2011 Paper 5 Question 8

Concurrent and Distributed Systems

(*a*) Consider a queue data structure, with the following interface:

```
interface Queue {
  void push(Object val);
  Object pop();
};
```

(*i*) Write pseudocode for a semaphore-based implementation of `Queue`. Your implementation should allow concurrent push and pop when it is safe to do so, but not when it is unsafe. The queued data should be stored in an array of fixed length n. [6 marks]

(*ii*) Explain the specific circumstances under which concurrent push and pop is unsafe. Explain how your solution in part (*i*) addresses these. [2 + 2 marks]

(*b*) You decide to use this data structure to manage a service, where people around the world push jobs of work to be done, and others pop jobs and do them. (This is a description of Amazon's Mechanical Turk.)

This could be implemented as a centralised service, offered through Object-Oriented Middleware.

(*i*) How can the Object-Oriented Middleware maximise the utilisation of this service? [2 marks]

(*ii*) Object-Oriented Middleware makes the calls to push and pop look local. Name **two** things that this hides from the programmer using the service and say why each is a problem. [4 marks]

(*iii*) What aspects, that are not part of the middleware, might hinder scalability in terms of the number of potential pushers and poppers? [4 marks]