Spencer West

Ryan Golden

Tyler Rose

CSCD 350

Project 5

**Test Sensor B.1: Standalone Sensor**

1.The rationale behind the test; i.e., what is it testing and why we care.

Answer: The purpose of this test is to have a minimal sensor object with no groups or mapper or watchdog. It is simple attached to the network and monitors that network. This is a position sensor that should be attached to an actuator to monitor the position of that actuator but the test is to examine if it is possible to attach a watchdog to the network.

2. A general English description of the initial conditions of the test.

Answer: the parser is giving the text command to create a sensor called mySensor1. The follow

creation commands are given as well such as a controller called myController1 that registers the sensor to it. The controller is a must for the sensor. The last three commands given are create a network with components of mySensor1 followed by a set command to set the value of the identification of the sensor to 35 and a command to get that value.

3. The commands for (2), which must appear in a standalone form that could be directly copied into a text file to reproduce the test without manual intervention. Do not cross-reference other tests.

CREATE SENSOR POSITION mySensor1

BUILD NETWORK WITH COMPONENTS mySensor1

CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

SET SENSOR mySensor1 VALUE 35

GET SENSOR mySensor1 VALUE

4. A brief English narrative of the expected results of executing the test. (Proper testing discipline expects that you do this before running the test.)

Answer: The parser will parse the sensor creation creating a sensor object of position called mySensor1. There is a network that needs to be created that the sensor will be attached to. After both the network and the sensor are created and attached the next command is to set a value to the sensor id and then retrieve it

5. At least one representation of the actual results. The form is your choice.

Welcome to your ParserHelper

Welcome to your Startup class

PARSE> @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

SCHEDULE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

PARSE> CREATE SENSOR POSITION mySensor1

SCHEDULE | CREATE SENSOR POSITION mySensor1

PARSE> BUILD NETWORK WITH COMPONENTS mySensor1

SCHEDULE | BUILD NETWORK WITH COMPONENTS mySensor1

PARSE> CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

SCHEDULE | CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

PARSE> SET SENSOR mySensor1 VALUE 35

SCHEDULE | SET SENSOR mySensor1 VALUE 35

PARSE> GET SENSOR mySensor1 VALUE

SCHEDULE | GET SENSOR mySensor1 VALUE

PARSE> @exit

SCHEDULE | @exit

TIME | 0.02

EXECUTE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

EXECUTE | CREATE SENSOR POSITION mySensor1

EXECUTE | BUILD NETWORK WITH COMPONENTS mySensor1

EXECUTE | CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

EXECUTE | SET SENSOR mySensor1 VALUE 35

EXECUTE | GET SENSOR mySensor1 VALUE

The value of Identifier{name=mySensor1} is 35.0

EXECUTE | @exit

EXITING |

6. A brief discussion on how the actual results differ from the expected results.

Answer: The result of the set and get value differ as the input was an integer for set however the output is a double. The sensor does attach to the network but doesn’t monitor anything but the network position. The attachment of the sensor to the network is otherwise successful.

7. A suggestion for how to extend this test to cover related aspects not required here. Your document must be formatted professionally. It must be consistent in all respects across.

Answer: The sensor could have other components such as watchdogs or mapper to apply a more strenuous test experience.

**Test C.1: Passthrough Mapper**

1. The rationale behind the test; i.e., what is it testing and why we care.

Answer: This is testing the creation of a passthrough mapper and assigning it to a position sensor, and then getting it’s value. We care about this because it shows that the passthrough mapper creation is working and that it gets assigned to a sensor.

2. A general English description of the initial conditions of the test.

Answer: The initial conditions are that there is no mapper, no sensor, no actuators, and no reporters. The mapper must be made before the sensor, the actuators need to be made before the reporter, and the reporter needs to be made before the sensor.

3. The commands for (2), which must appear in a standalone form that could be directly copied into a text file to reproduce the test without manual intervention. Do not cross-reference other tests.

