= loanToValue <=> insured))   ]        -- if temporary then … or if permanent …
```

It can now be clearly seen that the program is designed for only temporary and permanent residents, both the value proposition and the lower bound for insurance requirement depend on the resident type. All other properties are inherited from the concept Mortgage.

At this point, we have modeled all fixed and variable rate mortgages as well as all special program mortgages.

## Scotia Total Equity Plan

It is a plan that can be custom built for the particular needs and circumstances of the borrower. It is not a mortgage itself but it can combine a variety of other borrowing programs to borrow for up to 80% of the value of client's property (that is *home equity*).

```
abstract ScotiaTotalEquityPlan
    borrowingLimit -> Currency
        establishedWithaSingleApplication
    currentMortgage -> Mortgage     -- requires an existing mortgage
    [ borrowingLimit = currentMortgage.principalMortgageAmount  - currentMortgage.balance ]
    homeEquity -> Percentage
    [ homeEquity >= 20
      homeEquity = currentMortgage.balance * 100 / currentMortgage.principalMortgageAmount ]
    -- choose any combination of products
    'FixedRate *
    'VariableRate *
    'LongAndShortMortgage *
    'ScotiaLinePersonalLineofCredit ?
    'CreditLineForBusiness *
    'ScotiaLineForBusinessVISACard *
    'ScotiaGoldPassportForBusinessVISACard *
    'ScotiabankVisaCard *
    'ScotiaPlanPersonalLoans *
    'TermLoanForBusiness *
    'OverdraftProtection ?
```

Such a definition makes it explicit that

1. An existing mortgage with Scotiabank is required
2. Limit of home equity and how to calculate it
3. Borrowing limit and how to calculate it
4. List of products that can be instantiated as part of the plan and valid numbers of instances (e.g., a client may have any number of fixed rate mortgages but only at most one overdraft protection). The list assumes that all products were already defined.

Interestingly, the above definition is what the author has understood from reading the *Overview* and *How it works?* tabs of the source only. Crosschecking the definition with the example presented in *Meet*

*the Wongs* tab revealed critical mistakes (and precisely illustrates the power of "specification by example" and "concrete to abstract" modeling approaches)!

Actually, the borrowing limit is established on the basis of the appraised home value. The correct definition is as follows:

```
abstract ScotiaTotalEquityPlan
    borrowingLimit -> Currency       -- calculated
        establishedWithASingleApplication
    appraisedPropertyValue -> Currency       -- provided in application
    [ borrowingLimit = .80 * appraisedPropertyValue ]
    currentMortgageBalance -> Currency       -- provided in application
    homeEquity -> Percentage         -- calculated
    [ homeEquity >= 20
      homeEquity = currentMortgageBalance * 100 / appraisedPropertyValue]
    -- choose any combination of products
    'FixedRate *
    'VariableRate *
    'LongAndShortMortgage *
    'ScotiaLinePersonalLineofCredit *
    'CreditLineForBusiness *
    'ScotiaLineForBusinessVISACard *
    'ScotiaGoldPassportForBusinessVISACard *
    'ScotiabankVisaCard *
    'ScotiaPlanPersonalLoans *
    'TermLoanForBusiness *
    'OverdraftProtection ?
    [ (sum FixedRate.balance                                +
      sum VariableRate.balance                              +
      sum LongAndShortMortgage.balance                      +
      sum ScotiaLinePersonalLineofCredit.balance            +
      sum CreditLineForBusiness.balance                     +
      sum ScotiaLineForBusinessVISACard.balance             +
      sum ScotiaGoldPassportForBusinessVISACard.balance     +
      sum ScotiabankVisaCard.balance                        +
      sum ScotiaPlanPersonalLoans.balance                   +
      sum TermLoanForBusiness.balance                       +
      OverdraftProtection.balance)
      <= borrowingLimit ]            -- the grand total of all balances must be less or equal to the borrowing limit
```

Any existing products can be *transferred* to the total equity plan in which case they are reinstantiated within the plan.

Given this definition of the total equity plan and assuming existing definitions of other products, such as lines of credit and visa cards, we can express the Wong example as an instance of the total equity plan as follows:

