SWE 261P Project Part 3

Introduction

Structural testing, also known as white-box testing, involves the testing of the software's internal structures or workings. It differs from black-box testing by requiring knowledge of the internal paths, data structures, and coding mechanisms of the application. This form of testing is critical for several reasons:

1. Ensures Code Quality: By examining the internal workings of an application, structural testing helps identify hidden errors that might not surface during functional testing.
2. Improves Design: It encourages better program design and promotes code optimization, as developers must consider testability from the onset.
3. Facilitates Debugging: When a test fails, it's easier to pinpoint the issue within the code, given the tester's familiarity with the internal structures.

Coverage Tool

Before introducing new test cases, the coverage tool revealed certain areas within the project that lacked sufficient testing. Specifically, within the HTTP folder, method coverage stood at 47% (161 out of 337 methods), and line coverage was at 53% (826 out of 1553 lines). The client folder exhibited even lower coverage, with method coverage at 19% (98 out of 501 methods) and line coverage at a mere 10% (136 out of 1266 lines). These figures highlighted significant portions of the codebase that remained untested, posing a risk to the software's stability and reliability.

Despite the existing test suite, several areas within the codebase lacked coverage, including:

1. Certain error-handling routines within the server and client modules that were not triggered by the current tests.
2. Edge cases in the server's HTTP routing logic and client's internal transport mechanisms.
3. Specific utility functions and algorithms within the server plugins were not exercised by any test case.

New Test Cases

To address these gaps, new test cases were devised targeting critical yet previously untested functionalities:

1. Server Plugins (SPIClassIterator): Tests were introduced to validate the isParentClassLoader method, ensuring it accurately identifies parent-child and grandparent-grandchild class loader relationships and unrelated class loaders. These tests are crucial for verifying the class loading hierarchy, an essential aspect of the plugin system's functionality.
2. HTTP Route Statistics (HttpRouteStatsTests): New tests focused on the merging of statistical data from two HttpRouteStats instances, ensuring accurate aggregation of

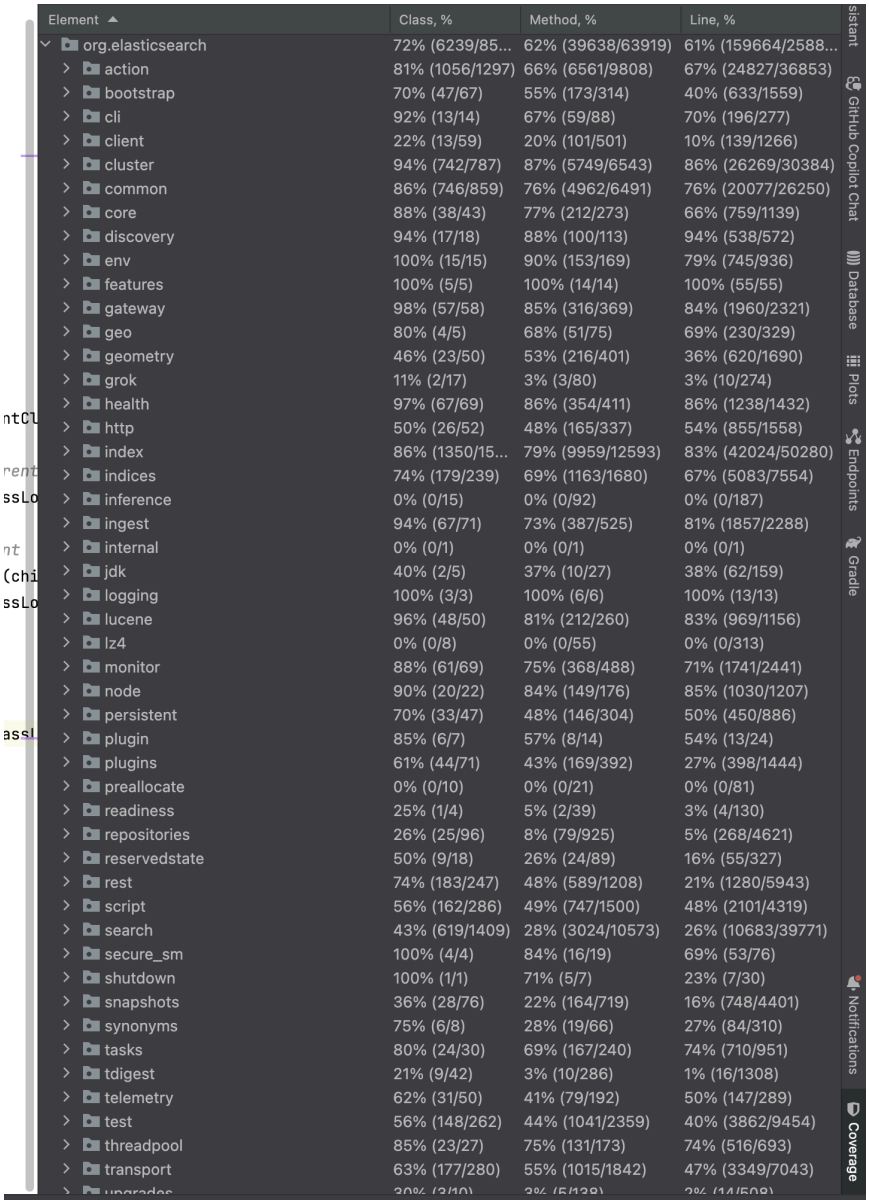
fields. Additional tests assessed the correctness of the equals and hashCode methods, which are fundamental for the integrity of statistical data management.

