Dependable Systems
SS 2011

Assignment 3 (v1.0)

(Submission deadline: July 31th 2011, 23:59 CET)

Fault-Tolerant Software

In this assignment, it is your task to develop a fault-tolerant remote file server with the Simplex approach from Lui Sha.

You have to implement the high-performance-control subsystem (HPC), the high-assurance-control subsystem (HAC), and the decision logic as part of your server implementation. The whole server with all parts must be either written in Python >= 2.5, providing an XML-RPC interface, **or** in Java >= 1.6, providing an Java RMI interface.

Your HPC subsystem should implement all functionality, including the exclusive opening of written files. Your HAC subsystem should only provide a restricted read-only access to the data. The server must work as console application. The decision logic module must be able to switch between HPC and HAC subsystem without a restart of the server application. Some connected clients may loose connectivity in this process. All other clients should, as much as possible, not be influenced by the switch from HPC to HAC subsystem. The fail-back from HAC to HPC subsystem is not part of the assignment.

As **mandatory part** of the assignment, you must design your decision logic to detect a violation of the rules for *open()* calls. As **optional part** of the assignment, you should harden your system against main memory faults leading to data corruption in the running server instance. Beyond the pure code inspection done for grading, we will inject faults in your HPC module sources to test the reaction.

As solution, please submit a **ZIP file** containing the complete server sources. In the Java case, you must additionally provide a shell script called *build.sh* which rebuilds the Java server on a Linux system with *javac* installed. Typically command-line build systems (ANT, Maven, ...) as precondition for compilation are allowed.

For both languages, the server must be executable with the single command *run.sh* on a Linux system.

Please document the according compilation and execution guides, plus your architectural decisions, in a **separate** PDF file.

Python server interface

The remotely accessible interface for Python must have the following layout:

```
class SimpleFileServerInterface:
        """The interface for the simple file server. Only plain text files are supported,
           all stored in the same directory.
           The file server should assume a fixed character encoding for the text.
        """
        def create(name):
                """Creates a new empty file with the given name. The file is not opened.
                   If the file already exists and is not opened, it should be truncated.
                   If the file already exists and is already opened by another client,
                   nothing should happen.
                   There is no return value.
                """
                pass
        def open(name, mode):
                """Opens the given file. Only plain file names must be allowed.
                   Mode can be either 'r' for reading, or 'w+' for reading and writing.
                   Only one client must be able to have a file opened in 'w+' mode.
                   Multiple clients must be allowed to have the same file open in 'r' mode.
                   On success, the method returns a handle to the opened file, otherwise a None value.
                """
                pass
        def read(handle, start, numbytes):
                """Reads the given number of bytes from the given start position in the file referenced
                   by the given handle. The start index is 0.
                   On success, the method returns a string, otherwise a None value.
                """
                pass
        def write(handle, start, text):
                """Writes the text to the file referenced by the given handle,
                   from the position given by the start parameter.
                   There is no return value.
                """
                pass
        def close(handle):
                """Closes the open file referenced by the given handle.-
                   There is no return value.
                """
                pass
```

Java server interface

The Java version must provide the following interface, with the functional behavior as described in the Python version:

```
public interface SimpleFileServerInterface extends Remote {
    void create(String name) throws RemoteException;
    int open(String name, String mode) throws RemoteException;
    String read(int handle, int start, int numbytes) throws RemoteException;
    String write(int handle, int start, String text) throws RemoteException;
    void close(int handle) throws RemoteException;
}
```