15-440: Distributed Systems
Rubrics of Project 1

School of Computer Science
Carnegie Mellon University, Qatar
Fall 2011

Common Package (Points: 2%)

- Path class

RMI (Points: 53%)

- Multi-threaded servers (Points: 3%)
  - All servers/skeletons should be multi-threaded. No clients should wait while another client is waiting
- SkeletonTest(Points: 25%)
  - Skeleton unmarshalls variables and marshalls return values correctly (Points: 7%)
  - Skeleton links to the correct Class object (Points: 5%)
  - Skeleton dispatches to the correct method for any invocation class (Points: 10%)
  - Error/Exception checks (Points: 3%)
    - Trivial errors, such as passing null as parameters, should be rejected
    - Skeleton should reject RMI methods that do not throw RemoteException
    - Skeleton rejects non-matching class parameter and server object
- Stub Test (Points: 25%)
  - Stub connects to skeleton (Points: 7%)
  - Stub marshalls all types of arguments correctly and unmarshalls return result (Points: 3%)
  - Stub can be created at a remote IP address and port (Points: 5%)
  - Stub can be created with a given skeleton class (Points: 5%)
  - Error/Exception checks (Points: 5%)
    - Code should check if skeleton has started
    - Stub should be created only from an interface
    - Cases where interface does not throw RemoteException should be rejected
    - Trivial errors, such as passing null as parameters, should be rejected

Storage server (Points: 25%)

- NOTE: You should have used your own RMI to implement below stuff. Otherwise there will be no points awarded
- Registration Test(Points: 5%)
  - Storage server should register to the naming server
  - Storage server should prune empty directories
- FileAccessTest(Points: 10%)
  - Size test
    - For Valid files
    - For Invalid files
    - For null args (trivial args)

- Read test
  - Should not be able to read directories
  - Reading empty file
  - null args (trivial args)
- Write test
  - Write to non-existent file should be rejected
  - Write a directory should be rejected
  - null args for file and/or data (trivial args)
- Write-local read
  - The file written through client should be same as local file stored on disk
- Write-Read OutOfBounds
  - Read beyond file size should not be allowed
  - Reading negative lengths should not be allowed
  - Writing files with negative offset should not be permitted
- Append test

- Directory tests(Points: 5%)
  - File create Test
    - Should not be able to create root dir
    - File should be created directly under root directory
    - Creating directory with null args (trivial case) should not be allowed
    - Cannot create file/dir that is already present
    - Check if file is created with read/write perms
  - Delete test
    - Client should not be able to delete root
    - Client should not be able to delete file/dir with null args (trivial case)
    - Client should not be allowed to delete a non-existing file/dir
    - Prune empty directories

Naming server (Points: 15%)

- NOTE: You should have used your own RMI to implement below stuff. Otherwise there will be no points awarded
- Contact tests (Points: 2%)
  - Test if naming server has opened one port for registration and one for service
- Registration test(Points: 3%)
  - Cannot register with null args (trivial test)
  - Merge two storage server file meta data
- List files(Points: 3%)
  - Cannot list with null args (trivial test)
  - List non-existent file/dir
  - Test listing files/dirs
- Create files/directory (Points: 3%)
  - Cannot create file/dir with null args (trivial test)
  - Cannot create root directory
  - Cannot create file/dir under parent directory that does not exist
  - Cannot create a file/dir with an already existing file/dir
- Stub retrieval tests(Points: 4%)

- Trivial tests
- Start two storage servers with initial set of files. Retrieve the storage server stub from naming for those files. Check and see if they match

Code Style (Points 5%)

- Method Comments, Block comments, Readability, Dead code, Code design

Bonus (Points: 10%)

- Implementation of Delete functionality