Wets Domain Integration Test Report

Andrew Mangogna

2025-12-01

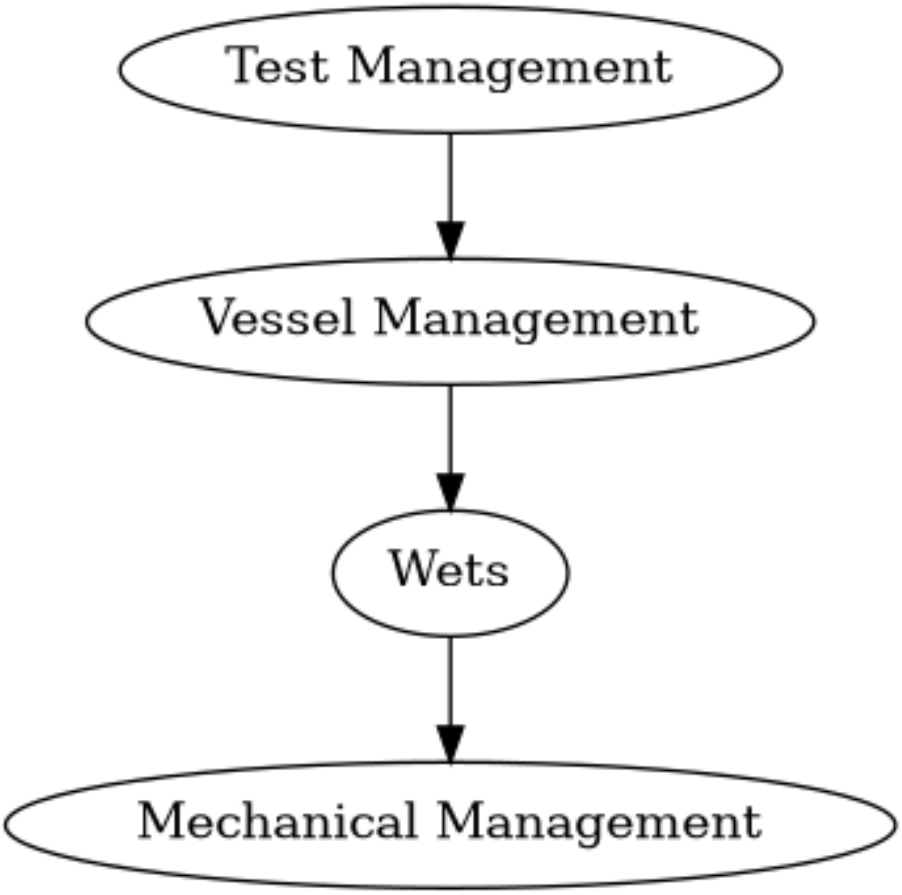
Introduction

This document contains a report on an integration test suite for the Wets domain. The Wets domain models the actions of canal locks as might be found on a river. The domain model was produced by Michael Lee as part of a case study project. This document shows how the translated Wets domain was tested in a simulated environment.

The Wets domain model was translated using Rosea. Rosea is a Tcl based model execution environment. Three other domains were constructed to complete the testing environment:

- The Mechanical Management domain provides services to the Wets domain for manipulating the gates, valves and flow sensors that are used to operate a lock.
- The Vessel Management domain provides services to request vessels to be transfer up or down the lock.
- The Test Management Domain orchestrates the running of the test cases.

The following diagram shows a domain chart of the testing environment.



The Testing Management domain requires the Vessel Management domain to make requests to transfer the vessels used for the various test cases. The Vessel Management domain requires the Wets domain to sequence the necessary controls to cause the requested vessel transfers to happen. The Wets domain requires the Vessel Management domain to move vessels between chambers in the lock. The Wets domain requires the Mechanical Management domain to operate the physical gates, values, and flow sensors to effect a transfer.

Test Management Domain

The Test Management domain is a realized domain. It is constructed using TclRAL. Since Rosea is implemented using TclRAL, a plain TclRAL implementation of Test Management easily interoperates with the Wets integration test environment. The Wets integration test environment is unaware of the Test Management domain and no special considerations were necessary to have the Test Management domain monitor and interact with the Rosea domain translations.

