

**Formal Modeling of the Web Summit in VDM++**

Mestrado Integrado em Engenharia Informática e Computação

Métodos Formais em Engenharia de Software

4º ano 1º Semestre

**Autores:**

Andreia Rodrigues, up201404691

Inês Gomes, up201405778

3 Janeiro 2018

# Índice

[**Índice**](#_gjdgxs) **1**

[**Descrição Informal do Sistema e Descrição da Lista de Requisitos**](#_1fob9te) **2**

[Descrição Informal do Sistema](#_3znysh7) 2

[Lista de Requisitos](#_2et92p0) 2

[**Modelo UML**](#_tyjcwt) **4**

[Modelo de Casos de Uso](#_3dy6vkm) 4

[Modelo de Classes](#_1t3h5sf) 8

[**Modelo Formal VDM++**](#_4d34og8) **10**

[Classe Attendee](#_2s8eyo1) 10

[Classe Common](#_3rdcrjn) 11

[Classe Company](#_26in1rg) 11

[Classe Conference](#_lnxbz9) 12

[Classe Exhibit](#_35nkun2) 15

[Classe Influential](#_1ksv4uv) 17

[Classe New](#_44sinio) 18

[Classe Startup](#_6mstxoytcm7a) 19

[Classe Talk](#_2jxsxqh) 21

[Classe WebSummit](#_z337ya) 24

[Classe Utilities](#_1y810tw) 29

[**Validação do Modelo**](#_922yt1crwjal) **32**

[Classe MyTestClass](#_4i7ojhp) 32

[Classe WebSummitTest](#_2xcytpi) 32

[Resultados](#_1ci93xb) 42

[**Verificação do Modelo**](#_3whwml4) **43**

[Exemplo de uma verificação de domínio](#_2bn6wsx) 43

[Exemplo de uma verificação de invariante](#_qsh70q) 44

[**Geração de Código**](#_3as4poj) **45**

[Interface.java](#_1pxezwc) 45

[**Conclusões**](#_49x2ik5) **61**

[**Referências**](#_2p2csry) **62**

# 

# **Descrição Informal do Sistema e Descrição da Lista de Requisitos**

## **Descrição Informal do Sistema**

O presente trabalho tem como objetivo a modelação do evento *WebSummit*.

Este evento tem uma data inicial e final e contém um montra de exibição, um conjunto de participantes, um conjunto de conferências e um conjunto de notícias acerca do evento. A montra de exibição é constituída por *Startups* e investidores. As conferências são constituídas por diferentes empresas e palestras (*talks*), dadas por oradores e assistidas por participantes.

Neste modelo é possível aceder a diferentes informações como o número total de participantes, o calendário do evento, as palestras a começar/decorrer num determinado dia do evento e hora, informação sobre cada investidor ou startup, entre outras.

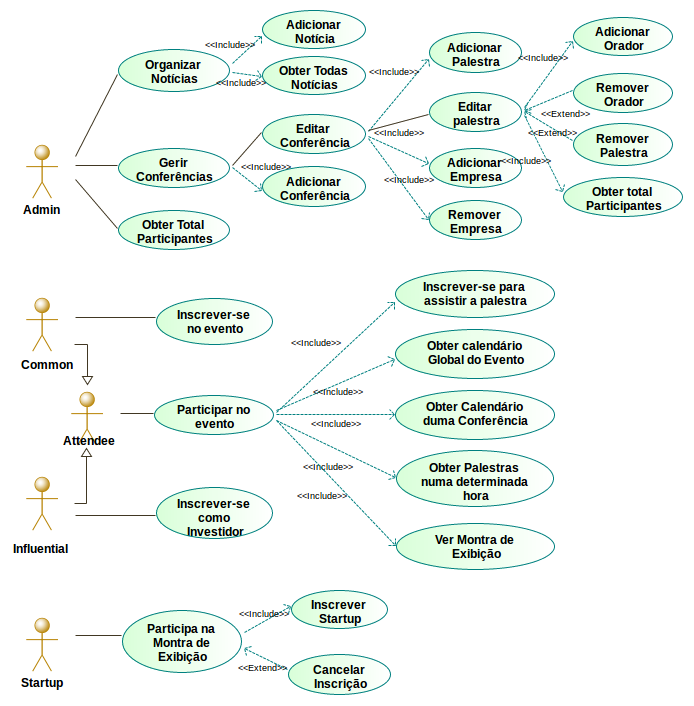
## **Lista de Requisitos**

|  |  |  |
| --- | --- | --- |
| **Requisitos** | **Prioridade** | **Descrição** |
| **R01** | Obrigatória | Adicionar novo participante |
| **R02** | Obrigatória | Adicionar nova conferência |
| **R03** | Obrigatória | Adicionar nova palestra a uma conferência |
| **R04** | Obrigatória | Adicionar uma nova empresa a uma conferência |
| **R05** | Obrigatória | Adicionar oradores a palestras |
| **R06** | Obrigatória | Adicionar participantes a palestras |
| **R07** | Obrigatória | Adicionar Startups à montra de exibição |
| **R08** | Obrigatória | Adicionar investidores à montra de exibição |
| **R09** | Obrigatória | Associar investidores/oradores a empresas |
| **R10** | Obrigatória | Obter o calendário global do evento |
| **R11** | Obrigatória | Obter o calendário de uma conferência que faz parte do evento |
| **R12** | Obrigatória | Obter as palestras a decorrer numa determinada hora |
| **R13** | Obrigatória | Obter o número total de participantes do evento/de cada palestra |
| **R14** | Opcional | Remover/Cancelar uma palestra |
| **R15** | Opcional | Remover/Cancelar a participação de um orador numa palestra |
| **R16** | Opcional | Remover/Cancelar a participação de uma startup na montra de exibição |
| **R17** | Opcional | Remover/Cancelar a participação de uma empresa numa conferência |
| **R18** | Opcional | Adicionar uma nova noticia |
| **R19** | Opcional | Obter todas as notícias feitas acerca do evento |

*Tabela 1 : Lista de Requisitos*

# **Modelo UML**

## **Modelo de Casos de Uso**



*Figura 1 - Modelo de casos de uso do evento WebSummit*

|  |  |
| --- | --- |
| **Cenário** | **Organizar Conferência** |
| **Descrição** | Adicionar conferência ao evento WebSummit |
| **Pré Condições** | 1. A conferência ainda não existe no conjunto de conferências do evento. |
| **Pós Condições** | 1. A conferência passa a pertencer ao conjunto de conferências do evento. |
| **Passos** | 1. No menu inicial, o administrador executa a seguinte sequência de opções “Administration” → “Conferences” → “Add new conference”; 2. O administrador preenche os dados pedidos; 3. Conferência é adicionada e o administrador volta ao menu das conferências. |
| **Exceções** | 1. Já existe uma palestra igual na conferência 2. Já existe uma palestra a decorrer durante o período desta palestra. |

*Tabela 2 : Adicionar conferência ao evento WebSummit*

|  |  |
| --- | --- |
| **Cenário** | **Organizar Palestra** |
| **Descrição** | Adicionar palestra a uma conferência. |
| **Pré Condições** | 1. A palestra ainda não existe no conjunto de palestras dessa conferência 2. Data definida para a palestra está dentro do período de tempo definido para a ocorrência do evento. 3. Não existe outra palestra naquela conferência a decorrer à mesma hora, nem a começar no decorrer da palestra. |
| **Pós Condições** | 1. A palestra passa a pertencer ao conjunto de palestras dessa conferência |
| **Passos** | 1. No menu, o administrador executa a seguinte sequência de ações: “Administration“ → “Conferences” → “Organize conferences”; 2. É apresentada uma lista ao administrador com as conferências possíveis, numeradas; 3. No novo menu correspondente à conferência escolhida o administrador escolhe a opção “Add Talk”; 4. Preencher os campos pedidos para criar a palestra; 5. A palestra é adicionada e o administrador volta ao menu da conferência. |
| **Exceções** | 1. Já existe uma palestra igual na conferência 2. Já existe uma palestra a decorrer durante o período desta palestra. |

