Análise, Modelação e Teste

18 de Fevereiro de 2009

1. Considere o seguinte modelo para representar os leilões em curso numa conhecida leiloeira online:

```
open util/ordering[Bid]

sig Client, Product, Bid {}
sig Auction {
  product : Product
}
sig Ebay {
  clients : set Client,
  auctions : Client -> Auction,
  bids : Client -> Auction -> Bid
}

pred A [e : Ebay] {
  e.auctions in e.clients -> Auction
  e.bids in e.clients -> Client.(e.auctions) -> Bid
}
pred B [e : Ebay] {
  all disj a,b : e.clients | no e.auctions[a] & e.auctions[b]
}
pred C [e : Ebay] {
  no e.bids.Bid.product & e.auctions.product
}
pred NoEqualBids[e : Ebay] {...}

pred Inv [e : Ebay] {
  A[e] && B[e] && C[e] && NoEqualBids[e]
}

pred IsCurrentWinner [e : Ebay, c : Client, p : Product] {...}

pred NewBid [e,e' : Ebay, c : Client, p : Product] {...}

assert NewBidOk {...}
```

   (a) Qual o objectivo dos invariantes `A`, `B`, e `C`?

   (b) Especifique o invariante `NoEqualBids`, cujo objectivo é impedir que haja dois bids de igual valor no mesmo leilão.

   (c) Especifique o predicado `IsCurrentWinner`, que deverá ser verdadeiro se o cliente `c` tem um bid vencedor num leilão para o produto `p`.

   (d) Defina a asserção `NewBidOk`, cujo objectivo é garantir que a operação `NewBid` está correctamente especificada: deve ser garantida a preservação do invariante e que após a sua execução com sucesso o cliente `c` tem um bid vencedor num leilão para o produto `p`.

   (e) Especifique a operação `NewBid` por forma a satisfazer a asserção `NewBidOk`.

2. Considere o seguinte fragmento de uma classe Java para modelar conjuntos de inteiros:

```
public class Set {
  private static int MAXSIZE=10;
  public int[] contents;
  public int numElems;
  ...
}
```

   (a) Anote o fragmento apresentado com invariantes e outras anotações que entenda adequadas. Considere para o efeito que os elementos do conjunto são armazenados de forma ordenada e sem repetições no array.

   (b) Especifique o método `pertence` que testa se um dado elemento pertence (ou não) ao conjunto.

   (c) Especifique o método `adiciona` que adiciona um novo elemento ao conjunto. Considere para o efeito que é lançada a excepção `SetIsFull` se o conjunto exceder a capacidade.

   (d) Especifique o método `subtrai` que, recebendo um conjunto como argumento, subtrai os elementos desse conjunto (i.e. $X' = X \setminus Y$).

3. Google Chrome (`http://www.google.com/chrome`) is an open source web browser created by Google. The main programming language used for Chrome is `C++`. Source code metrics have been derived for Google Chrome at the file level and at the method level. They can be found in (see spreadsheet `GoogleChromeMetrics.xls`).

   (a) In order to assess the quality of the source code of Chrome, create a quality profile for complexity at the method level. Use the following guidelines:

   - Complexity risk categories are: *low risk* (1-10), *moderate risk* (11-20), *high risk* (21-50), and *very high risk* (50+).
   - Sum the volume (lines of code) per risk category.
   - Derive the relative volume (percentage) per risk category.
   - Create a pie-chart of the percentages.

   (b) What should the risk categories be for redundancy at the file level, if we want at most 10% of the volume (lines of code) of the system to fall into the very high risk category, 10% of the volume in the high risk category, and 10% of the volume in the moderate risk category? Take the following steps:

   - Sort by redundancy.
   - Compute the relative volume (percentage) for each file.
   - Complete the following table: *low risk* (0% - 2.6%), *moderate risk* (2.6% - ...), *high risk* (...- ...), *very high risk* (...- 100%).

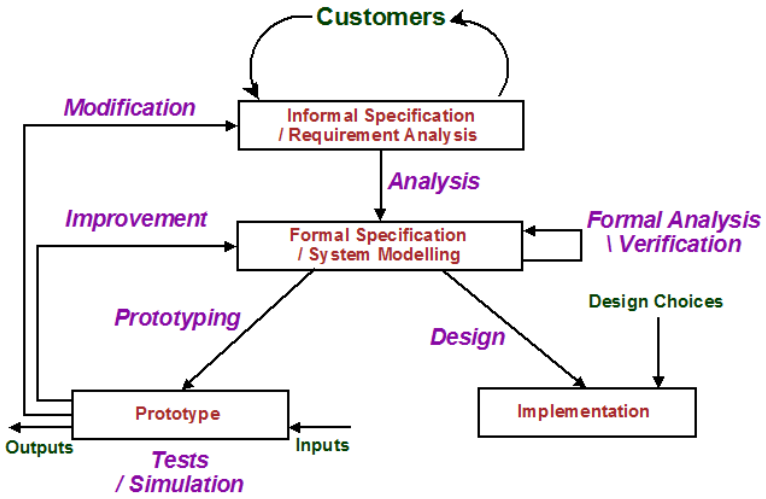
4. Considere a seguinte função que recebe dois inteiros, `m` e `n`, não negativos e não ambos nulos, e devolve o máximo divisor comum (`mdc`) de `m` e `n`.

```
1: long mdc(long m, long n)  {
2:         long d;
3:         if (n == 0) return m;
4:         if (m == 0) return n;
5:         d = min(m,n);
6:         while (m % d != 0 || n % d != 0)  {
7:                 d--;
8:         }
9:         return d;
10: }
```

   (a) Defina um conjunto mínimo de casos de teste para cobrir todas as instruções, condições e decisões. Justifique a cobertura.

   (b) Os casos de teste da alínea anterior garantem cobertura modificada de condições e decisões (MC/DC - *multiple condition decision coverage*)? Justifique.

5. Considere o seguinte ciclo de desenvolvimento de software proposto por Balzer:



   Que papel advoga para cada uma das linguagens / técnicas / ferramentas leccionadas nesta disciplina nas diversas fases deste ciclo de desenvolvimento. Justifique as suas escolhas.