Your Name:

College of Charleston
Department of Computer Science
CSCI 360 Software Architecture and Design
Spring 2007
Jim Bowring

Final Examination

Answer all questions with complete sentences unless noted otherwise in the question. You may use the back of the exam papers to organize your thoughts and for continuations of answers when necessary. Please write your answers legibly.

1. (5) What are four kinds of visibility supported in UML?

   _____

   _____

2. (7) Describe two creational and two transformational techniques for generating a mid-level design model.

   _____

   _____

   _____

   _____

   _____

   _____

   _____

   _____

3. (5) Compare and contrast inheritance and delegation.

   _____

   _____

   _____

   _____

   _____

   _____

   _____

   _____

Your Name:

4. (5) Is there such a thing as an abstract attribute? If so, explain what it is, and if not, explain why not.

5. (3) Sequence diagrams: What is a selector in a lifeline identifier, and for what is it used?

6. (5) Compare and contrast *polling* and *notification* as design approaches for object interaction.

7. (5) What are the main differences between centralized, delegated, and dispersed control styles?

Your Name:

8. (10) Draw a sequence diagram based on the following Java program. Your diagram should illustrate the call **write(2)** directed to the **fibNumber** object in the **main()** method. Show all calls (including recursive calls) with execution occurrences.

```
Public class Fibonacci {
        Public static void main( String[] argv){
                Fibonacci fibNumber = new Fibonacci();
                fibNumber.write(2);
        }
        Public void write (int n ) {
                int result = fib(n);
                System.out.println(result);
        }
        Public int fib(int n) {
                If (n < 2)
                        return 1;
                else
                        return fib(n-1) + fib(n-2);
        }
}
```

Your Name:

9. (15) Suppose that a program simulates checkout lines at a supermarket. An instance of a **Queue** class that holds instances of **Customers** simulates each lane. A **Checkout** Object has 10 **Queue** objects. Time is simulated by a **Clock** instance. Every simulated minute, the **Clock** notifies the **Checkout** that time has passed. The **Checkout** then calls its own **addCustomers()** operation. This operation adds new **Customer** objects to each **Queue** by generating a random number between zero and four (using a **Random** object) and adding that many new **Customer** objects to the **Queue**. It them calls its own **processCustomers()** operation. This method removes **Customers** from each **Queue** by generating a random number between one and three (using a **Random** object) and removing that number of **Customers** form the **Queue** an destroying them. It also records statistics about the simulation for later display. Design this interaction and document it using a class diagram and one or more sequence diagrams.

Your Name:

(9 continued if you need the space)

Your Name:

10. (10) In a room with two doors, it is common to have two switches controlling a single light: one by each door. Flipping either switch changes the light's state from on to off, or vice versa. Make a state diagram illustrating this situation with a concurrent composite state that includes regions for each switch and the light. Your state diagram should track the states of switches **A** and **B** (**Up** or **Down**) and the state of the **Light** (**On** or **Off**.) The events are all switch **flips**.

11. (7) The parity of a bit string is even if the bit string has an even number of ones and odd otherwise. Design an acceptor to recognize all even-parity bit strings.

Your Name:

12. (5) What two strategies are integral to restricting access to program entities?

13. (5) What are two occasions when it is appropriate to extend access beyond visibility?

14. (8) Define Design by Contract as we studied it in the text.

Your Name:

15. (10) A parking garage control program has a **Garage** class responsible for keeping track of the number of spaces, the number of free spaces, the number of occupied spaces, and whether it is full or not. The **Garage** class has **enter()** and **leave()** operations called when a vehicle enters or leaves. It also has **setSpaces(int n)** operation used to modify the total number of spaces (in case of construction or some other problem) and a **setOccupiedSpaces(int n)** operation (to correct any miscounts.) Finally, it has operations to indicate whether the garage is full or empty. Write op-specs for the operations in the **Garage** class. Include class invariants.

16. (3) What fields of an op-spec are most concerned with detailing an operation's interface?

Your Name:

17. (EC 1) What does ACM stand for?

_____

_____

18. (EC 2) Who is David Harel?

_____

19. (EC 2) Who is Bertrand Meyer?

_____

20. (EC 2) Suggest two improvements for this course.

_____

_____

_____

_____

_____

_____

_____