*Tabela 3 : Adicionar palestra a uma conferência*

|  |  |
| --- | --- |
| **Cenário** | **Organizar Palestra** |
| **Descrição** | Adicionar orador a uma palestra. |
| **Pré Condições** | 1. A conferência onde a palestra está incluída já existe 2. A palestra especificada existe naquela conferência |
| **Pós Condições** | 1. O orador passa a fazer parte do conjunto de pessoas envolvidas na palestra especificada |
| **Passos** | 1. Seguir os passos 1 e 2 da tabela 3; 2. No novo menu correspondente à conferência escolhida o administrador escolhe a opção “Organize Talks”; 3. É apresentada uma lista ao administrador com as palestras dessa conferência, numeradas; 4. No novo menu correspondente a essa palestra o administrador escolhe a opção “Add New Speaker”; 5. Preencher os dados relativos ao novo orador; 6. O orador é adicionado e o administrador volta ao menu dessa palestra. |
| **Exceções** | 1. A conferência onde a palestra está incluída não existe 2. A palestra especificada não existe |

*Tabela 4 : Adicionar orador a uma palestra*

|  |  |
| --- | --- |
| **Cenário** | **Organizar Conferência** |
| **Descrição** | Remover/Cancelar envolvimento de uma empresa numa dada conferência. |
| **Pré Condições** | 1. Conferência de onde a empresa será retirada existe. 2. Empresa está presente no conjunto de empresas da conferência. 3. Existe pelo menos 1 empresa associada. |
| **Pós Condições** | 1. A empresa deixa de fazer parte do conjunto de empresas |
| **Passos** | 1. Seguir os passos 1 e 2 da tabela 3; 2. No novo menu correspondente à conferência escolhida o administrador escolhe a opção “Remove Company”; 3. É apresentada uma lista ao administrador com as empresas dessa conferência, numeradas; 4. O administrador escolhe a empresa a remover; 5. Empresa é removida da conferência especificada e o administrador volta ao menu das conferências. |
| **Exceções** | 1. A empresa não existe no conjunto de empresas da conferência. |

*Tabela 5 : Remover/Cancelar envolvimento de uma empresa numa dada conferência*

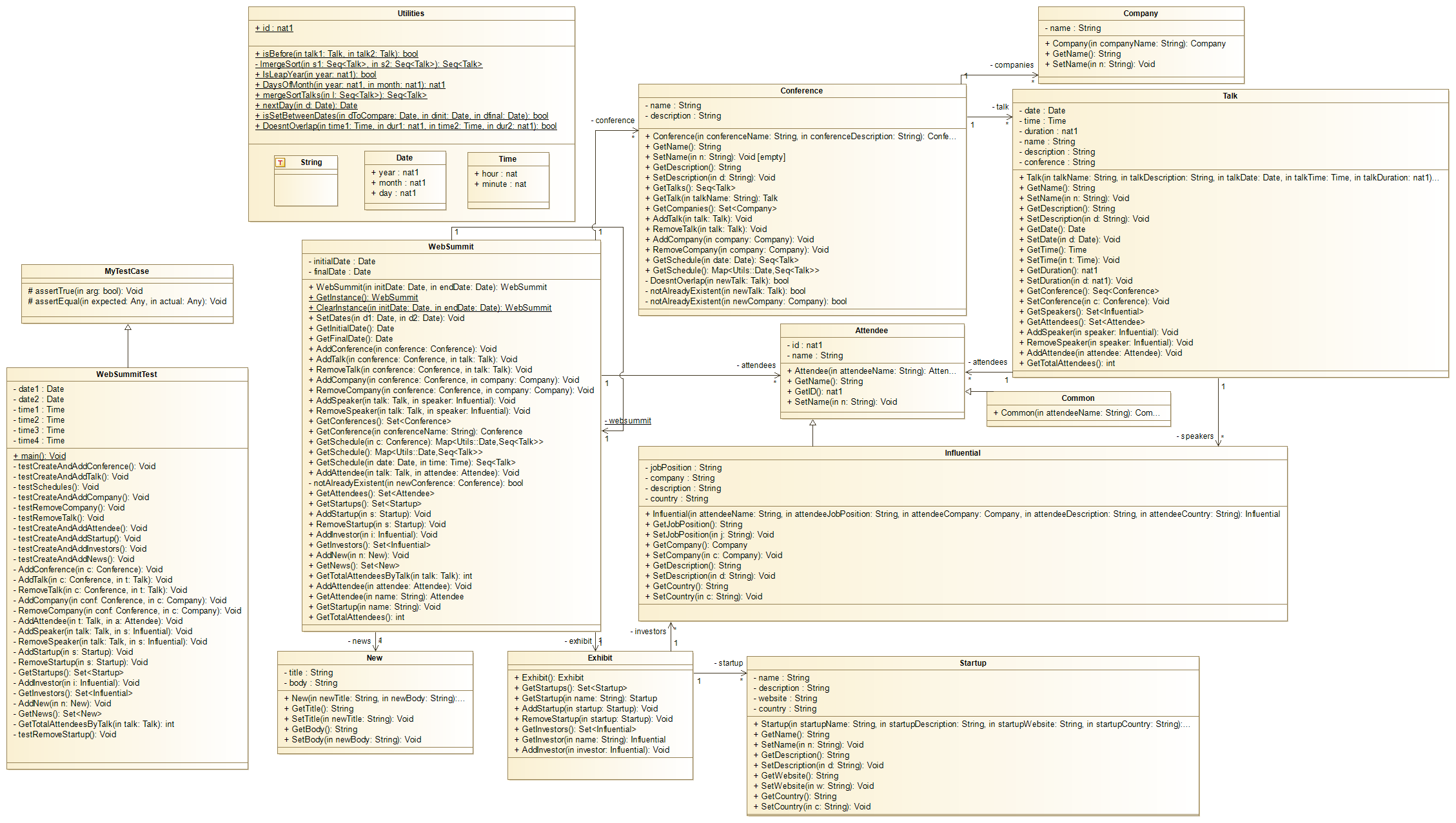
|  |  |
| --- | --- |
| **Cenário** | **Participar no evento** |
| **Descrição** | Obter o calendário global do evento. |
| **Pré Condições** | (nenhuma) |
| **Pós Condições** | (nenhuma) |
| **Passos** | 1. No menu principal, o participante escolhe a opção “Attend Event”; 2. O participante deve introduzir o nome com o qual se registou no evento; 3. No novo menu o utilizador escolhe a opção “Get WebSummit Schedule”; 4. O calendário global do evento é apresentado ao participante, dividido por dias. |
| **Exceções** | 1. Se o utilizador em causa não se registou no evento, não será possível avançar para o passo 3. |

*Tabela 6 : Obter o calendário global do evento*

|  |  |
| --- | --- |
| **Cenário** | **Participar na montra de exibição** |
| **Descrição** | Inscrever startup na montra de exibição. |
| **Pré Condições** | 1. A *startup* ainda não foi inscrita na montra de exibição. 2. Não existe outra *startup* com o mesmo nome. |
| **Pós Condições** | 1. A *startup* passa a pertencer à lista de startups a apresentar na montra de exibição do evento |
| **Passos** | 1. No menu principal, o utilizador executa as seguintes operações “Startups” → “Register”; 2. Prencher os campos pedidos; 3. Utilizador volta ao menu principal. |
| **Exceções** | 1. A startup já estava inscrita. 2. Existe outra startup com o mesmo nome. |

*Tabela 7 : Inscrever startup na montra de exibição*

## **Modelo de Classes**



*Figura 2 - Modelo de classes do evento WebSummit*

|  |  |
| --- | --- |
| **Classe** | **Descrição** |
| **Attendee** | Superclasse de um participante no evento. |
| **Common** | Define um participante que vai assistir a uma palestra no evento. |
| **Influential** | Define um orador que vai estar presente numa palestra do evento ou um investidor que vai participar na montra de exibição. |
| **Company** | Define uma empresa. |
| **Conference** | Define uma conferência. A uma conferência são associadas palestras e empresas. |
| **Exhibit** | Define a montra de exibição. A esta são associadas diversas startups e investidores. |
| **New** | Define uma noticia acerca do evento WebSummit. |
| **Startup** | Define uma startup que vai participar na montra de exibição. |
| **Talk** | Define uma palestra que faz parte do calendário de uma conferência. A uma palestra são associados oradores e participantes. |
| **WebSummit** | Core model. Define o evento WebSummit e todas as operações que podem ser efetuadas no seu contexto. |
| **Utilities** | Define novos tipos de variáveis (como Date ou Time) e funções necessárias e úteis para as outras classes. |
| **MyTestClass** | Superclasse para a classe de teste. Define método assertEquals e assertTrue. |
| **WebSummitTest** | Define todos os cenários de teste possíveis e testa-os. |