```
theWongsEquityPlan : ScotiaTotalEquityPlan
    [ appraisedPropertyValue  = 240000
      currentMortgageBalance = 85000 ]
    -- the calculated borrowingLimit is 192000
    -- the minimum 20% home equity requirement is satisfied
    -- the example does not specify exactly which mortgage the Wongs have. Let's assume a variable rate
    'ScotiaFlexValueMortgageOpenTerm
        [ balance = 85000 ]
    'ScotiaLinePersonalLineofCredit
        [principalAmount = 35000
          balance = 25000 ]
    'ScotiaLinePersonalLineofCredit
        [principalAmount = 25000
          balance = 5000 ]
    'ScotiaPlanPersonalLoan
        [principalAmount = 25000
          balance = 25000 ]
    'ScotiabankVisaCard
        [creditLimit = 10000
          balance = 3150 ]
```

Finally, another possible way to define the total equity plan would be to simply link to existing products using ->, for example:

```
abstract ScotiaTotalEquityPlan
    ...
    fixedRateMortgages -> FixedRate *
    ...
```

In this case, each product would need to be configured to be part of the total equity plan, for example,

```
WongsFixedMortgage : 5YearClosedTermMortgages
    ...
    [ partOfTotalEquityPlan = theWongsEquityPlan ]
    ...

theWongsEquityPlan : ScotiaTotalEquityPlan
    ...
    [WongsFixedMortgage in fixedRateMortgages ]
```

## Further possibilities

The definitions can further be extended with terms and conditions (missing for fixed and variable rate mortgages), additional business rules not present in the source or missed by the author, current interest rates and different calculations such as current payment and actual amortization, and the lifecycle of the mortgages (e.g., LongAndShortMortgage has an interesting lifecycle where the borrower may choose to extend for a 5year term after the initial 1year term has matured. Upon such extension, a different interest rate becomes used in calculations etc.).

Lifecycle modeling presents an interesting challenge. First, the states which the mortgage may be in and transitions between these states must be identified. Next, the states may impact other properties and validity of some constraints of the mortgage. Let's briefly sketch the idea by assuming that a regular mortgage can be in the following states: appliedFor, active, cancelled, and matured. The only valid sequences of transitions are: appliedFor->active-> cancelled (cancelled during term), appliedFor-> cancelled (never approved), and appliedFor->active->matured (ran full term).

```
abstract Mortgage
  xor states
      appliedFor   -- initial state
      active
      cancelled
      matured
   [ appliedFor ---> active            -- these are the only valid transitions
     appliedFor ---> cancelled
     active---> cancelled
     active---> matured ]
```

Lifecycle modeling is currently a proposed extension of Clafer and it is not implemented.

## Summary and Conclusion

In this document, we presented an exact record of the analysis performed by the author and the way Clafer can be used for "concrete to abstract" modeling. The final set of definitions is presented in the Appendix.

Such definitions can be used in many ways, including:

- Precise communication among all stakeholders including business analysts, developers, testers.
- Exploring all possible examples of concepts using *interactive concept instantiation* or *automatic instance generation* (tool support required). Such exploration enables systematic elicitation of concept constraints (i.e., overconstrained concept will not allow correct instances, and underconstrained concept will allow incorrect instances).
- Validation of the concept with business stakeholders using both examples (instances) and the abstraction.
- Applying domain-driven design (DDD). For example, Clafer naturally supports aggregates with property nesting and '. Exploring using Clafer for DDD is future work.

## Appendix – Final and Complete Set of Definitions

Please note that the entire set of definitions that covers all products from the source is represented only in less than five pages.

