2012 October 20

CSE 3311 Software Design
Report 3: Specification

Due: Thursday, November 8, 2012
Where: In class
If the class has begun your report is late

1 Main Points

The explicit specification is the union of this document plus the program text you are given. Be sure to consider all the implicit specifications, for example and without limitation, be sure to read and follow all the guidelines from the links on **reports** and **academic honesty** from the WWW home page for the course.

1.1 Learning objectives

- Using verification to find and correct errors in an algorithm
- Contracts for classes
- Writing agents and tuples for use in contracts

1.2 To hand in

For reports done in pairs, include an appendix describing the contributions of the two team members.

- Cover page – printed from the course web pages
- Report
  1. Document your proof and correction for the routine **prime** described in Section 2.1.
  2. Document the contract you developed for the TOOL_RENTAL class.
- Listing of the file tool_rental.e

1.3 Electronic submission

Before the deadline, submit the following directories

- `submit 3311 r3 tool_rental`
  The directory should contain all `.e` files, as well as the **.ecf** file for the **tool_rental** system.

No other files should be submitted (e.g. EIFGEN files etc). Files cannot be deleted – the submit command can only add or replace files – so be very careful to clean up your directory before any submission. You should use `eclean` on Prism before submitting to clear away the unnecessary files.

While you can develop your system on your personal computer, be sure your program will compile on Prism using **estudio66**.

1.4 To get started

Download the file `3311_report3.tar.gz` to a local directory. When you untar with the command `tar xzf 3311_report3.tar.gz`, you will get a directory called `report3` that contains the following files.

- Directory **tool_rental**
  - **tool_rental.e** – The class representing a tool renting shops. You are to modify this class.
  - **rental.ecf** – Do not modify this file

- **rental_desc.e**, **repair_desc.e**, **tool_desc.e** – They are used to implement the tool rental shop routines and its contracts. Do not modify these files.
- **owner.e** – the start of system execution, needed for compilation. Do not modify this file.

2 Tasks

2.1 Use a proof to correct the function prime

The following Eiffel routine employs a loop to determine the number **n** is prime. There is, however, an error in its implementation. Your task is to use the given contract and loop invariant to prove the "correctness" of the routine. If you are careful with your proof, then the source of the error should reveal itself at the proper step of the proof. Follow the normal course of proof until you reach an inconsistency, then indicate what is the source of the error in the routine that gives the inconsistency, make the simplest correction and complete the proof of the entire routine – one change is better than two.

NOTE: ¬(k | n) is read as "not k divides n"

```
prime (n: INTEGER): BOOLEAN
    -- Asserts that n a prime number
  local
    i   : INTEGER  -- Test divisor
    max : INTEGER  -- Maximum integer to try
  do
    from
      i := 2
      max := floor (sqrt (n))
      Result := True
    invariant
      Result ⇔ ∀ k : INTEGER | 2 ≤ k < i •  ¬(k | n)
    until
      i = max  or  not Result
    loop
      Result := not ((n \\ i) = 0)  -- \\ is the mod function
      i := i + 1
     variant max - i + 1
   end
  ensure
    Result ⇔  ∀ k : INTEGER | 2 ≤ k ≤ ⌊√n⌋ •  ¬(k | n)
  end
```

2.3 Contract for the class TOOL_RENTAL

This part of the report is a subset of ADT documentation that is restricted to a description of the contracts.

Complete the contract implementation for the class TOO_RENTAL for the given routines; the core of the tool rental system. All your contract clauses are to be executable, in that they should compile. You are not to implement the body of the routines. But you must implement all supporting quantifier and agent routines. For your support routines you are to write appropriate assertions. For agents the assertions should be compliable. For quantifiers, the assertions should be comments. For all loops in your support routines, you are to write, as comments, loop invariant and variant clauses.

3 Grading scheme

The grade for the report is partitioned into the following parts.

1. Presentation – 10%
2. Correcting and proving the correctness of the prime routine – 30%
3. The contract for tool rental – 60%
   35% for the program and 25% for the report