Reliable & Secure Systems Design
ECE 422
Fall 2013
Project #1: Fault-Tolerant Systems
Due date: Monday Oct. 21, 1pm

Overview
In this project, we will explore the design & construction of fault-tolerant systems using Java and C. We will implement a fault-tolerant sorting mechanism using recovery blocks (see Chapter 4 of our textbook). 2 separate programs must be written: a data generator that creates random integer values and writes them to a file, and a sorting program that reads that file, sorts the values, and writes them to another file. The sorting program is to be fault-tolerant, with a primary sorting routine written in Java, and a backup sorting routine written in C. The C routine should be called as a native method from within Java. There will also be a small chance of either the main or backup sorting routine failing on any memory access operation; these are probabilities that must be passed as command-line arguments to the sorting program.

Failure Mode to be Considered
The type of failure we are going to protect against in this project is a transient hardware fault that results in the failure of a memory operation during the sorting routines. These faults are independent of one another, and occur at random times. This means that, any time a variable is accessed during your program, there is a small chance that you could have a failure. We are going to simulate this situation in the following way: within your two sorting routines, keep a count of the number of memory accesses that occur during execution. Multiply this number by the chance of a memory failure (the main and backup routines are to have different probabilities of failing), we will refer to this as the HAZARD. Now generate a random number in the range [0,1], and compare this number to the range [0.5, 0.5+HAZARD]. If the random number falls within that range, issue a failure (i.e. end the routine with a return value indicating failure). The probabilities of failure for the primary and backup variants are to be command-line arguments to your sorting program.

Requirements

Data Generator: The data generator must accept an output filename and the number of integer values to be generated as command-line arguments. Integer values are to be randomly generated. This program is to be written entirely in Java.

Data Sorter: The data sorter must accept input and output filenames, failure probabilities for the primary and backup variants, and a time limit as command-line arguments. All elements of the recovery block scheme discussed by Pullum in Chapter 4 must be implemented; the executive, primary & backup variants, the adjudicator, and the watchdog timer. The executive, primary & backup variants and the watchdog timer must be implemented as separate threads within your program; the adjudicator may be a part of the executive thread. The primary variant will be a Heapsort algorithm implemented in

Java; the backup routine is to be an Insertion sort algorithm, implemented as a native method in C. The stored checkpoint will simply be our input file. If the primary experiences a failure, fails to pass the adjudicator or times out, then you must restore the checkpoint, print an error message, and call the backup routine. If the backup experiences a failure, does not pass the adjudicator or times out, print a second error message and terminate the program. If the program terminates with a failure, the output file should be deleted before termination.

Submitting Your Programs

You must submit both design documents and source code to me before the deadline. The design documents will be a UML class diagram for both the Data Generator and the Data Sorter, as well as UML sequence diagrams for each of the three use cases for the Data Sorter: Primary variant succeeds, Primary fails but backup succeeds, and Both primary and backup variants fail.

There are 3 phases to the submission of your programs. First, all of all, your source code and design documents should be emailed to me (as a gzipped tar archive) by 1pm on the due date. Second, you must provide me with a hardcopy printout of your design documents in class on the due date. Finally, you must demo your program for me within one week after the deadline; this demo will take place in my office, where I will remotely log in to the software engineering laboratory (E5-005). Your program MUST run on the Linux machines in the lab to receive credit. You must email me for a demo time.

Special Note: There is no lab section for ECE 422, and so we do not have assigned lab hours. I will set up card-swipe access to the E5-005 laboratory for you. Please be aware that the lab may from time to time be closed for lab sections in other courses.

Grading

Designs	30%
Correct Operation	30%
Multithreaded implementation	20%
JNI implementation	20%