*Tabela 8 : Descrição das classes*

# **Modelo Formal VDM++**

## **Classe Attendee**

class Attendee

**types**

**values**

**instance variables**

**private** name : Utilities‘String := []; **private** id : **nat1**;

**operations**

-- constructor of the attendee class

|  |
| --- |
| **public** Attendee : Utilities‘String ==> Attendee |
| Attendee (attendeeName) == ( |
| name := attendeeName; |
| id := Utilities‘id; |
| Utilities‘id := Utilities‘id + 1; |
| **return self** |
| ) |
| **pre len** attendeeName > 0; |
| -- returns the attendee’s name |
| pure **public** GetName : () ==> Utilities‘String |
| GetName () == ( |
| **return** name; |
| ); |
| -- returns the attendee’s id |
| pure **public** GetID : () ==> **nat1** |
| GetID () == ( |
| **return** id; |
| ); |
| -- sets the attendee’s name |
| **public** SetName : Utilities‘String ==> () |
| SetName (n) == ( |
| name:=n; |
| ); |
| **functions** |
| **traces** |
| **end** Attendee |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [Attende](#1v1yuxt)e | 13 | 100.0% | 4 |
| [GetI](#17dp8vu)D | 31 | 100.0% | 20 |
| [GetNam](#17dp8vu)e | 24 | 100.0% | 4 |
| [SetNam](#17dp8vu)e | 37 | 100.0% | 1 |
|  |  |  |  |
| Attendee.vdmpp |  | 100.0% | 29 |

*Tabela 9 : Cobertura da clases Attendee*

## **Classe Common**

**class** Common **is subclass of** Attendee

**types**

**values**

**instance variables**

**operations**

|  |
| --- |
| -- constructor of the common attendee class |
| **public** Common : Utilities‘String ==> Common |
| Common (attendeeName) == ( |
| Attendee(attendeeName); |
| ) |
| **pre len** attendeeName > 0; |
| **functions** |
| **traces** |
| **end** Common |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [Commo](#4f1mdlm)n | 11 | 100.0% | 2 |
|  |  |  |  |
| Common.vdmpp |  | 100.0% | 2 |

*Tabela 10 : Cobertura da classe Common*

## **Classe Company**

**class** Company

**types**

**values**

**instance variables**

**private** name : Utilities‘String := [];

**operations**

-- constructor of the company class

**public** Company : Utilities‘String ==> CompanyCompany (companyName) == (

name := companyName; **return self**

)

**pre len** companyName > 0;

-- returns the company’s name

pure **public** GetName : () ==> Utilities‘String GetName () == (

**return** name;

);

-- sets the company’s name

**public** SetName : Utilities‘String ==> ()SetName (n) == (

name := n; );

**functions**

**traces**

**end** Company

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [Compan](#4f1mdlm)y | 13 | 100.0% | 6 |
| GetName | 21 | 100.0% | 4 |
| SetName | 27 | 100.0% | 1 |
|  |  |  |  |
| Company.vdmpp |  | 100.0% | 11 |

*Tabela 11 : Cobertura da classe Company*

## **Classe Conference**

**class** Conference **types**

**values**

**instance variables**

**private** name : Utilities‘String := [];

**private** description : Utilities‘String := [];

**private** talks : **set of** Talk := {};

**private** companies : **set of** Company := {};

**inv not exists** t1, t2 **in set** talks & t1 <> t2 **and** t1.GetName() = t2.GetName(); **inv not exists** c1, c2 **in set** companies & c1 <> c2 **and** c1.GetName() = c2.GetName();

**operations**

-- constructor of the conference class

**public** Conference : Utilities‘String\*Utilities‘String ==> Conference

Conference (conferenceName, conferenceDescription) == (

name := conferenceName;

description := conferenceDescription;

**return self**

)

**pre len** conferenceName > 0;

-- returns the conference name

pure **public** GetName : () ==> Utilities‘String

GetName () == (**return** name;);

-- set the conference name

**public** SetName : Utilities‘String ==> ()

SetName (n) == (name := n; );

-- returns the conference description

pure **public** GetDescription : () ==> Utilities‘String

GetDescription () == (**return** description;);

-- set the conference description

**public** SetDescription : Utilities‘String ==> ()

SetDescription (d) == (description := d; );

-- returns the conference talks

pure **public** GetTalks : () ==> **set of** Talk

GetTalks () == (**return** talks;);

-- returns a conference talk by it’s name

pure **public** GetTalk : Utilities‘String ==> [Talk]

GetTalk (talkName) == (

**for all** talk **in set** talks **do** ( **if** (talk.GetName() = talkName) **then return** talk;

);

**return** nil

)

**pre len** talkName > 0;

-- returns the conference attending companies

pure **public** GetCompanies : () ==> **set of** Company GetCompanies () == (**return** companies;);

-- adds a new talk to the conference

**public** AddTalk : Talk ==> ()

AddTalk (talk) == (

talk.SetConference(name); talks := talks **union** {talk};

)

**pre** talk **not in set** talks **and** notAlreadyExistent(talk) = **true and** (DoesntOverlap(talk)) = **true**

**post** talks = talks˜ **union** {talk};

-- remove Talk

**public** RemoveTalk : Talk ==> ()RemoveTalk (talk) == (

talks := talks \ {talk};

)

**pre** talk **in set** talks **and card** talks >= 1 **post** talks = talks˜ \ {talk};

-- adds a new company attending the conference

**public** AddCompany : Company ==> ()AddCompany (company) == (

companies := companies **union** {company};

)

**pre** company **not in set** companies **and** notAlreadyExistent(company) = **true**

**post** companies = companies˜ **union** {company};

-- remove company attending

**public** RemoveCompany : Company ==> ()RemoveCompany (company) == (

companies := companies \ {company};

)

**pre** company **in set** companies **and card** companies >= 1

**post** companies = companies˜ \ {company};

-- returns schedule of the day, sorted by time

pure **public** GetSchedule : Utilities‘Date ==> **seq of** Talk

GetSchedule (date) == (

**dcl** talkSet: **seq of** Talk := [];

**for all** talk **in set** talks **do** ( **if**(talk.GetDate() = date)

**then** talkSet := talkSet ˆ [talk];

);

**return** Utilities‘mergeSortTalks(talkSet);

);

-- returns schedule of the conference, sorted

pure **public** GetSchedule : () ==> **map** Utilities‘Date **to seq of** Talk

GetSchedule () == (

**dcl** result: **map** Utilities‘Date **to seq of** Talk := {|->};

**dcl** currentDate : Utilities‘Date := WebSummit‘GetInstance().GetInitialDate(); **dcl** finalDate : Utilities‘Date := WebSummit‘GetInstance().GetFinalDate();

**while** (currentDate <> Utilities‘nextDay(finalDate)) **do** (

result := result **munion** {currentDate |-> GetSchedule(currentDate)}; currentDate := Utilities‘nextDay(currentDate);

);

**return** result

);

-- checks if talk doesnt overlap existing one: for precondition of AddTalk

pure **private** DoesntOverlap : Talk ==> **bool** DoesntOverlap (newTalk) == (

**dcl** doesntOverlap : **bool** := **true**;

**for all** talk **in set** talks **do** ( **if**(talk.GetDate() = newTalk.GetDate())

**then if**(Utilities‘DoesntOverlap(newTalk.GetTime(), newTalk.GetDuration(), talk.GetTime(),talk.GetDuration()) = **false**)

**then**

(

doesntOverlap := false;

**return** doesntOverlap

)

);

**return** doesntOverlap;

)

