Answer Set 2

23 FEBRUARY, 1993

1. How to do producer/consumer with two computers:

   The computer *Passaic* is the producer of files, and *Venice* is the consumer. Initially, they must agree on a sequence of file names, such as $F_1, F_2, \ldots$. Passaic keeps a counter i, initially 1, and creates and writes to F_i. When it is finished with file F_i, i is incremented by one. Venice waits until it sees file F_{i+1} before reading F_i. Once Venice is done with F_i it erases it.

   If disk space is limited, Passaic can wait until F_j disappears before creating F_i, where $j < i - 1$. The actual gap between i and j is set according to the amount of disk space one has.

2. Adapt Dekker's algorithm to a shared file configuration.

   The solution to this problems depends on the assumptions one makes about the nature of the shared file system. It would not be in the spirit of the problem to assume that the shared file system has the same operational semantics as shared memory: that would make the problem too easy. I will assume that file creation is atomic and files are tagged with the owner of the file.

   The computers Passaic and Venice will agree on three files: *flag-passaic*, *flag-venice* and *turn*. Comparing with "Larry's Correct Algorithm" on page 21 of the class text:

   - The file *flag-passaic* is analogous to the flag `flag-larry`. Passaic will create the file *flag-passaic* to signal "true" and delete the file to signal "false".
   - Likewise with the file *flag-venice*, the flag `flag-jim` and the computer Venice.
   - The variable `turn` in the algorithm will be replaced by the file *turn*. When Venice creates it, its owner is Venice, when Passaic creates it, its owner is Passaic. In this way, each computer can test the `turn` variable by looking at the ownership of the *turn* file.