3. HTTP Route Stats Tracker: Tests were added to cover the tracking of HTTP requests and response statistics, including the aggregation of route-specific data. This is vital for monitoring and optimizing the performance of HTTP services.
4. Client Internal Transport (NoNodeAvailableException): The introduction of tests for various constructors and the status method of NoNodeAvailableException ensures proper error reporting and handling in scenarios where no nodes are available, a critical component of the client's resilience.

In summary, the following images show the metrics after adding the test cases, which suggests a significant increase in a particular test count or code metric.

After integrating these new test cases, coverage metrics improved noticeably. In the HTTP folder, method coverage increased slightly to 48% (165 out of 337 methods), and line coverage rose to 54% (855 out of 1558 lines). In the client folder, method coverage saw a modest rise to 20% (101 out of 501 methods), with line coverage incrementally improving to 10% (139 out of 1266 lines).

These enhancements to the test suite bolster the software's reliability by extending coverage and underscore the importance of continuous testing efforts in uncovering and addressing previously untested code paths and functionalities.



Element	Class, %	Method, %	Line, %
org.elasticsearch	72% (6239/85...	62% (39638/63919)	61% (159664/2588...
action	81% (1056/1297)	66% (6561/9808)	67% (24827/36853)
bootstrap	70% (47/67)	55% (173/314)	40% (633/1559)
cli	92% (13/14)	67% (59/88)	70% (196/277)
client	22% (13/59)	20% (101/501)	10% (139/1266)
cluster	94% (742/787)	87% (5749/6543)	86% (26269/30384)
common	86% (746/859)	76% (4962/6491)	76% (20077/26250)
core	88% (38/43)	77% (212/273)	66% (759/1139)
discovery	94% (17/18)	88% (100/113)	94% (538/572)
env	100% (15/15)	90% (153/169)	79% (745/936)
features	100% (5/5)	100% (14/14)	100% (55/55)
gateway	98% (57/58)	85% (316/369)	84% (1960/2321)
geo	80% (4/5)	68% (51/75)	69% (230/329)
geometry	46% (23/50)	53% (216/401)	36% (620/1690)
grok	11% (2/17)	3% (3/80)	3% (10/274)
health	97% (67/69)	86% (354/411)	86% (1238/1432)
http	50% (26/52)	48% (165/337)	54% (855/1558)
index	86% (1350/15...	79% (9959/12593)	83% (42024/50280)
indices	74% (179/239)	69% (1163/1680)	67% (5083/7554)
inference	0% (0/15)	0% (0/92)	0% (0/187)
ingest	94% (67/71)	73% (387/525)	81% (1857/2288)
internal	0% (0/1)	0% (0/1)	0% (0/1)
jdk	40% (2/5)	37% (10/27)	38% (62/159)
logging	100% (3/3)	100% (6/6)	100% (13/13)
lucene	96% (48/50)	81% (212/260)	83% (969/1156)
lz4	0% (0/8)	0% (0/55)	0% (0/313)
monitor	88% (61/69)	75% (368/488)	71% (1741/2441)
node	90% (20/22)	84% (149/176)	85% (1030/1207)
persistent	70% (33/47)	48% (146/304)	50% (450/886)
plugin	85% (6/7)	57% (8/14)	54% (13/24)
plugins	61% (44/71)	43% (169/392)	27% (398/1444)
preallocate	0% (0/10)	0% (0/21)	0% (0/81)
readiness	25% (1/4)	5% (2/39)	3% (4/130)
repositories	26% (25/96)	8% (79/925)	5% (268/4621)
reservedstate	50% (9/18)	26% (24/89)	16% (55/327)
rest	74% (183/247)	48% (589/1208)	21% (1280/5943)
script	56% (162/286)	49% (747/1500)	48% (2101/4319)
search	43% (619/1409)	28% (3024/10573)	26% (10683/39771)
secure_sm	100% (4/4)	84% (16/19)	69% (53/76)
shutdown	100% (1/1)	71% (5/7)	23% (7/30)
snapshots	36% (28/76)	22% (164/719)	16% (748/4401)
synonyms	75% (6/8)	28% (19/66)	27% (84/310)
tasks	80% (24/30)	69% (167/240)	74% (710/951)
tdigest	21% (9/42)	3% (10/286)	1% (16/1308)
telemetry	62% (31/50)	41% (79/192)	50% (147/289)
test	56% (148/262)	44% (1041/2359)	40% (3862/9454)
threadpool	85% (23/27)	75% (131/173)	74% (516/693)
transport	63% (177/280)	55% (1015/1842)	47% (3349/7043)

Before: 159355 -> Now: 169664.