@CONFIGURE LOG \"a.txt\" DOT SEQUENCE \"b.txt\" NETWORK \"c.txt\" XML \"d.txt\"

CREATE MAPPER myMapper EQUATION PASSTHROUGH

CREATE SENSOR POSITION mySensor1 MAPPER myMapper

CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

GET SENSOR mySensor1 VALUE

SET SENSOR mySensor1 VALUE 10

GET SENSOR mySensor1 VALUE

@exit

4. A brief English narrative of the expected results of executing the test. (Proper testing discipline expects that you do this before running the test.)

Answer: The expected results are for the GET SENSOR command to retrieve the value of the sensor and output it to the screen once before we set the value to see what the default is and once after to see how the mapper changed it, if it did. The expected result from passthrough is that the mapper doesn’t change the value at all so the value should be 0 at first, then 10 after it gets set.

5. At least one representation of the actual results. The form is your choice.

Welcome to your ParserHelper

Welcome to your Startup class

PARSE> @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

SCHEDULE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

PARSE> CREATE MAPPER myMapper EQUATION PASSTHROUGH

SCHEDULE | CREATE MAPPER myMapper EQUATION PASSTHROUGH

PARSE> CREATE SENSOR POSITION mySensor1 MAPPER myMapper

SCHEDULE | CREATE SENSOR POSITION mySensor1 MAPPER myMapper

PARSE> CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

SCHEDULE | CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

PARSE> GET SENSOR mySensor1 VALUE

SCHEDULE | GET SENSOR mySensor1 VALUE

PARSE> SET SENSOR mySensor1 VALUE 10

SCHEDULE | SET SENSOR mySensor1 VALUE 10

PARSE> GET SENSOR mySensor1 VALUE

SCHEDULE | GET SENSOR mySensor1 VALUE

PARSE> @exit

SCHEDULE | @exit

TIME | 0.02

EXECUTE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

EXECUTE | CREATE MAPPER myMapper EQUATION PASSTHROUGH

EXECUTE | CREATE SENSOR POSITION mySensor1 MAPPER myMapper

EXECUTE | CREATE CONTROLLER FORWARDING myController1 WITH COMPONENTS mySensor1

EXECUTE | GET SENSOR mySensor1 VALUE

The value of Identifier{name=mySensor1} is 0.0

EXECUTE | SET SENSOR mySensor1 VALUE 10

EXECUTE | GET SENSOR mySensor1 VALUE

The value of Identifier{name=mySensor1} is 10.0

EXECUTE | @exit

EXITING |

6. A brief discussion on how the actual results differ from the expected results.

Answer: The value was not affected by the mapper at all, which is expected as the passthrough mapper wasn’t supposed to change the value.

7. A suggestion for how to extend this test to cover related aspects not required here

Answer: Reporter or Watchdog could be added in to further test how mapper alters the value.

**Task C.2: Scaled Mapper**

1. The rationale behind the test; i.e., what is it testing and why we care.

Answer: This is testing the creation of a scaled mapper and assigning it to a position sensor, and then getting it’s value. We care about this because it shows that the scaled mapper creation is working and that it gets assigned to a sensor.

2. A general English description of the initial conditions of the test.

Answer: The initial conditions are that there is no mapper, no sensor, no actuators, and no reporters. The mapper must be made before the sensor, the actuators need to be made before the reporter, and the reporter needs to be made before the sensor.

3. The commands for (2), which must appear in a standalone form that could be directly copied into a text file to reproduce the test without manual intervention. Do not cross-reference other tests.