**pre** newTalk **not in set** talks;

-- checks if a talk with the same name doesn’t exist already: for precondition of AddTask

pure **private** notAlreadyExistent : Talk ==> **bool**

notAlreadyExistent (newTalk) == (

**dcl** doesntExist : **bool** := **true**; **for all** talk **in set** talks **do** (

**if**(talk.GetName() = newTalk.GetName()) **then**(

doesntExist := false;

return doesntExist

|  |  |  |
| --- | --- | --- |
| ) |  |  |
| ); |  |  |
| **return** doesntExist; |  |  |
| ) |  |  |
| **pre** newTalk **not in set** talks; |  |  |
| -- checks if a company with the same name doesn’t exist already: for precondition of AddCompany |  |  |
|  |  |  |
| pure **private** notAlreadyExistent : Company ==> **bool** |  |  |
| notAlreadyExistent (newCompany) == ( |  |  |
| **dcl** doesntExist : **bool** := **true**; |  |  |
| **for all** company **in set** companies **do** ( |  |  |
| **-**- tested on failed tests  **if**(company.GetName() =newCompany.GetName()) |  |  |
| **then**( |  |  |
| doesntExist := false; |  |  |
| **return** doesntExist |  |  |
| ) |  |  |
| ); |  |  |
| **return** doesntExist; |  |  |
| ) |  |  |
| **pre** newCompany **not in set** companies; |  |  |
| **functions** |  |  |
| **traces** |  |  |
| **end** Conference |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [AddCompan](#2u6wntf)y | 93 | 100.0% | 2 |
| [AddTal](#19c6y18)k | 76 | 100.0% | 10 |
| [Conferenc](#3tbugp1)e | 19 | 100.0% | 7 |
| [DoesntOverla](#28h4qwu)p | 138 | 84.8% | 12 |
| [GetCompanie](#2u6wntf)s | 70 | 100.0% | 7 |
| [GetDescriptio](#3tbugp1)n | 40 | 100.0% | 2 |
| [GetNam](#3tbugp1)e | 28 | 100.0% | 45 |
| [GetSchedul](#28h4qwu)e | 109 | 100.0% | 22 |
| [GetTal](#2u6wntf)k | 58 | 88.2% | 1 |
| [GetTalk](#2u6wntf)s | 52 | 100.0% | 11 |
| [RemoveCompan](#28h4qwu)y | 101 | 100.0% | 1 |
| [RemoveTal](#2u6wntf)k | 85 | 100.0% | 1 |
| [SetDescriptio](#2u6wntf)n | 46 | 100.0% | 1 |
| [SetNam](#3tbugp1)e | 34 | 100.0% | 1 |
| [notAlreadyExisten](#28h4qwu)t | 157 | 42.8% | 2 |
|  |  |  |  |
| Conference.vdmpp |  | 90.2% | 125 |

*Tabela 12 : Cobertura da classe Conference*

## **Classe Exhibit**

**class** Exhibit

**types**

**values**

**instance variables**

**private** startups : **set of** Startup := {}; **private** investors : **set of** Influential := {};

**inv not exists** s1, s2 **in set** startups & s1 <> s2 **and** s1.GetName()=s2.GetName(); **inv not exists** i1, i2 **in set** investors & i1 <> i2 **and** i1.GetID() = i2.GetID();

**operations**

-- default constructor of the exibit class

**public** Exhibit : () ==> Exhibit

Exhibit () == (

**return self**

);

-- returns the exibit startups

pure **public** GetStartups : () ==> **set of** Startup

GetStartups () == (**return** startups;);

pure **public** GetStartup : Utilities‘String ==> [Startup]

GetStartup (name) == (

**for all** startup **in set** startups **do** (

**if** (startup.GetName() = name)

**then return** startup

);

**return nil**

)

**pre len** name > 0;

-- add startup

**public** AddStartup: Startup ==> ()

AddStartup (startup) == (

startups := startups **union** {startup}

)

**pre** startup **not in set** startups **and** GetStartup(startup.GetName()) = **nil post** startups = startups˜ **union** {startup};

-- remove startup

**public** RemoveStartup: Startup ==> ()

RemoveStartup (startup) == (

startups := startups \ {startup};

)

**pre** startup **in set** startups **and card** startups >= 1 **post** startups = startups˜ \ {startup};

-- returns the exibit investors

|  |
| --- |
| pure **public** GetInvestors : () ==> **set of** Influential |
| GetInvestors () == ( |
| **return** investors; |
| ); |
| pure **public** GetInvestor : Utilities‘String ==> [Influential] |
| GetInvestor (name) == ( |
| **for all** investor **in set** investors **do** ( |
| if (investor.GetName() = name) |
| **then return** investor |
| ); |
| **return nil** |
| ) |
| **pre len** name > 0; |
| -- add investor |
| **public** AddInvestor: Influential ==> () |
| AddInvestor (investor) == ( |
| investors := investors **union** {investor} |
| ) |
| **pre** investor **not in set** investors **and** GetInvestor(investor.GetName()) = **nil** |
| **post** investors = investors˜ **union** {investor}; |
| **functions** |
| **traces** |
| **end** Exhibit |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [AddInvesto](#nmf14n)r | 73 | 100.0% | 1 |
| [AddStartu](#37m2jsg)p | 40 | 100.0% | 2 |
| [Exhibi](#37m2jsg)t | 17 | 100.0% | 12 |
| [GetInvesto](#nmf14n)r | 61 | 52.9% | 1 |
| [GetInvestor](#37m2jsg)s | 56 | 100.0% | 13 |
| [GetStartu](#37m2jsg)p | 28 | 100.0% | 3 |
| [GetStartup](#37m2jsg)s | 23 | 100.0% | 14 |
| [RemoveStartu](#37m2jsg)p | 48 | 100.0% | 1 |
|  |  |  |  |
| Exhibit.vdmpp |  | 89.3% | 47 |

*Tabela 13 : Cobertura da classe Exhibit*

## **Classe Influential**

**class** Influential **is subclass of** Attendee

**types**

**values**

**instance variables**

**private** jobPosition : Utilities‘String := [];

**private** company : Company;

**private** description : Utilities‘String := [];

**private** country : Utilities‘String := [];

**operations**

-- constructor of the influential attendee class

**public** Influential : Utilities‘String\*Utilities‘String\*Company\*Utilities‘String\*Utilities‘String ==> Influential

Influential (attendeeName, attendeeJobPosition, attendeeCompany, attendeeDescription, attendeeCountry) == (

jobPosition := attendeeJobPosition;

company := attendeeCompany;

description := attendeeDescription;

country := attendeeCountry; Attendee(attendeeName);

)

**pre len** attendeeName > 0 **and len** attendeeJobPosition > 0 **and len** attendeeCountry > 0;

-- returns the influential attendee’s job position

**public** GetJobPosition : () ==> Utilities‘String

GetJobPosition () == (**return** jobPosition;);

-- sets the influential attendee’s job position

**public** SetJobPosition : Utilities‘String ==> ()

SetJobPosition (j) == (jobPosition := j; );

-- returns the influential attendee’s company

**public** GetCompany : () ==> CompanyGetCompany () == (**return** company;);

-- sets the influential attendee’s company

**public** SetCompany : Company ==> ()SetCompany (c) == (company := c; );

-- returns the influential attendee’s description

**public** GetDescription : () ==> Utilities‘String

GetDescription () == (**return** description;);

-- sets the influential attendee’s description

**public** SetDescription : Utilities‘String ==> ()

SetDescription (d) == (description := d; );

-- returns the influential attendee’s country

**public** GetCountry : () ==> Utilities‘String

GetCountry () == (

|  |
| --- |
| **return** country; |
| ); |
| -- sets the influential attendee’s country |
| **public** SetCountry : Utilities‘String ==> () |
| SetCountry (c) == ( |
| country := c; |
| ); |
| **functions** |
| **traces** |
| **end** Influential |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [GetCompan](#1mrcu09)y | 40 | 100.0% | 2 |
| [GetCountr](#1mrcu09)y | 64 | 100.0% | 2 |
| [GetDescriptio](#1mrcu09)n | 52 | 100.0% | 2 |
| [GetJobPositio](#1mrcu09)n | 28 | 100.0% | 2 |
| [Influentia](#1mrcu09)l | 17 | 100.0% | 3 |
| [SetCompan](#1mrcu09)y | 46 | 100.0% | 1 |
| [SetCountr](#46r0co2)y | 70 | 100.0% | 1 |
| [SetDescriptio](#1mrcu09)n | 58 | 100.0% | 1 |
| [SetJobPositio](#1mrcu09)n | 34 | 100.0% | 1 |
|  |  |  |  |
| Influential.vdmpp |  | 100.0% | 15 |

*Tabela 14 : Cobertura da classe Influential*

## **Classe New**

**class** New

**types**

**values**

**instance variables**

**private** title : Utilities‘String := [];

**private** body : Utilities‘String := [];

**operations**

-- constructor of the new class

**public** New : Utilities‘String\*Utilities‘String ==> New

New (newTitle, newBody) == (

title := newTitle;

body := newBody; **return self**

)