```
-- most general concept: Mortgage

abstract Mortgage
    valueProposition -> string
    term -> MortgageTerm
    xor kind
        open
        closed

    principalMortgageAmount -> Currency
    [ 5000 <= principalMortgageAmount <= 9999999 ]
    balance -> Currency
    [ balance <= principalMortgageAmount ]
    propertyValue -> Currency
    loanToValue -> Percentage
    [ loanToValue = balance /propertyValue ]
    amortization -> integer
    [ 1 <= amortization <= 30 ]
    xor interestRate
        fixedForTheFullTerm
        resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges
    currentInterestRate -> Percentage
    [ 0.5 <= currentInterestRate <= 25 ]

    currentPayment -> Currency
    xor paymentFrequency
        weekly ; biweekly ; semi-monthly ; monthly
    xor financingAvailable
        conventional
        insured

-- prepayment options

abstract 15%Prepayment15%Increase
        prepayUpTo15%OfOriginalAmount
        increasePaymentsByUpTo15%OfCurrentTermPayment

abstract anyAmountAnytime
       anyAmountUpToFullAmount
            atAnyTime
            withoutPenalty

-- fixed rate mortgages

abstract FixedRate : Mortgage
    [ valueProposition = “Play it safe by knowing exactly what your mortgage rate and payment will be for the full
```

```
term."
      fixedForTheFullTerm ]

abstract OpenTerm : FixedRate
    [ valueProposition = "Get fixed payments, the flexibility to pay off your mortgage faster, and the security of
locking into another term at any time."
      open
      term in  6months + 1year ]
    prepaymentOptions -> anyAmountAnytime

abstract Flexible/Closed Mortgage : FixedRate
    [ valueProposition = "Lock into a competitive rate for 6 months with the option to convert to a longer term
without penalty."
      closed
      term = 6months ]
    prepaymentOptions -> 15%Prepayment15%Increase

abstract 1,2YearClosedTermMortgages : FixedRate
    [ valueProposition = "Get the stability of a fixed rate and payment over the short-term at a very competitive rate."
    closed
    term in 1year + 2years ]
    prepaymentOptions -> 15%Prepayment15%Increase

abstract 3,4,5,7YearClosedTermMortgages : FixedRate
    [ valueProposition = "Get the stability of a fixed rate and payment over the long-term at a very competitive rate."
      closed
      term in 3years + 4years + 5years + 7years ]
    prepaymentOptions -> 15%Prepayment15%Increase
    cashBackOption
        cashBackUpTo5%ofMortgagePrinciple
            upfront

abstract 5YearClosedTermMortgages : FixedRate
    [ valueProposition = "By locking into a longer term mortgage, especially while current interest rates are so low,
you can have the security of knowing your payment won't change."
      closed
      term = 5years ]
    prepaymentOptions -> 15%Prepayment15%Increase

abstract 10YearClosedTermMortgages : FixedRate
    [ valueProposition = "A competitive low rate and payment that are guaranteed for 10 years."
      closed
      term = 10years ]
    prepaymentOptions -> 15%Prepayment15%Increase
    cashBackOption
        cashBackUpTo5%ofMortgagePrinciple
            upFront

-- variable rate mortgages
```

```
abstract VariableRate : Mortgage
   [ valueProposition = [ “Consider a variable rate mortgage. Where the rate you pay fluctuates with Scotiabank
Prime Rate.”
     term = 5years
     resetTogetherWithPaymentAmountEachTimeScotiabankPrimeRateChanges ]
   addedBenefits
      convertAnytimeToFixedTermProductWithTermGreaterThanTheRemainingTerm
         withoutPenalty

abstract ScotiaFlexValueMortgageClosedTerm : VariableRate
   [ valueProposition = “Take advantage of a mortgage with a low rate and low payment.”
     closed ]
   prepaymentOptions -> 15%Prepayment15%Increase

abstract ScotiaFlexValueMortgageOpenTerm : VariableRate
   [ valueProposition = “Flexibility means greater mortgage value: you get a low rate and low payments, and a
guaranteed rate discount when you lock into Scotiabank's 5-year fixed rate.”
     open ]
   prepaymentOptions -> anyAmountAnytime

-- special programs mortgages

abstract SaveNow,SaveLaterMortgage : FixedRate
   [ valueProposition = “Save now with a competitive mortgage rate and save later with a guaranteed rate discount.
Limited time offer.”
     closed
     term = 1year ]
   optionalRenewalTerm -> MortgageTerm = 5years
   initialInterestRateDiscount -> Percentage = 1.66
   renewalInterestRateDiscount -> Percentage = 1.25
   prepaymentOptions -> 15%Prepayment15%Increase
   addedBenefits
      renewTo5-year,FixedRate,ClosedTerm
         noEarlyRenewalInterest
         noEarlyPrepaymentPenaltie

abstract LongAndShortMortgage : Mortgage
   [ valueProposition = “Trying to decide between short-term rates and more secure long-term borrowing options?
Here's a mortgage for you.” ]
   fixedRate : FixedRate
      [ closed ]
   flexValue : VariableRate