@CONFIGURE LOG \"a.txt\" DOT SEQUENCE \"b.txt\" NETWORK \"c.txt\" XML \"d.txt\"

CREATE MAPPER myMapper2 EQUATION SCALE 10

CREATE SENSOR POSITION mySensor2 MAPPER myMapper2

CREATE CONTROLLER FORWARDING myController2 WITH COMPONENTS mySensor2

GET SENSOR mySensor2 VALUE

SET SENSOR mySensor2 VALUE 1

GET SENSOR mySensor2 VALUE

@exit

4. A brief English narrative of the expected results of executing the test. (Proper testing discipline expects that you do this before running the test.)

Answer: The expected results are for the GET SENSOR command to retrieve the value of the sensor and output it to the screen once before we set the value to see what the default is and once after to see how the mapper changed it, if it did. The expected result is that the default value is 0, and then after we set the value to 1, it should be 10 as we are scaling it by 10. So the value should be multiplied by 10.

5. At least one representation of the actual results. The form is your choice.

Welcome to your ParserHelper

Welcome to your Startup class

PARSE> @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

SCHEDULE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

PARSE> CREATE MAPPER myMapper2 EQUATION SCALE 10

SCHEDULE | CREATE MAPPER myMapper2 EQUATION SCALE 10

PARSE> CREATE SENSOR POSITION mySensor2 MAPPER myMapper2

SCHEDULE | CREATE SENSOR POSITION mySensor2 MAPPER myMapper2

PARSE> CREATE CONTROLLER FORWARDING myController2 WITH COMPONENTS mySensor2

SCHEDULE | CREATE CONTROLLER FORWARDING myController2 WITH COMPONENTS mySensor2

PARSE> GET SENSOR mySensor2 VALUE

SCHEDULE | GET SENSOR mySensor2 VALUE

PARSE> SET SENSOR mySensor2 VALUE 1

SCHEDULE | SET SENSOR mySensor2 VALUE 1

PARSE> GET SENSOR mySensor2 VALUE

SCHEDULE | GET SENSOR mySensor2 VALUE

PARSE> @exit

SCHEDULE | @exit

TIME | 0.02

EXECUTE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

EXECUTE | CREATE MAPPER myMapper2 EQUATION SCALE 10

EXECUTE | CREATE SENSOR POSITION mySensor2 MAPPER myMapper2

EXECUTE | CREATE CONTROLLER FORWARDING myController2 WITH COMPONENTS mySensor2

EXECUTE | GET SENSOR mySensor2 VALUE

The value of Identifier{name=mySensor2} is 0.0

EXECUTE | SET SENSOR mySensor2 VALUE 1

EXECUTE | GET SENSOR mySensor2 VALUE

The value of Identifier{name=mySensor2} is 10.0

EXECUTE | @exit

EXITING |

6. A brief discussion on how the actual results differ from the expected results.

Answer: The value of Sensor was unchanged at 0 as 0\*10 = 0, and when it was changed to 1 then it became 10, which was expected.

7. A suggestion for how to extend this test to cover related aspects not required here

Answer: Reporter or Watchdog could be added in to further test how mapper alters the value.

**Task C.3: Normalized Mapper**

1. The rationale behind the test; i.e., what is it testing and why we care.

Answer: This is testing the creation of a normalized mapper and assigning it to a position sensor, and then getting it’s value. We care about this because it shows that the normalized mapper creation is working and that it gets assigned to a sensor.

2. A general English description of the initial conditions of the test.

Answer: The initial conditions are that there is no mapper, no sensor, no actuators, and no reporters. The mapper must be made before the sensor, the actuators need to be made before the reporter, and the reporter needs to be made before the sensor.

3. The commands for (2), which must appear in a standalone form that could be directly copied into a text file to reproduce the test without manual intervention. Do not cross-reference other tests.

@CONFIGURE LOG \"a.txt\" DOT SEQUENCE \"b.txt\" NETWORK \"c.txt\" XML \"d.txt\"

CREATE MAPPER myMapper3 EQUATION NORMALIZE 10 20

CREATE SENSOR POSITION mySensor3 MAPPER myMapper3

CREATE CONTROLLER FORWARDING myController3 WITH COMPONENTS mySensor3

GET SENSOR mySensor3 VALUE

SET SENSOR mySensor3 VALUE 15

GET SENSOR mySensor3 VALUE

@exit

4. A brief English narrative of the expected results of executing the test. (Proper testing discipline expects that you do this before running the test.)