**pre len** newTitle > 0 **and len** newBody > 0;

-- returns the news title

**public** GetTitle : () ==> Utilities‘StringGetTitle () == (

**return** title;);

-- set the news title

**public** SetTitle : Utilities‘String ==> ()

SetTitle (newTitle) == (

title := newTitle; );

-- returns the news body

**public** GetBody : () ==> Utilities‘String

GetBody () == ( **return** body;);

|  |  |
| --- | --- |
| -- set | the news body |
| **public** | SetBody : Utilities‘String ==> () |
| SetBody (newBody) == ( | |
| body | := newBody; |
| ); |  |
| **functions** | |
| **traces** |  |
| **end** New |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [GetBod](#2lwamvv)y | 35 | 100.0% | 2 |
| [GetTitl](#2lwamvv)e | 23 | 100.0% | 2 |
| [Ne](#2lwamvv)w | 14 | 100.0% | 1 |
| [SetBod](#111kx3o)y | 41 | 100.0% | 1 |
| [SetTitl](#2lwamvv)e | 29 | 100.0% | 1 |
|  |  |  |  |
| New.vdmpp |  | 100.0% | 7 |

*Tabela 15 : Cobertura da classe New*

## **Classe Startup**

**class** Startup

**types**

**values**

**instance variables**

**private** name : Utilities‘String := [];

**private** description : Utilities‘String := [];

**private** website : Utilities‘String := [];

**private** country : Utilities‘String := [];

**operations**

-- constructor of the startup class

**public** Startup : Utilities‘String\*Utilities‘String\*Utilities‘String\*Utilities‘String==> Startup

Startup (startupName, startupDescription, startupWebsite, startupCountry) == (

name := startupName;

description := startupDescription; website := startupWebsite;

country := startupCountry;

**return self**

);

-- returns the startup name

|  |
| --- |
| pure **public** GetName : () ==> Utilities‘String |
| GetName () == ( |
| **return** name; |
| ); |
| -- set the startup name |
| **public** SetName : Utilities‘String ==> () |
| SetName (n) == ( |
| name := n; |
| ); |
| -- returns the startup description |
| **public** GetDescription : () ==> Utilities‘String |
| GetDescription () == ( |
| **return** description; |
| ); |
| -- set the startup description |
| **public** SetDescription : Utilities‘String ==> () |
| SetDescription (d) == ( |
| description := d; |
| ); |
| -- returns the startup website |
| **public** GetWebsite : () ==> Utilities‘String |
| GetWebsite () == ( |
| **return** website; |
| ); |
| -- set the startup website |
| **public** SetWebsite : Utilities‘String ==> () |
| SetWebsite (w) == ( |
| website := w; |
| ); |
| -- returns the startup country |
| **public** GetCountry : () ==> Utilities‘String |
| GetCountry () == ( |
| **return** country; |
| ); |
| -- set the startup country |
| **public** SetCountry : Utilities‘String ==> () |
| SetCountry (c) == ( |
| country := c; |
| ); |
| **functions** |
| **traces** |
| **end** Startup |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [GetCountr](#3l18frh)y | 62 | 100.0% | 2 |
| [GetDescriptio](#3l18frh)n | 38 | 100.0% | 2 |
| [GetNam](#111kx3o)e | 26 | 100.0% | 5 |
| [GetWebsit](#3l18frh)e | 50 | 100.0% | 2 |
| [SetCountr](#3l18frh)y | 68 | 100.0% | 1 |
| [SetDescriptio](#3l18frh)n | 44 | 100.0% | 1 |
| [SetNam](#3l18frh)e | 32 | 100.0% | 1 |
| [SetWebsit](#3l18frh)e | 56 | 100.0% | 1 |
| [Startu](#111kx3o)p | 16 | 100.0% | 2 |
|  |  |  |  |
| Startup.vdmpp |  | 100.0% | 17 |

*Tabela 16 : Cobertura da classe Startup*

## **Classe Talk**

**class** Talk

**types**

**values**

**instance variables**

**private** name : Utilities‘String := [];

**private** description : Utilities‘String := [];

**private** date : Utilities‘Date;

**private** time : Utilities‘Time;

**private** duration : **nat1**;

**private** conference : Utilities‘String := [];

**private** speakers : **set of** Influential := {};

**private** attendees : **set of** Attendee := {};

**inv not exists** s1, s2 **in set** speakers & s1 <> s2 **and** s1.GetID()=s2.GetID();

**inv not exists** a1, a2 **in set** attendees & a1<>a2 **and** a1.GetID() = a2.GetID();

**operations**

-- constructor of the talk class

**public** Talk : Utilities‘String\*Utilities‘String\*Utilities‘Date\*Utilities‘Time\* **nat1** ==> Talk

Talk (talkName, talkDescription, talkDate, talkTime, talkDuration) == ( name := talkName;

description := talkDescription;

date := talkDate;

time := talkTime;

duration := talkDuration;

**return self**

)

**pre len** talkName > 0 **and** talkDuration > 0;

-- returns the talk’s name

pure **public** GetName : () ==> Utilities‘String

GetName () == (

**return** name;);

-- set the talk name

**public** SetName : Utilities‘String ==> ()

SetName (n) == (

name := n; );

-- returns the talk’s description

**public** GetDescription : () ==> Utilities‘String

GetDescription () == (

**return** description;);

-- set the talk description

**public** SetDescription : Utilities‘String ==> ()

SetDescription (d) == (

description := d; );

-- returns the talk’s date

pure **public** GetDate : () ==> Utilities‘Date

GetDate () == (

**return** date;);

-- set the talk date

**public** SetDate : Utilities‘Date ==> ()

SetDate (d) == (

date := d; );

-- returns the talk’s time

pure **public** GetTime : () ==> Utilities‘Time

GetTime () == (

**return** time;);

-- set the talk time

**public** SetTime : Utilities‘Time ==> ()

SetTime (t) == (

time := t; );

-- returns the talk’s duration

pure **public** GetDuration : () ==> **nat1**

GetDuration () == (

**return** duration;);

-- set the talk duration

**public** SetDuration : **nat1** ==> ()

SetDuration (d) == (

duration := d; );

-- returns the talk’s conference

pure **public** GetConference : () ==> Utilities‘String

|  |
| --- |
| GetConference () == ( |
| **return** conference; |
| ); |
| -- set the conference conference |
| **public** SetConference : Utilities‘String ==> () |
| SetConference (c) == ( |
| conference := c; |
| ); |
| -- returns the talk’s speakers |
| **public** GetSpeakers : () ==> **set of** Influential |
| GetSpeakers () == ( |
| **return** speakers; |
| ); |
| -- returns the talk’s attendees |
| **public** GetAttendees : () ==> **set of** Attendee |
| GetAttendees () == ( |
| **return** attendees; |
| ); |
| -- adds a new speaker to the talk |
| **public** AddSpeaker : Influential ==> () |
| AddSpeaker (speaker) == ( |
| speakers := speakers **union** {speaker}; |
| ) |
| **pre** speaker **not in set** speakers |
| **post** speakers = speakers˜ **union** {speaker}; |
| -- removes a speaker from the talk |
| **public** RemoveSpeaker : Influential ==> () |
| RemoveSpeaker (speaker) == ( |
| speakers := speakers \ {speaker}; |
| ) |
| **pre** speaker **in set** speakers |
| **post** speakers = speakers˜ \ {speaker}; |
| -- adds a new attendee to the talk |
| **public** AddAttendee : Attendee ==> () |
| AddAttendee (attendee) == ( |
| attendees := attendees union {attendee}; |
| ) |
| **pre** attendeenot **in set** attendees |
| **post** attendees =attendees˜union{attendee}; |
| -- returns the total of attendees |
| **public** GetTotalAttendees: () ==> **int** |
| GetTotalAttendees() == ( |
| return card attendees; |
| ); |
| **functions** |
| **traces** |
| **end** Talk |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [AddAttende](about:blank)e | 135 | 100.0% | 1 |
| [AddSpeake](about:blank)r | 119 | 100.0% | 1 |
| [GetAttendee](about:blank)s | 113 | 100.0% | 1 |
| [GetConferenc](about:blank)e | 95 | 100.0% | 4 |
| [GetDat](about:blank)e | 59 | 100.0% | 324 |
| [GetDescriptio](about:blank)n | 47 | 100.0% | 2 |
| [GetDuratio](about:blank)n | 83 | 100.0% | 22 |
| [GetNam](about:blank)e | 35 | 100.0% | 108 |
| [GetSpeaker](about:blank)s | 107 | 100.0% | 3 |
| [GetTim](about:blank)e | 71 | 100.0% | 430 |
| [GetTotalAttendee](about:blank)s | 143 | 0.0% | 0 |
| [RemoveSpeake](about:blank)r | 127 | 100.0% | 2 |
| [SetConferenc](about:blank)e | 101 | 100.0% | 10 |
| [SetDat](about:blank)e | 65 | 100.0% | 1 |
| [SetDescriptio](about:blank)n | 53 | 100.0% | 1 |
| [SetDuratio](about:blank)n | 89 | 100.0% | 1 |
| [SetNam](about:blank)e | 41 | 100.0% | 1 |
| [SetTim](about:blank)e | 77 | 100.0% | 1 |
| [Tal](about:blank)k | 23 | 100.0% | 10 |
|  |  |  |  |
| Talk.vdmpp |  | 95.4% | 922 |