   [ principalMortgageAmount = fixedRate.principalMortgageAmount + flexValue. principalMortgageAmount]
   [ balance = fixedRate.balance + flexValue.balance ]

   [ conventional <=> fixedRate.conventional && flexValue.conventional ]
   [ insured <=> fixedRate.insured && flexValue.insured ]
```

```
    [ weekly <=> fixedRate.weekly && flexValue.weekly ]
    [ biweekly <=> fixedRate.biweekly && flexValue.biweekly ]
    [ semi-monthly <=> fixedRate.semi-monthly && flexValue.semi-monthly]
    [ monthly <=> fixedRate.monthly && flexValue.monthly]

    prepaymentOptions ->  15%Prepayment15%Increase
    addedBenefits
        partOfScotiaTotalequityPlan

abstract SecondaryHomeFinancingProgram : Mortgage
    [ valueProposition = “If you're looking for a getaway or a second home to call your own, consider your many
financing options.”
      term != 10years ]
    prepaymentOptions ->  15%Prepayment15%Increase
    conditions
        [ loanToValue <= 90
          75 <= loanToValue <= 90 <=> insured ]

abstract ScotiaMortgageForSelfemployed : Mortgage
    [ valueProposition = “Are you self-employed and looking to buy a home? See how much easier it can be.”
      term != 10years ]
    prepaymentOptions ->  15%Prepayment15%Increase
    conditions
        [ loanToValue <= 90
          75 <= loanToValue <= 90 <=> insured ]

abstract ScotiabankStartRightMortgageProgram : Mortgage
    xor residentType
        temporary
            [ valueProposition = “We can help you feel at home faster if you are working and living in Canada
temporarily.” ]
        permanent
            [ valueProposition = “A specially designed program to meet your mortgage needs and help you in obtaining
your first home in Canada.” ]
    prepaymentOptions -> 15%Prepayment15%Increase
    conditions
        [ loanToValue <=  95
          (temporary => ( 65 <= loanToValue <=> insured) ) | |
          (permanent => ( 75 <= loanToValue <=> insured))   ]          -- if temporary then ... or if permanent ...
```

```
-- Scotia Total Equity Plan

abstract ScotiaTotalEquityPlan
    borrowingLimit -> Currency       -- calculated
        establishedWithASingleApplication
    appraisedPropertyValue -> Currency       -- provided in application
    [ borrowingLimit = 80% * appraisedPropertyValue ]
    currentMortgageBalance -> Currency       -- provided in application
    homeEquity -> Percentage         -- calculated
    [ homeEquity >= 20
      homeEquity = currentMortgageBalance * 100 / appraisedPropertyValue]
    -- choose any combination of products
    'FixedRate *
    'VariableRate *
    'LongAndShortMortgage *
    'ScotiaLinePersonalLineofCredit *
    'CreditLineForBusiness *
    'ScotiaLineForBusinessVISACard *
    'ScotiaGoldPassportForBusinessVISACard *
    'ScotiabankVisaCard *
    'ScotiaPlanPersonalLoans *
    'TermLoanForBusiness *
    'OverdraftProtection ?
    [ (sum FixedRate.balance                                +
      sum VariableRate.balance                              +
      sum LongAndShortMortgage.balance                      +
      sum ScotiaLinePersonalLineofCredit.balance            +
      sum CreditLineForBusiness.balance                     +
      sum ScotiaLineForBusinessVISACard.balance             +
      sum ScotiaGoldPassportForBusinessVISACard.balance  +
      sum ScotiabankVisaCard.balance                        +
      sum ScotiaPlanPersonalLoans.balance             +
      sum TermLoanForBusiness.balance                       +
      OverdraftProtection.balance)
      <= borrowingLimit ]            -- the grand total of all balances must be less or equal to the borrowing limit
```