SOFT 437
Quiz #3
March 19, 2015

Do not turn this page until the quiz officially begins.

STUDENT NUMBER ____ ____ ____ ____ ____ ____ ____ ____

Please do not write your name anywhere on this quiz. I recommend writing your student number at the top of each page.

You are expected to behave considerately towards your fellow students. Noisy or disruptive behavior will not be tolerated. **Turn your phone off.** If you finish early, you may quietly leave the room after handing in your quiz. **However, nobody will be permitted to leave during the last 10 minutes of the quiz.**

If you leave the quiz early, do not stand around outside the exam room talking. This is extremely distracting for students still working.

Academic dishonesty will not be tolerated. Keep your eyes on your own paper or the blackboard at the front of the room. You may not refer to other notes or books during the quiz. You may use a calculator during the quiz.

Please try to write your answers in the space provided. If you need to write your answers elsewhere, please indicate clearly where on the quiz your answers are to be found.

You have 40 **minutes** to complete the quiz. Good luck!

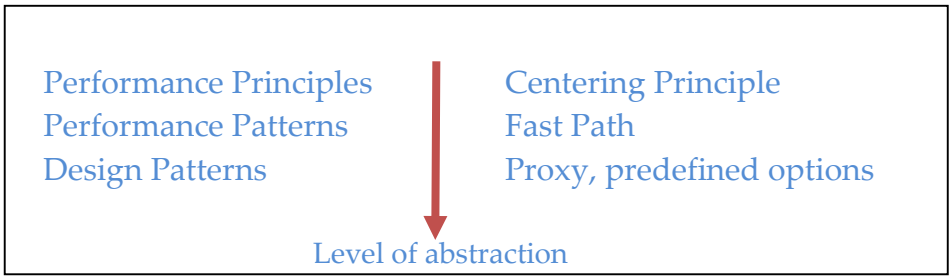
Question 1	/10
Question 2	/15
Question 3	/10
Total	/35

Student ID:

Question 1 **(10 Marks)**

What is the difference between *Performance Principles*, *Performance Patterns* and *Design Patterns*? How do the three concepts relate to each other (clarify you answer with an example)?

- Design pattern: is a common solution to a problem that occurs in many different contexts. Example: Proxy
- Performance Patterns: describe the best practices for producing responsive, scalable software. Example: Fast path
- Performance principles: provide generalized and abstracted knowledge used by experienced performance specialists when constructing software systems and Identify design alternatives likely to meet performance objectives. Example: Centering Principle



Each performance pattern is a realization of one or more of the performance principles. The performance patterns are at a higher level of abstraction than design patterns. A design pattern may provide an implementation of a performance pattern

Student ID:

Question 2 **(15 Marks)**

In multi-tier architectures of Web applications, the business logic often reside on one node, while the database resides on another, possibly far away.

1- Explain how the batching performance pattern can enhance the responsiveness and scalability of the system.
2- What is the difference between client-side batching and server-side batching?
3- When is batching inappropriate in these architectures?

1-Batching can combine database requests (reads, updates and inserts) into batches so the overhead processing is executed once for the entire batch instead of for each individual item. Batching can:

- Reduce the total amount of processing required for all tasks
- Improve responsiveness by reducing the contention delay
- Improve scalability by freeing up resources

2- Client-side batching (also called sender-side) may create a session to aggregate multiple requests (e.g., related inserts and updates) and send them at once to the database for a single commit command.

Server-side (also called receiver-side) may receive multiple requests and process them all at once. Example: back-order items. The server may collect missing items until a reasonable number is complete (or a time threshold is reached) and place a single purchase order for all missing items.

3- Batching is inappropriate if the database must be instantaneously consistent. Batching may compromise consistency. Also, batching is inappropriate for tasks with limited overhead. Batching is most appropriate when both the amount of overhead and frequency of requests are high.

Student ID:

Question 3 **(10 Marks)**

A software system receives and stores cartographic data (i.e., data used to generate maps) with a specific data structure. The system may receive cartographic data in various data structures. However, before data is stored, the system must transform the received data into a specific data structure (say *DataStruct1*). The developers are considering the following two transformation algorithms:

Algorithm 1:

```
    foreach item in ReceivedDataSet
      {
          check the item data structure
          if NOT comply with DataStruct1
          transform item into DataStruct1
      }
save item
```

Algorithm 2:

```
    upload grammars of DataStruct1 into memory
    upload grammars of potential data structures into memory
    upload transformation rules into memory
    read ReceivedDataSet
    foreach item in ReceivedDataSet
      {
          apply transformation rules on item
      }
    save newDataFormat
```

1- Compare the two algorithms in light of the independent performance principles (i.e., Centering, Fixing-point, Locality, and Processing versus Frequency).
2- Which algorithm is better (justify your choice)?

[please answer in the next page]

Student ID:

- Centering principle: the dominant workload is transformation. Both algorithms try to reduce the processing for the transformations.

- Fixing-point: Algorithm 2 is better since it establishes the fixing earlier using grammar and transformation rules.

- Locality: Algorithm 2 is better since the transformation is done in memory.

- Processing versus frequency: Algorithm 2 is better since the transformation is done once after loading the entire data set into the memory and generate the output at once.