

Library Extension Assignment

DT228/DT282

Software Engineering 2

**Shane Buckley**

**C20703429**

School of Computer Science

Technological University Dublin

**04/05/2022**

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:



\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Shane Buckley

04/05/2022

Table of Contents

*1.* Overview 4

*2.* Class Diagram 4

*3.* Object Diagram 5

*4.* Sequence diagrams 5

*5.* Constraints 6

*6.* TESTING 7

*7.* source cODE 8

# Overview

For this assignment I chose the “Extend and test a more comprehensive USE model for the Library system in USE” option. The use case I have developed for the Library system is the reserve book use case and includes constraints to ensure that when a book is being reserved, that it hasn’t been reserved or borrowed already, and also if a member has reserved a book that they are trying to borrow. I have implemented statemachines to handle the status of a copy. Included with my code is a SOIL file containing some sample SOIL commands to create member, book and copy objects for testing and demonstration purposes.

# Class Diagram

A picture containing diagram

Description automatically generated

# Object Diagram

A picture containing text

Description automatically generated

# Sequence Diagrams

Diagram

Description automatically generated

A picture containing diagram

Description automatically generated

# Constraints

Text

Description automatically generated

# Testing

When a member attempts to reserve a book that has already been reserved by another member:

Text

Description automatically generated

A member successfully borrowing a book they have reserved:

A picture containing text

Description automatically generated

A member trying to borrow a book they have not reserved:

Text

Description automatically generated

# Source Code

**StateMachines.use**

model Library

class Book

  attributes

    title : String

    author : String

    no\_copies : Integer

    no\_onshelf : Integer

  operations

    borrow()

    begin

        self.no\_onshelf := self.no\_onshelf - 1

    end

    pre copiesOnShelf: no\_copies >0

    post: no\_onshelf = no\_onshelf@pre - 1

    return()

    begin

        self.no\_onshelf := self.no\_onshelf + 1

    end

    reserve()

    begin

        self.no\_onshelf := self.no\_onshelf - 1

    end

end

class Copy

  attributes

    status : String

  operations

    borrow( m : Member)

    begin

        self.status := 'onLoan';

        self.book.borrow()

    end

    return( m : Member)

    begin

        self.status := 'onShelf';

        self.book.return()

    end

    reserve( m : Member)

    begin

        self.status:= 'isReserved';

        self.book.reserve();

    end

  statemachines

    psm States

    states

        newCopy : initial

        available   [status = 'onShelf']

        taken       [status = 'onLoan']

        reserved    [status = 'isReserved']

    transitions

        newCopy -> available { create }

        available -> taken  { borrow() }

        available -> reserved { reserve() }

        reserved -> taken { borrow() }

        taken -> available { return() }

    end

end

class Member

  attributes

    name : String

    address : String

    no\_onloan : Integer

    status : String

    fine : Integer

  operations

    borrow(c : Copy)

    begin

        insert (self, c) into HasBorrowed;

        self.no\_onloan := self.no\_onloan + 1;

        c.borrow(self);

    end

    return( c: Copy)

    begin

        self.no\_onloan := self.no\_onloan - 1;

        c.return(self);

        delete (self, c) from HasBorrowed;

    end

    reserve( c: Copy)

    begin

        insert (self, c) into HasReserved;

        c.reserve(self);

    end

end

association HasBorrowed between

    Member[0..1] role borrower

    Copy[\*] role borrowed

end

association CopyOf between

    Copy[1..\*] role copies

    Book[1] role book

end

association HasReserved between

    Member[0..1] role reserver

    Copy[\*] role copy

end

constraints

context Member::borrow(c:Copy)

    pre limit: self.no\_onloan < 1

    pre: self.borrowed->excludes(c)

    pre: c.status = 'onShelf' or self.copy->includes(c)

    post: c.status = 'onLoan'

    post: self.borrowed->includes(c)

context Member::reserve(c:Copy)

    pre: c.status = 'onShelf'

    post: self.copy->includes(c)

    post: c.status = 'isReserved'

context Member::return(c:Copy)

    pre: c.status = 'onLoan'

    pre: self.borrowed->includes(c)

    post: c.status = 'onShelf'