*Tabela 17 : Cobertura da classe Talk*

## **Classe WebSummit**

**class** WebSummit

**types**

**values**

**instance variables**

**private** conferences : **set of** Conference := {};

**private** exhibit : Exhibit := **new** Exhibit();

**private** attendees : **set of** Attendee := {};

**private** news : **set of** New := {};

-- default dates

**private** initialDate : Utilities‘Date := mk\_Utilities‘Date(2001,1,1);

**private** finalDate : Utilities‘Date := mk\_Utilities‘Date(2001,1,2);

**private static** websummit: WebSummit := **new** WebSummit();

**inv not exists** c1, c2 **in set** conferences & c1 <> c2 **and** c1.GetName() = c2.GetName();

**inv not exists** a1, a2 **in set** attendees & a1 <> a2 **and** a1.GetID() = a2.GetID();

**operations**

-- constructor of the websummit class

**public** WebSummit : Utilities‘Date\*Utilities‘Date ==> WebSummit

WebSummit (initDate, endDate) == (

initialDate := initDate;

finalDate := endDate;

**return self**

);

-- singleton - return the existent instance

**public** pure **static** GetInstance: () ==> WebSummit

GetInstance() == (

**return** websummit;);

-- singleton - reset the instance

**public static** ClearInstance: Utilities‘Date\*Utilities‘Date ==> WebSummit

ClearInstance(initDate, endDate) == (

websummit := **new** WebSummit(initDate, endDate);

**return** GetInstance();

)

**post RESULT**.conferences = {} **and RESULT**.exhibit.GetStartups() = {} **and RESULT**.exhibit.GetInvestors() = {} **and RESULT**.attendees = {};

-- sets websummit dates

**public** SetDates : Utilities‘Date\*Utilities‘Date ==> ()SetDates (d1, d2) == (

initialDate := d1; finalDate := d2;

)

**post** initialDate = d1 **and** finalDate = d2;

* returns websummit initial date

pure **public** GetInitialDate : () ==> Utilities‘Date

GetInitialDate () == (

**return** initialDate);

-- returns websummit final date

pure **public** GetFinalDate : () ==> Utilities‘Date

GetFinalDate () == ( **return** finalDate);

-- creates a new conference

**public** AddConference : Conference ==> ()

AddConference (conference) == (

conferences := conferences **union** {conference};

)

**pre** conference **not in set** conferences **and** notAlreadyExistent(conference) = **true**

**post** conferences = conferences˜ **union** {conference};

-- adds a new talk to an existing conference

**public** AddTalk : Conference\*Talk ==> ()

AddTalk (conference, talk) == (

conference.AddTalk(talk);

)

**pre** conference **in set** conferences **and** Utilities‘isSetBetweenDates(talk.GetDate(), initialDate, finalDate);

-- removes a talk from an existing conference

**public** RemoveTalk : Conference\*Talk ==> ()

RemoveTalk (conference, talk) == (

conference.RemoveTalk(talk);

)

**pre** conference **in set** conferences;

-- adds a new company to an existing conference

**public** AddCompany : Conference\*Company ==> ()

AddCompany (conference, company) == ( conference.AddCompany(company);

)

**pre** conference **in set** conferences;

-- removes a company from an existing conference

**public** RemoveCompany : Conference\*Company ==> ()

RemoveCompany (conference, company) == ( conference.RemoveCompany(company);

)

**pre** conference **in set** conferences;

-- adds a speaker no an existent talk

**public** AddSpeaker: Talk\*Influential ==> ()

AddSpeaker (talk, speaker) == (

talk.AddSpeaker(speaker);

**if**(speaker **not in set** attendees)

**then** attendees := attendees **union** {speaker}

)

**pre** GetConference(talk.GetConference()) **in set** conferences **post** speaker **in set** attendees;

-- removes a speaker from an existent conference

**public** RemoveSpeaker: Talk\*Influential ==> ()

RemoveSpeaker (talk, speaker) == (

talk.RemoveSpeaker(speaker);

)

-- all talks from all conferences

**pre** GetConference(talk.GetConference()) **in set** conferences

**post** speaker **in set** attendees;

-- returns all confereces

pure **public** GetConferences : () ==> **set of** Conference

GetConferences () == (

**return** conferences);

-- returns a specific conferece by it’s name

**public** pure GetConference : Utilities‘String ==> [Conference]

GetConference (conferenceName) == (

**for all** conference **in set** conferences **do** (

**if** conference.GetName() = conferenceName

**then return** conference

);

return nil

)

**pre len** conferenceName > 0;

-- returns the full schedule of a conference

**public** GetSchedule : Conference ==> **map** Utilities‘Date **to seq of** Talk

GetSchedule (c) == (

**return** c.GetSchedule();

)

**pre** c **in set** conferences;

-- returns the full event schedule

**public** GetSchedule : () ==> **map** Utilities‘Date **to seq of** Talk

GetSchedule () == (

**dcl** temp: **map** Utilities‘Date **to seq of** Talk := {|->}; **dcl** currentDate : Utilities‘Date := initialDate;

-- joins all the events

**for all** conference **in set** conferences **do** ( **if**(temp = {|->})

**then** temp := conference.GetSchedule() **else**

(

**while** (currentDate <> Utilities‘nextDay(finalDate))

**do** (

temp(currentDate) := temp(currentDate) ˆ conference.GetSchedule(currentDate);

currentDate := Utilities‘nextDay(currentDate);

);

);

);

currentDate := WebSummit‘GetInstance().GetInitialDate();

-- orders talks by time

**while** (currentDate <> Utilities‘nextDay(finalDate))

**do** (

temp(currentDate) := Utilities‘mergeSortTalks(temp(currentDate));

currentDate := Utilities‘nextDay(currentDate);

);

**return** temp;

);

-- returns the event schedule by date/time

**public** GetSchedule : Utilities‘Date\*Utilities‘Time ==> **seq of** Talk

GetSchedule (date, time) == (

**dcl** temp: **seq of** Talk := [];

-- joins all talks from that day starting or occuring at the given time

**for all** conference **in set** conferences **do** (

**for all** talk **in set elems** conference.GetSchedule()(date) **do**(

**if**(talk.GetTime().hour = time.hour)

**then** temp := temp ˆ [talk]

**else**

**if**(talk.GetTime().hour + 1 = time.hour)

**then if**((talk.GetDuration()) >= (60 - talk.GetTime().minute))

**then if**((talk.GetTime().minute + talk.GetDuration() - 60) <= 60)

**then** temp := temp ˆ [talk]

);

);

-- orders them by time

temp := Utilities‘mergeSortTalks(temp);

**return** temp;

)

**pre forall** conference **in set** conferences & date **in set** (**dom** (conference.GetSchedule)()) ;

-- adds a new attendee to event

**public** AddAttendee : Attendee ==> ()

AddAttendee (attendee) == (

attendees := attendees **union** {attendee}

)

**pre** attendee **not in set** attendees **post** attendee **in set** attendees;

--adds a new attendee to an existing talk

**public** AddAttendee : Talk\*Attendee ==> ()

AddAttendee (talk, attendee) == (

talk.AddAttendee(attendee);

