6 State machines

Now that we have monsters that can move, we should adjust the Engine such that the monsters are indeed activated. We first create a regression test suite for the engine as it exists at the moment (i.e., without monster moves), and then extend it.

Exercise 38 Analyze the description of the state machine from the Engine class as discussed in the design document (doc/pacman-design.txt). Create a UML state diagram from this description. Note that the actual implementation of the state machine can be inspected in the Engine class itself.

Exercise 39 Make a state-to state transition table for the PacMan state machine (Binder Chapter 7, Figure 7.9, p. 190-191).

Exercise 40 Create and describe a series of test case specifications that achieves transition coverage.

Exercise 41 Implement the test cases in JUnit in the EngineTest class.

Exercise 42 Finite state machine specifications are often incomplete. Omitted (state, stimulus) pairs are usually to be ignored. Identify test case specifications for all omitted pairs, and verify that these indeed are ignored, i.e., that the state machine does not secretly implement any sneak path (Binder Chapter 7, section 7.3.6, p. 223-228).

Exercise 43 Implement the sneak path test suite in JUnit.

Exercise 44 Analyze the coverage of your test suite using Emma, report and fix any missing cases.

Exercise 45 Extend the models, implementation and test suites to cater for monster moves as well.

Exercise 46 Re-analyze the coverage using Emma.