**Report STV Iteration 2**Sjors Derksen (5939321)  
Stijn Schroevers (5995256)  
Rick Goossens (5915910)

Table of Contribution Iteration 1:

|  |  |  |
| --- | --- | --- |
| Sjors Derksen | Stijn Schroevers | Rick Goossens |
| 55% Fight() | Dungeon() | 45% Fight() |
| Pack() | Commands() | DistributePotions() |
| TestPlayer() | PopulateDungeon() |  |
|  |  |  |
| 25% Player Tests | Test Game & Dungeon | Test Node & Bridge |
| Test Pack & Items | Test Commands | 75% Player Tests |
|  |  | Distribute Potions Test |
| Effort% = 30% | Effort% = 40% | Effort% = 30% |

Table of Contribution Iteration 2:

|  |  |  |
| --- | --- | --- |
| Sjors Derksen | Stijn Schroevers | Rick Goossens |
| Turn structure | Specifications (structure + tests) | Distribute Items (Fix) |
| Move + Fight Updates | Recording Structure | UI |
| Pack Rules Implementation(70%) | Pack Rules Implementation(30%, fixes) |  |
|  |  |  |
| Bug Fixing | Making Replays (Files) | Making Replays (Files) |
|  | Bug Fixing | Bug Fixing |
| Effort% = 33% | Effort% = 33% | Effort% = 33% |

1. The general statistics of your implementation:

|  |  |
| --- | --- |
| N = total # classes | 27 (not counting Utils) |
| locs = total # lines of codes | 921 (still not counting Utils) |
| locsavg = average # lines of codes | 34.1 |
| Mavg = average # methods per class | 3.4 |
| Mmax = max # methods per class | 12 (Dungeon) |
| cabe = the total mcCabe complexity | 405 |
| cabeavg = average mcCabe complexity per class | 15 |

2.

**RZone:** Our two inputs are, move random or move to player. This is our first characteristic. In our second characteristic we have the blocks: Do not move to previous zone and do not move to next zone. Our playthroughs have all possible combinations of these two characteristics, so this gives us full PWC.

**RNode:** We have two characteristics. In the first characteristic, we have two blocks: Node is full and node is not full. Our second characteristic has 4 blocks: Move to node, Do not move to node, Flee to node and Do not flee to node. If we apply PWC on these characteristics, we get full coverage.

3. Statistics of your unit-testing eﬀort:

Unit Tests

|  |  |
| --- | --- |
| N0 = number of classes targeted by your unit-tests | 16 (GameCreationException is neither used nor covered) |
| T = number of test cases | 63 |
| Tlocs = total # lines of codes of your unit-tests | 987 |
| Tlocsavg = average # unit-tests’ lines of codes per target class | 61.7f |
| E = total time spent on writing and executing | 51 hours (rough estimate, we didn’t keep track. Roughly 17 hours/person) |
| Eavg = average eﬀort per target class | 3.2f |
| bugs = total number of bugs ever found by testing | 20 (rough estimate) |

* We fixed a couple of unit tests to work with some new implementations, but otherwise they are the same as in Iteration 1

System Tests

|  |  |
| --- | --- |
| Total # lines of code of testing infrastructure | 78 |
| Testing of **RZone**  T = total number of test scenarios  BBCov = achieved black box coverage  CCov = achieved code coverage | 7  100%  100% |
| Testing of **RNode**  T = total number of test scenarios  BBCov = achieved black box coverage  CCov = achieved code coverage | 7  80-100%\*  100% |
| Testing of **RAlert**  T = total number of test scenarios  CCov = achieved code coverage | 5  100% |
| Testing of **REndZone**  T = total number of test scenarios  CCov = achieved code coverage | 2  100% |
| Slocs = total # lines of code of specification  E = total time spent on constructing specifications and their tests  Slocsavg = average # lines of code per spec.  Eavg = average effort  Bugs = total number of bugs found | 47  12 hours (rough estimate, with roughly 20 to 30 hours finishing the game rules)  9.4  1  22 (mostly pretty small bugs) |

Overview of your unit-test strength.

|  |  |
| --- | --- |
| Test Class | Coverage% over implementation |
| MSTest\_Bridge | 100 |
| MSTest\_Commands | 100 |
| MSTest\_Dungeon | 97,86 |
| MSTest\_Game | 100 |
| MSTest\_Items | 100 |
| MSTest\_Node | 100 |
| MSTest\_Pack | 98,70 |
| MSTest\_Player | 97,54 |
| MSTest\_Predicates | 100 |

Average Coverage% = 98.74  
Failure to reach 100% coverage was caused in general by our use of rather specific test cases, which only cover a single path. This means that blocks that aren’t covered by one test case, will be covered by another. These will still be counted as missing coverage.  
Furthermore, some missing blocks consist of just the “}”s after Exception or break calls.