The following figure shows a class model for the Test Management Domain:

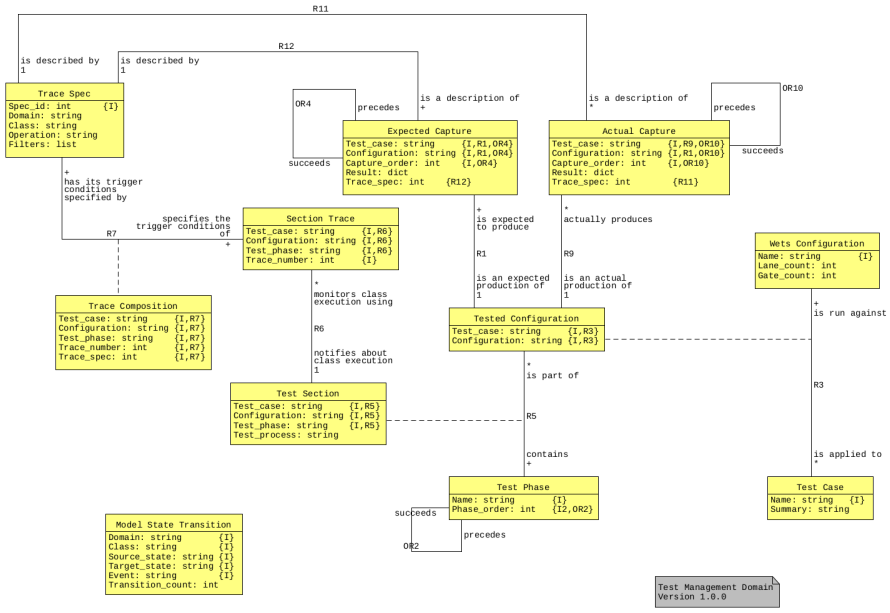


Figure 1: Test Management Class Model

The fundamental mechanism the Test Management domain uses to track the execution is the *relvar trace* facilities of TclRAL. Since the other domains are Rosea translations and since Rosea uses TclRAL relvars to store classes, the Test Management domain can use relvar tracing to keep track of the high order execution of the domains. This mechanism also works for trace state transitions, since Rosea stores the current state of class instances in a relvar.

The Test Managment domain considers the 11 test cases outlined in the *Testing Considerations* document and applies them to the 9 configurations of Wets classes also defined in the *Testing Considerations* document (*R3*). Each test run of a *Tested Configuration* is divided into one or more phases. The phases are:

Setup The setup phase establishes the pre-conditions for the test.

Trigger The trigger phase executes the necessary interactions with the modeled domains to cause the test to run.

Reset The reset phase causes the domains to complete any outstanding transfers and to cause the system to return to statis.

Finalize The finalize phase performs any remaining detailed work to ensure the system is in its original state.

For each test and each phase, there is specific code that executes to accomplish the goals of that phase. The Test Management domain will add relvar traces before the phase specific code executes and remove those traces at the end of

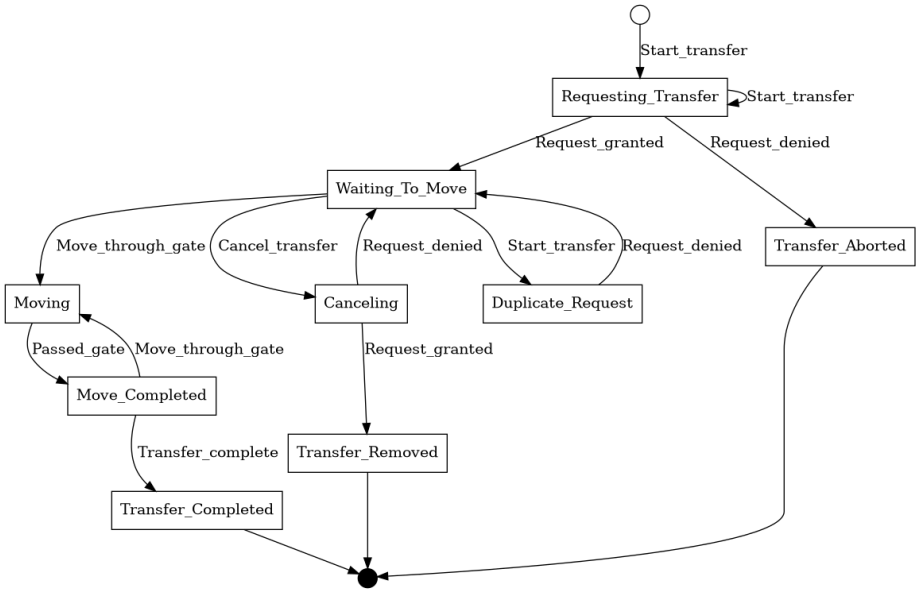
the section. The details of the relvar traces are specified and an expected result based on the trace is also specified. During test execution, the actual trace capture result is recorded. A test passes when the specified expected capture result matches the actual one.