**if**(attendee **not in set** attendees)

**then** attendees := attendeesunionfattendee}

)

**pre** GetConference(talk.GetConference()) **in set** conferences **post** attendee **in set** attendees;

* checks if a conference with the same name doesn’t exist already: for precondition of AddConference

pure **private** notAlreadyExistent : Conference ==> **bool** notAlreadyExistent (newConference) == (

**dcl** doesntExist : **bool** := **true**;

**for all** conference **in set** conferences **do** (

**if**(conference.GetName() = newConference.GetName())

**then**(

doesntExist := false;

return doesntExist

)

);

**return** doesntExist;

)

**pre** newConference **not in set** conferences;

-- returns all websummit attendees

**public** GetAttendees : () ==> **set of** Attendee

GetAttendees () == ( **return** attendees;);

-- returns all websummit attendees

**public** GetAttendee : Utilities‘String ==> [Attendee]

GetAttendee (name) == (

**for all** a **in set** attendees **do** (

**if** (a.GetName() = name)

**then return** a

);

return nil

)

**pre len** name > 0;

--returns all startups

**public** GetStartups: () ==> **set of** Startup

GetStartups () == ( exhibit.GetStartups(););

--get startup by name

**public** GetStartup: Utilities‘String ==> Startup

GetStartup (name) == (

**return** exhibit.GetStartup(name);

);

--add new startup to exhibit

**public** AddStartup: Startup ==> ()

AddStartup (s) == ( exhibit.AddStartup(s););

--remove startup from exhibit

**public** RemoveStartup: Startup ==> ()

RemoveStartup (s) == ( exhibit.RemoveStartup(s););

--add investor to exhibit

**public** AddInvestor: Influential ==> ()

AddInvestor(i) == ( exhibit.AddInvestor(i););

--return all investors

**public** GetInvestors: () ==> **set of** Influential

GetInvestors() == (

exhibit.GetInvestors();

);

-- add a new

**public** AddNew: New ==> ()

AddNew(n) == (

news := news **union** {n};

)

**pre** n **not in set** news

|  |  |
| --- | --- |
| **post** news = news˜ **union** {n}; | |
| --return all news | |
| **public** GetNews: () ==> **set of** New | |
| GetNews() == ( | |
| **return** news; | |
| ); |  |
| -- get total attendees by talk | |
| **public** GetTotalAttendeesByTalk : Talk ==> **int** | |
| GetTotalAttendeesByTalk (talk) == ( | |
| return talk.GetTotalAttendees(); | |
| ); |  |
| -- get total attendees at the event | |
| **public** GetTotalAttendees : () ==> **int** | |
| GetTotalAttendees () == ( | |
| **return card** attendees; | |
| ); |  |
| **functions** | |
| **traces** | |
| **end** WebSummit | |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  | Function or operation | Line | Coverage | Calls | |  |
|  |  |  |  |  |  | |  |  |  |
|  |  | [AddAttende](about:blank)e | 209 | 100.0% | 1 | |  |  |  |
|  |  | [AddCompan](about:blank)y | 91 | 100.0% | 2 | |  |  |  |
|  |  | [AddConferenc](about:blank)e | 68 | 100.0% | 7 | |  |  |  |
|  |  | [AddInvesto](about:blank)r | 284 | 100.0% | 1 | |  |  |  |
|  |  | [AddNe](about:blank)w | 296 | 100.0% | 1 | |  |  |  |
|  |  | [AddSpeake](about:blank)r | 106 | 100.0% | 2 | |  |  |  |
|  |  | [AddStartu](about:blank)p | 272 | 100.0% | 2 | |  |  |  |
|  |  | [AddTal](about:blank)k | 76 | 100.0% | 10 | |  |  |  |
|  |  | [ClearInstanc](about:blank)e | 39 | 100.0% | 11 | |  |  |  |
|  |  | [GetAttende](about:blank)e | 249 | 88.2% | 1 | |  |  |  |
|  |  | [GetAttendee](about:blank)s | 243 | 100.0% | 1 | |  |  |  |
|  |  | [GetConferenc](about:blank)e | 131 | 88.2% | 0 | |  |  |  |
|  |  | [GetConference](about:blank)s | 125 | 100.0% | 4 | |  |  |  |
|  |  | [GetFinalDat](about:blank)e | 62 | 100.0% | 23 | |  |  |  |
|  |  | [GetInitialDat](about:blank)e | 56 | 100.0% | 25 | |  |  |  |
|  |  | [GetInstanc](about:blank)e | 33 | 100.0% | 96 | |  |  |  |
|  |  | [GetInvestor](about:blank)s | 290 | 100.0% | 2 | |  |  |  |
|  |  | [GetNew](about:blank)s | 304 | 100.0% | 2 | |  |  |  |
|  |  | [GetSchedul](about:blank)e | 143 | 100.0% | 6 | |  |  |  |
|  |  | [GetStartu](about:blank)p | 266 | 100.0% | 1 | |  |  |  |
|  |  | [GetStartup](about:blank)s | 260 | 100.0% | 3 | |  |  |  |
|  |  | [GetTotalAttendee](about:blank)s | 316 | 100.0% | 2 | |  |  |  |
|  |  | [GetTotalAttendeesByTal](about:blank)k | 310 | 100.0% | 1 | |  |  |  |
|  |  | [RemoveCompan](about:blank)y | 98 | 100.0% | 1 | |  |  |  |
|  | | [RemoveSpeake](about:blank)r | 116 | 100.0% | 1 |
|  | | [RemoveStartu](about:blank)p | 278 | 100.0% | 1 |
|  | | [RemoveTal](about:blank)k | 84 | 100.0% | 1 |
|  | | [SetDate](about:blank)s | 48 | 0.0% | 0 |
|  | | [WebSummi](about:blank)t | 25 | 100.0% | 11 |
|  | | [notAlreadyExisten](about:blank)t | 227 | 76.1% | 0 |
|  | |  |  |  |  |
|  | | WebSummit.vdmpp |  | 94.3% | 219 |

*Tabela 18 : Cobertura da classe WebSummit*

## **Classe Utilities**

**class** Utilities

**types**

**public** String = **seq of char**;

**public** Date :: year : **nat1** month: **nat1** day : **nat1**

**inv** d == d.month >= 1 **and** d.month <= 12 **and** d.day >= 1 **and** d.day <= DaysOfMonth(d.year,d.month);

**public** Time :: hour : **nat** minute: **nat**

**inv** t == t.hour < 24 **and** t.minute < 60;

**values**

**instance variables**

**public static** id : **nat1** := 1;

**operations**

pure **public static** isBefore: Talk \* Talk ==> **bool** isBefore(talk1, talk2) == (

**if** (talk1.GetTime().hour < talk2.GetTime().hour) **then return true**

**elseif**(talk1.GetTime().hour = talk2.GetTime().hour)

**then if**(talk1.GetTime().minute < talk2.GetTime().minute) **then return true**

**else return false else return false**

);

pure **private static** lmergeSort : **seq of** Talk \* **seq of** Talk ==> **seq of** Talk

lmergeSort (s1,s2) == (

**if** s1 = [] **then return** s2

**elseif** (s2 = []) **then return** s1 **elseif**

isBefore(**hd** s1, **hd** s2)

**then return** [**hd** s1] ˆ (lmergeSort (**tl** s1, s2)) **else return** [**hd** s2] ˆ (lmergeSort (s1, **tl** s2))

);

**functions**

**public** IsLeapYear: **nat1** +> **bool**

IsLeapYear(year) == year **mod** 4 = 0 **and** year **mod** 100 <> 0 **or** year **mod** 400=0;

**public** DaysOfMonth: **nat1** \* **nat1** -> **nat1**

DaysOfMonth(year, month) == (

**cases** month : 1, 3, 5, 7,8, 10, 12 -> 31, 4, 6, 9, 11 -> 30, 2 -> if IsLeapYear(year) **then** 29 **else** 28 **end**

)

**pre** month >= 1 **and** month <= 12;

**public** mergeSortTalks : **seq of** Talk -> **seq of** TalkmergeSortTalks (l) == (

**if** l = [] **or len** l = 1 **then** l

**else** lmergeSort (mergeSortTalks([**hd** l]), mergeSortTalks(**tl** l))

);