Answer: The expected results are for the GET SENSOR command to retrieve the value of the sensor and output it to the screen once before we set the value to see what the default is and once after to see how the mapper changed it, if it did. The expected value at the default is 0 as 0 lies outside the bounds of 10 and 20 so it is 0. After it’s set to 15 it should return a value of 50 as 50 is halfway between 10 and 20.

5. At least one representation of the actual results. The form is your choice.

Welcome to your ParserHelper

Welcome to your Startup class

PARSE> @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

SCHEDULE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

PARSE> CREATE MAPPER myMapper3 EQUATION NORMALIZE 10 20

SCHEDULE | CREATE MAPPER myMapper3 EQUATION NORMALIZE 10 20

PARSE> CREATE SENSOR POSITION mySensor3 MAPPER myMapper3

SCHEDULE | CREATE SENSOR POSITION mySensor3 MAPPER myMapper3

PARSE> CREATE CONTROLLER FORWARDING myController3 WITH COMPONENTS mySensor3

SCHEDULE | CREATE CONTROLLER FORWARDING myController3 WITH COMPONENTS mySensor3

PARSE> GET SENSOR mySensor3 VALUE

SCHEDULE | GET SENSOR mySensor3 VALUE

PARSE> SET SENSOR mySensor3 VALUE 15

SCHEDULE | SET SENSOR mySensor3 VALUE 15

PARSE> GET SENSOR mySensor3 VALUE

SCHEDULE | GET SENSOR mySensor3 VALUE

PARSE> @exit

SCHEDULE | @exit

TIME | 0.02

EXECUTE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

EXECUTE | CREATE MAPPER myMapper3 EQUATION NORMALIZE 10 20

EXECUTE | CREATE SENSOR POSITION mySensor3 MAPPER myMapper3

EXECUTE | CREATE CONTROLLER FORWARDING myController3 WITH COMPONENTS mySensor3

EXECUTE | GET SENSOR mySensor3 VALUE

The value of Identifier{name=mySensor3} is 0.0

EXECUTE | SET SENSOR mySensor3 VALUE 15

EXECUTE | GET SENSOR mySensor3 VALUE

The value of Identifier{name=mySensor3} is 50.0

EXECUTE | @exit

EXITING |

6. A brief discussion on how the actual results differ from the expected results.

Answer: The actual results is that at the default it remained 0 then after we set it to 15 it turned into 50.0 which was the expected value.

7. A suggestion for how to extend this test to cover related aspects not required here.

Answer: Reporter or Watchdog could be added in to further test how mapper alters the value.

**Test E.1 Instantaneous Band Watchdog**

1. The rationale behind the test; i.e., what is it testing and why we care.

Answer: The purpose of this the band watchdog is to create a standalone watchdog that is not attached to any other component. There is no value that it monitors for exceed the high or low value that it checks. There is also an invalid command passed to the parser which is to check if an invalid command is passed will it construct that invalid watchdog

2. A general English description of the initial conditions of the test.

Answer: The parser takes the command to create an instantaneous band watchdog object with a high and low value. An invalid instantaneous band watchdog command will also be present that will try to create the band watchdog with no high or low value to monitor.

3. The commands for (2), which must appear in a standalone form that could be directly copied into a text file to reproduce the test without manual intervention. Do not cross-reference other tests.

//valid  
CREATE WATCHDOG BAND myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3  
//invalid  
CREATE WATCHDOG BAND myWatchdog1 INSTANTANEOUS

4. A brief English narrative of the expected results of executing the test. (Proper testing discipline expects that you do this before running the test.)