Vessel Management Domain

The Vessel Management domain consists of only a single class, Vessel. An instance of Vessel is created each time a transfer request is made to the Wets domain.

Vessel Class State Model

The following figure shows the state model of the Vessel class.



The usual path through the state model is to asynchronously create a Vessel instance with the *Start transfer* creation event. The request is granted unless there is a conflict with the Vessel identifier. The Vessel instance then waits to fulfill move requests that appear as *Move through gate* events. The movement is simulated with a delayed event and eventually the *Passed gate* event is signaled. The machine bounces back and forth between the *Moving* and *Move Completed* state until the Wets domain completes the transfer, which is indicated by the *Transfer complete* event. Being a terminal state, the Vessel instance is deleted when the *Transfer complete* event is received.

The remainder of the states and transitions in the Vessel state model are used to test cancelation and duplication of Vessel transfer requests. These states are

necessary since separate granted and denied events are signaled as a result of any request made of the Wets domain.

Mechanical Management Domain

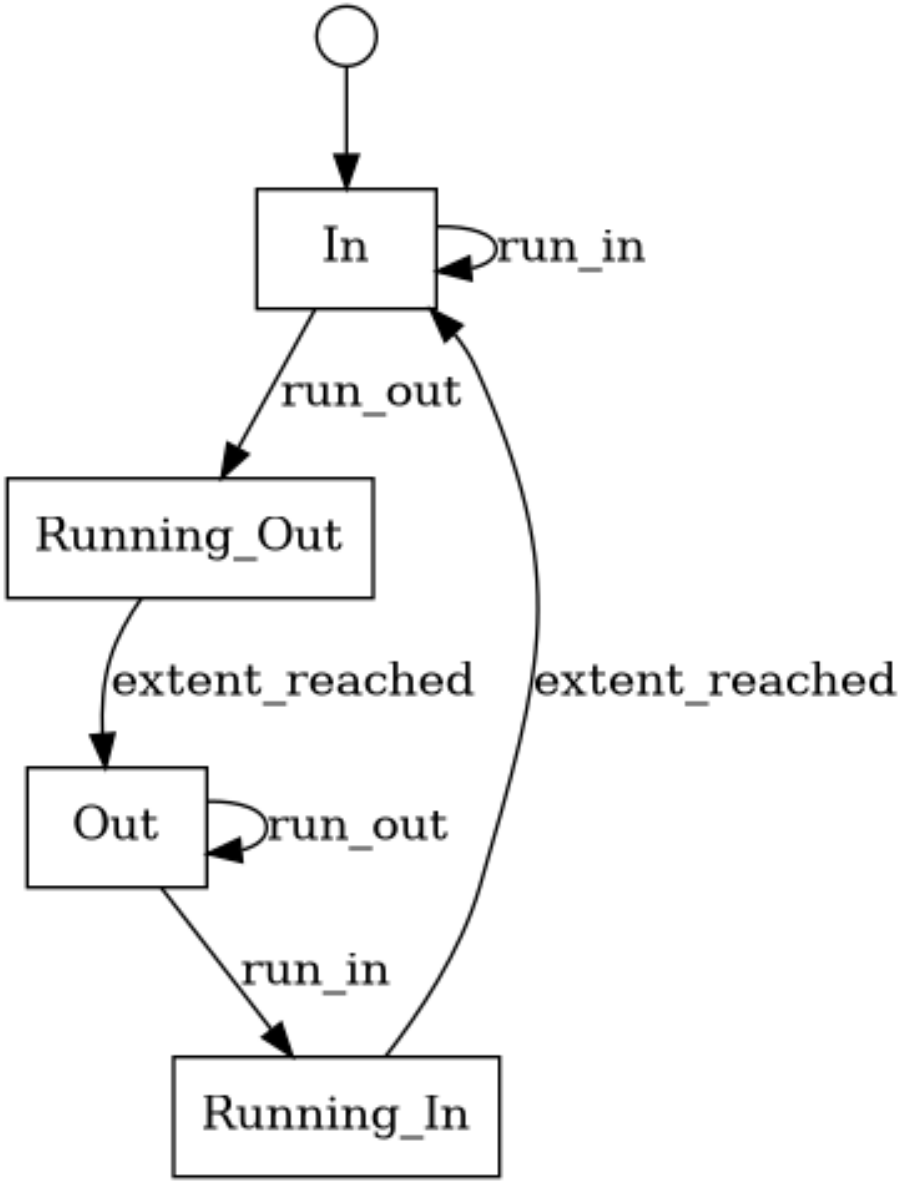
The Mechanical Management domain contains two classes:

