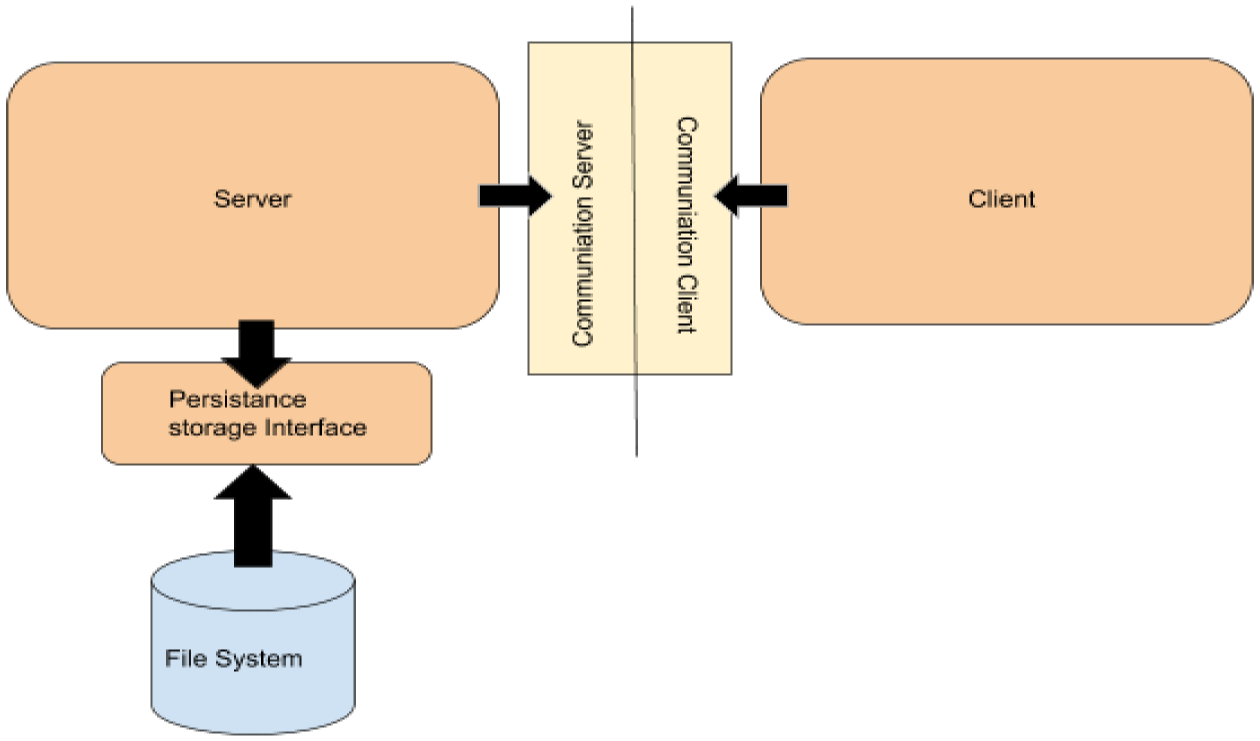
ECE419 Milestone 1 Design Document

Architecture Overview:



Persistence Storage Design

The persistence storage is being used as an interface to store the Key Value Pairs into a single file. This file keeps track of all the existing data. Therefore, when the server goes down, the data is still able to be retrieved. The current persistence storage design uses a JAVA property library which converts a local hashmap to a file. This implementation will be improved in the later milestones. The current design is capable of storing the Map and reading the key from the file in an asynchronous manner. This part will be replaced by file channels implementation which enables random access features in Java.

Communication Module Design

The team comes up with the design choice to further modularize the communication interface. Both client and server interact with the KVCommunication channel directly. This part of the system converts the string into byte arrays and processes the KVMessage packet through the protocol. This modulation further separates the functionality of the server and client. For both server and the client, the communication responsibility is distributed down to each of their separate communication modules.

Message Protocol Design

The KVMessage protocol serves at the core of KVCommunication channel between client and server. The protocol is designed to be lightweight, reliable and easy to use, which sends information over input and output streams using byte arrays. Each message contains three fields, which includes message status type, key, and value. Delimiters of “D” and “\n” are inserted in between, which are ASCII characters of “68” and “10” respectively in byte format.

Performance Evaluation

This component evaluates the performance of the system given different read and write requests ratios. We tested a total number of 1000 requests and the following table displays the final result of the performance.

Performance Report Chart

	put				get			
Payload	16B	512B	2048B	4096B	16B	512B	2048B	4096B
Performance/ Throughput (KB/s)	72.759022118 74272 KB/s	205.75708318 758873 KB/s	216.32028381 221235 KB/s	220.75664339 523718 KB/s	231.57797230 327452 KB/s	444.12861964 825015 KB/s	424.18265305 040353 KB/s	407.28013236 6043 KB/s
Average latency per 1000 request	214.0 ms	2430.0 ms	9245.0 ms	18119.0 ms	112.0 ms	1125.0 ms	4714.0 ms	9821.0 ms

	Put / Get Request ratio				
Allocation (put/get)	100% / 0%	80% / 20%	50% / 50%	20% / 80%	0% / 100%
Performance/ Throughput (KB/s)	60.78583933086948 KB/s	80.72362812884782 KB/s	104.73268866406892 KB/s	172.72191311396884 KB/s	267.20820863616933 KB/s
Average latency per 1000 request	257.0 ms	232.0 ms	223.0 ms	162.834 ms	113.0 ms

Appendix: Junit Test Case Explanation

The following tests focus mainly on the disk storage and the concurrency ability of the system. Some of the tests relate to performance testing and some of the test solely tests the persistence storage by itself.

Test Name	Test Disk Storage Sequence
Description	This test examines some of the basic sequence input and output of the persistence storage.

Test Name	Test Disk Storage Basic
Description	This test solely tests on the basic functionality of the existing get and put functionality of the disk storage.

Test Name	Test Multi-clients
Description	This test launches multiple KVClients (10 clients by default) that connect to the same KVServer. Then perform put and get operations synchronously to verify server and disk storage functionality with synchronization.

Test Name	Test Disk Storage Get Requests Stress Test
Description	The test applies stress tests on the disk storage alone to handle over 1000 get requests from the server end. This would ensure that there is no corruptions within the persistence storage.

Test Name	Test Empty Key
Description	This test sets the key to an empty string, and verifies the server can insert the key-value pair successfully.

Test Name	Test Max Length Key
Description	This test sets the key to a string that contains maximum allowed length of 20 bytes (characters), and verify put and get operations are successful. .

Test Name	Test Key Exceeding Max Length
Description	This test sets the key to a string that exceeds the maximum allowed length of 20 bytes (characters), and verify an exception is thrown.

Test Name	Test Empty Value

Description	This test ensures that a successful client input of empty value would be marked as a delete action for the process.

Test Name	Test Value Max Length
Description	This is a stress test that checks on the system meets the requirement of maximum length of 120 kbytes for the value constraint.

Test Name	Test Value Exceeding Max Length
Description	This is a stress test that checks on the system throws an exception when value exceeds maximum allowed length of 120 kbytes.

Test Name	Test Send Random Byte Array
Description	This test verifies that server-client message exchange handles random byte arrays, as a super set of character strings.

Test Name	Performance Test
Description	This test collects performance measurements for different allocation of put/ge]et, and the put and get performance test with different payload.