Answer: The result is that the valid command for instantaneous band watchdog will be created with its values. There is nothing else for the watchdog to do as it is a standalone and not attached to any other components and no value that is monitoring. The invalid should throw an error since it does not have all the required values such as low and high to monitor.

5. At least one representation of the actual results. The form is your choice.

Welcome to your ParserHelper

Welcome to your Startup class

PARSE> @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

SCHEDULE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

PARSE> CREATE WATCHDOG BAND myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3

SCHEDULE | CREATE WATCHDOG BAND myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3

PARSE> CREATE WATCHDOG BAND myWatchdog1 INSTANTANEOUS

SCHEDULE | CREATE WATCHDOG BAND myWatchdog1 INSTANTANEOUS

PARSE> @exit

SCHEDULE | @exit

TIME | 0.02

EXECUTE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

EXECUTE | CREATE WATCHDOG BAND myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3

EXECUTE | CREATE WATCHDOG BAND myWatchdog1 INSTANTANEOUS

EXECUTE | @exit

EXITING |

6. A brief discussion on how the actual results differ from the expected results.

Answer: I expected an error to be throw at the invalid case of the watchdog without a high or low value for the watchdog to monitor. The parser did accept the command though for the invalid case but both the high and low values are 0

7. A suggestion for how to extend this test to cover related aspects not required here. Your document must be formatted professionally. It must be consistent in all respects across

Answer: Having the watchdog attached to an actuator and a command to move the position of the actuator below the low value or above the high value. This would test if the value passed to the watchdog will trigger if the high or low are infringed.

**Test E.2 Instantaneous Acceleration Watchdog**

1. The rationale behind the test; i.e., what is it testing and why we care.

Answer: This test of an instantaneous acceleration watchdog is to implement a watchdog that is standalone. That acceleration value will be check by the low and high values that accompany the construction of the watchdog.

2. A general English description of the initial conditions of the test.

Answer: The parse is given the command to create a valid version of an instantaneous acceleration watchdog and an invalid version of an instantaneous acceleration watchdog. The valid version has both a high and low value attached to it. While the invalid version does not and thus should check for nothing

3. The commands for (2), which must appear in a standalone form that could be directly copied into a text file to reproduce the test without manual intervention. Do not cross-reference other tests.

//valid

CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3 //invalid

CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS

4. A brief English narrative of the expected results of executing the test. (Proper testing discipline expects that you do this before running the test.)

Answer: The watchdog should be created and its values both low and high with an optional grace value that is not a part of this test will be registered while the invalid should throw an error. There is no acceleration that the watchdog is examining so the watchdog is just a standalone watchdog.

5. At least one representation of the actual results. The form is your choice.

Welcome to your ParserHelper

Welcome to your Startup class

PARSE> @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

SCHEDULE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

PARSE> CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3

SCHEDULE | CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3

PARSE> CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS

SCHEDULE | CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS

PARSE> @exit

SCHEDULE | @exit

TIME | 0.02

EXECUTE | @CONFIGURE LOG "a.txt" DOT SEQUENCE "b.txt" NETWORK "c.txt" XML "d.txt"

EXECUTE | CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS THRESHOLD LOW 1 HIGH 3

EXECUTE | CREATE WATCHDOG ACCELERATION myWatchdog1 MODE INSTANTANEOUS

EXECUTE | @exit

EXITING |

6. A brief discussion on how the actual results differ from the expected results.

Answer: The result did not throw an error when creating the invalid version of the watchdog. Instead, the watchdog is created with a low and high value of 0. In a more extensive test this would mean that if the acceleration exceeds zero in the positive or negative direction the watchdog will trigger.

7. A suggestion for how to extend this test to cover related aspects not required here. Your document must be formatted professionally. It must be consistent in all respects across.

Answer:The test should have the watchdog attached to another component like an actuator to examine if the construction of the watchdog is correct and if it is monitoring the acceleration of the actuator.