Motor is an abstraction of an electric motor. A Motor is used to open and close mechanical devices. It is assumed that the Motor is attached to the necessary gears to accomplish the motion intended. The Motor is run in two direction. Running the Motor *out* opens the attached device. Running the Motor *in* closes it. Implicitly, the Motor has a mechanism to signal when when it has reached its furthest extent of running in either direction. The Mechanical Management domain uses Motors to open and close the lock gates and to open and close the culvert valves between chambers.

Flow Sensor is an abstraction of a transducer that can register fluid flow. The Flow Sensor is a simple transducer used to determine if there is any water flow between two chambers through their connecting culvert.

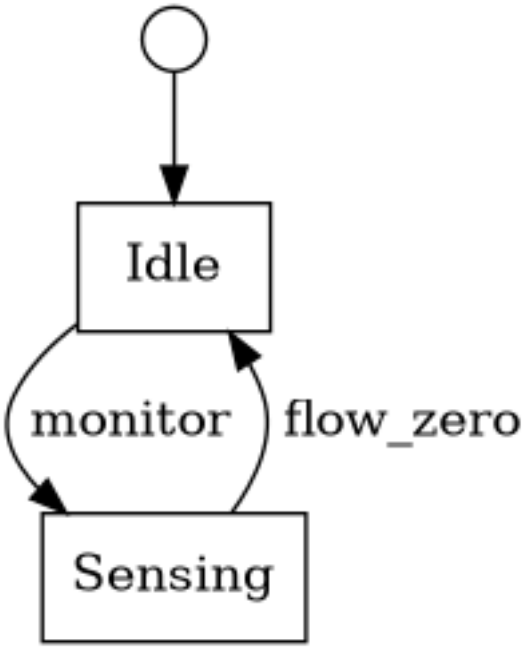
Motor Class State Model

The following figure shows the state model of the Motor class.



Flow Sensor State Model

The following figure shows the state model of the Flow Sensor class.



Test Results

The follow is a report generated by the test script:

```
[test_mgmt] [notice] 'PASSED: test_1 / Wets_1 (Vessel waits when no
transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_1 / Wets_3 (Vessel waits when no
transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_1 / Wets_4 (Vessel waits when no
transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_1 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_2 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_3 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_4 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_5 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_6 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_7 (Vessel requests up
transfer and an up transit lane is available)'
```

```
[test_mgmt] [notice] 'PASSED: test_2 / Wets_8 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_2 / Wets_9 (Vessel requests up
transfer and an up transit lane is available)'
[test_mgmt] [notice] 'PASSED: test_3 / Wets_1 (Vessel requests up
transfer when only down is available)'
[test_mgmt] [notice] 'PASSED: test_3 / Wets_4 (Vessel requests up
transfer when only down is available)'
[test_mgmt] [notice] 'PASSED: test_3 / Wets_9 (Vessel requests up
transfer when only down is available)'
[test_mgmt] [notice] 'PASSED: test_4 / Wets_1 (Vessel requests down
transfer when only up is available)'
[test_mgmt] [notice] 'PASSED: test_4 / Wets_4 (Vessel requests down
transfer when only up is available)'
[test_mgmt] [notice] 'PASSED: test_5 / Wets_1 (Vessel requests down
transfer when only down is available)'
[test_mgmt] [notice] 'PASSED: test_5 / Wets_4 (Vessel requests down
transfer when only down is available)'
[test_mgmt] [notice] 'PASSED: test_6 / Wets_1 (Duplicate transfer
request is denied)'
[test_mgmt] [notice] 'PASSED: test_7 / Wets_1 (Attempt a remove
transfer request and no Waiting Vessel matches)'
[test_mgmt] [notice] 'PASSED: test_8 / Wets_1 (Attempt a remove
transfer request with a matching Waiting Vessel)'
[test_mgmt] [notice] 'PASSED: test_9 / Wets_8 (Vessel must wait when
requesting an up transfer)'
[test_mgmt] [notice] 'PASSED: test_10 / Wets_6 (Vessel must wait when
requesting a down transfer)'
[test_mgmt] [notice] 'PASSED: ALL test cases'
[test_mgmt] [notice] 'PASSED: ALL state machine transitions taken'
```