**Lib.use**

model Library

class Book

  attributes

    title : String

    author : String

    no\_copies : Integer

    no\_onshelf : Integer

  operations

    borrow()

    begin

        self.no\_onshelf := self.no\_onshelf - 1

    end

    pre copiesOnShelf: no\_copies >0

    post: no\_onshelf = no\_onshelf@pre - 1

    return()

    begin

        self.no\_onshelf := self.no\_onshelf + 1

    end

end

class Copy

  attributes

    onLoan : Boolean

    reserved : Boolean

  operations

    borrow( m : Member)

    begin

        self.onLoan := true;

        self.book.borrow()

    end

    return( m : Member)

    begin

        self.onLoan := true;

        self.book.return()

    end

end

class Member

  attributes

    name : String

    address : String

    no\_onloan : Integer

    status : String

    fine : Integer

  operations

    borrow(c : Copy)

    begin

        insert (self, c) into HasBorrowed;

        self.no\_onloan := self.no\_onloan + 1;

        c.borrow(self);

    end

    return( c: Copy)

    begin

        self.no\_onloan := self.no\_onloan - 1;

        c.return(self);

        delete (self, c) from HasBorrowed;

    end

    reserve( c: Copy)

    begin

        insert (self, c) into HasReserved;

        c.reserved:= true;

    end

end

association HasReserved between

    Member[0..1] role reserver

    Copy[\*] role copy

end

association HasBorrowed between

    Member[0..1] role borrower

    Copy[\*] role borrowed

end

association CopyOf between

    Copy[1..\*] role copies

    Book[1] role book

end

constraints

context Member::return(c:Copy)

    pre: c.onLoan = true

    pre: self.borrowed->includes(c)

context Member::reserve(c:Copy)

    pre: c.reserved = false

    pre: c.onLoan = false

    post: self.copy->includes(c)

    post: c.reserved = true

context Member::borrow(c:Copy)

    pre limit: self.no\_onloan < 1

    pre: self.borrowed->excludes(c)

    pre: c.reserved = false or self.copy->includes(c)

    pre: c.onLoan = false

    post: c.onLoan = true

    post: self.borrowed->includes(c)

**Lib.soil**

-- Script generated by USE 4.1.1

!new Member('James')

!James.no\_onloan := 0

!new Book('Fear\_And\_Loathing\_In\_Las\_Vegas')

!Fear\_And\_Loathing\_In\_Las\_Vegas.title := 'Fear\_And\_Loathing\_In\_Las\_Vegas'

!Fear\_And\_Loathing\_In\_Las\_Vegas.author := 'HunterSThompson'

!new Copy('c1')

!c1.reserved := false

!c1.onLoan := false

!insert(c1,Fear\_And\_Loathing\_In\_Las\_Vegas) into CopyOf

!new Copy('c2')

!c2.onLoan := false

!c2.reserved := false

!insert(c2,Fear\_And\_Loathing\_In\_Las\_Vegas) into CopyOf

!insert (c1,Fear\_And\_Loathing\_In\_Las\_Vegas) into CopyOf

!insert (c2,Fear\_And\_Loathing\_In\_Las\_Vegas) into CopyOf

!Fear\_And\_Loathing\_In\_Las\_Vegas.no\_copies := 2

!Fear\_And\_Loathing\_In\_Las\_Vegas.no\_onshelf := 2

!new Member('Conor')

!Conor.no\_onloan := 0

!new Copy('c3')

!c3.reserved := false

!c3.onLoan := false

!insert(c3,Fear\_And\_Loathing\_In\_Las\_Vegas) into CopyOf

!new Book('Alice\_In\_WonderLand')

!WandP.author := 'LewisCarroll'

!WandP.title := 'Alice\_In\_WonderLand'

!Conor.reserve(c2)

--!James.reserve(c2)

--!James.borrow(c2)

!Conor.borrow(c2)