**public** nextDay: Date -> DatenextDay(d) == (

**if** (d.day = DaysOfMonth(d.year,d.month)) **then** if(d.month< 12)

**then** mk\_Date(d.year, d.month + 1,1) **else**

mk\_Date(d.year + 1, 1, 1) **else**

mk\_Date(d.year, d.month, d.day + 1)

);

**public** isSetBetweenDates: Date\*Date\*Date -> **bool** isSetBetweenDates(dToCompare, dinit, dfinal) == (

**if**(dinit = nextDay(dfinal))

**then** false

**else if** (dToCompare = dinit)

**then true**

**else** isSetBetweenDates(dToCompare, nextDay(dinit), dfinal)

);

**public** DoesntOverlap: Time\* **nat1** \*Time\* **nat1** -> **bool** DoesntOverlap(time1, dur1, time2, dur2) == (

**if** (time1.hour < time2.hour)

**then if** (time1.minute + dur1 <= 60) **then true**

**else** if(time1.minute + dur1 - 60 <= time2.minute)

**then** true

**else** false

**elseif** (time1.hour = time2.hour)

**then if**(time1.minute < time2.minute)

**then if**(time1.minute + dur1 <= time2.minute) **then true**

**else** false

**elseif**(time2.minute<time1.minute)

**then if**(time2.minute+dur2 <= time1.minute) **then** true

**else** false **else** false

**elseif** (time2.hour < time1.hour) **then if** (time2.minute + dur2 <= 60)

**then true**

**else if**(time2.minute + dur2 - 60 <= time1.minute)

|  |  |  |
| --- | --- | --- |
| **then true** |  |  |
| **else** false |  |  |
| **else** false |  |  |
| ); |  |  |
| **traces** |  |  |
| **end** Utilities |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [DaysOfMont](about:blank)h | 51 | 68.7% | 456 |
| [DoesntOverla](about:blank)p | 90 | 65.2% | 2 |
| [IsLeapYea](about:blank)r | 50 | 33.3% | 456 |
| [isBefor](about:blank)e | 24 | 100.0% | 392 |
| [isSetBetweenDate](about:blank)s | 80 | 93.7% | 20 |
| [lmergeSor](about:blank)t | 36 | 100.0% | 48 |
| [mergeSortTalk](about:blank)s | 53 | 100.0% | 214 |
| [nextDa](about:blank)y | 67 | 47.3% | 215 |
|  |  |  |  |
| Utilities.vdmpp |  | 75.0% | 1803 |

*Tabela 19 : Cobertura da classe Utilities*

# **Validação do Modelo**

## **Classe MyTestClass**

**class** MyTestCase/\*

Superclass **for** test classes, simpler but more practical than VDMUnit‘TestCase. For proper use, you have **to do**: New -> Add VDM Library -> IO.

JPF, FEUP, MFES, 2014/15.

\*/

**operations**

* Simulates assertion checking by reducing it to pre-condition checking.
* If ’arg’ does not hold, a pre-condition violation will be signaled.

**protected** assertTrue: **bool** ==>()

assertTrue(arg) == return **pre** arg;

* Simulates assertion checking by reducing it to post-condition checking.
* If values are not equal, prints a message in the console and generates
* a post-conditions violation.

**protected** assertEqual: ?\*? ==> ()assertEqual(expected, actual) == **if** expected <> actual **then** (

IO‘print("Actual value (");

IO‘print(actual);

IO‘print(") different from expected (");

IO‘print(expected);

IO‘println(")nn")

|  |  |  |
| --- | --- | --- |
| ) |  |  |
| **post** expected = actual |  |  |
| **end** MyTestCase |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| Function or operation | Line | Coverage | Calls |
|  |  |  |  |
| [assertEqua](#46r0co2)l | 20 | 38.8% | 0 |
| [assertTru](#46r0co2)e | 12 | 0.0% | 0 |
|  |  |  |  |
| MyTestClass.vdmpp |  | 35.0% | 0 |

*Tabela 20 : Cobertura da classe MyTestClass*

## **Classe WebSummitTest**

**class** WebSummitTest **is subclass of** MyTestCase

**types**

**values**

**instance variables**

date1 : Utilities‘Date := mk\_Utilities‘Date(2017,9,1);

date2 : Utilities‘Date := mk\_Utilities‘Date(2017,9,3);

time1 : Utilities‘Time := mk\_Utilities‘Time(15,20);

time2 : Utilities‘Time := mk\_Utilities‘Time(15,40);

time3 : Utilities‘Time := mk\_Utilities‘Time(16,40);

time4 : Utilities‘Time := mk\_Utilities‘Time(15,30);

**operations**

**public static** main: () ==> ()

main() ==(

**dcl** webSummitTest: WebSummitTest := **new** WebSummitTest();

IO‘print("testCreateAndAddConference -> "); webSummitTest.testCreateAndAddConference(); IO‘println("Success");

IO‘print("testCreateAndAddTalk -> "); webSummitTest.testCreateAndAddTalk(); IO‘println("Success");

IO‘print("testRemoveTalk -> "); webSummitTest.testRemoveTalk(); IO‘println("Success");

IO‘print("testSchedules -> "); webSummitTest.testSchedules(); IO‘println("Success");

IO‘print("testCreateAndAddCompany -> "); webSummitTest.testCreateAndAddCompany(); IO‘println("Success");

IO‘print("testRemoveCompany -> "); webSummitTest.testRemoveCompany(); IO‘println("Success");

IO‘print("testCreateAndAddAttendee -> "); webSummitTest.testCreateAndAddAttendee(); IO‘println("Success");

IO‘print("testCreateAndAddStartup -> "); webSummitTest.testCreateAndAddStartup(); IO‘println("Success");

IO‘print("testRemoveStartup -> "); webSummitTest.testRemoveStartup(); IO‘println("Success");

IO‘print("testCreateAndAddInvestors -> "); webSummitTest.testCreateAndAddInvestors(); IO‘println("Success");

IO‘print("testCreateAndAddNews -> "); webSummitTest.testCreateAndAddNews(); IO‘println("Success");

);

-- test if the creation of conferences is working correctly

**private** testCreateAndAddConference: () ==> ()testCreateAndAddConference() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2); **dcl** conference1 : Conference := **new** Conference("C1", "D1");

-- for tests supposed to fail

/\***dcl** conference2 : Conference; \*/

assertEqual(date1, webSummit.GetInitialDate());

assertEqual(date2, webSummit.GetFinalDate());

AddConference(conference1);

-- tests gets and sets

assertEqual("C1", conference1.GetName());

assertEqual("D1", conference1.GetDescription());

assertEqual({}, conference1.GetTalks());

assertEqual({}, conference1.GetCompanies());

assertEqual(conference1, webSummit.GetConference(conference1.GetName()));

conference1.SetName("Conference 1");

assertEqual("Conference 1", conference1.GetName());

conference1.SetDescription("Conference 1 details");

assertEqual("Conference 1 details", conference1.GetDescription());

assertEqual(1, **card** webSummit.GetConferences());

assertEqual({conference1}, webSummit.GetConferences());

* this test is supposed to fail (there can’t be two conferences with the same name)

/\*AddConference(conference2);\*/

* this test is supposed to fail (can’t create conferences with empty name)

/\*conference1 := **new** Conference("", "Conference 3 details");\*/

);

-- test if the creation of talks is working correctly

**private** testCreateAndAddTalk: () ==> ()testCreateAndAddTalk() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** conference : Conference := **new** Conference("Conference 1", "Conference 1 details");

**dcl** talk1 : Talk := **new** Talk("T1", "D1", date2, time2, 30);

**dcl** company : Company := **new** Company("Facebook");

**dcl** speaker1 : Influential := **new** Influential("Mark Zuckerberg","CEO",company,"SpeakerDescription","EN");

**dcl** attendee1 : Common := **new** Common("Ines");

-- for tests supposed to fail

/\***dcl** talk2 : Talk := **new** Talk("Talk 2", "Talk 2 description", date1, time2, 40);

**dcl** talk3 : Talk := **new** Talk("Talk 3", "Talk 3 description", mk\_Utilities‘Date(2017,8,30),time2, 40);

**dcl** talk4 : Talk := **new** Talk("Talk 1", "Talk 4 description", date1, time2, 40);\*/

AddConference(conference);

AddTalk(conference,talk1);

-- tests gets and sets

assertEqual("T1",talk1.GetName());

assertEqual("D1", talk1.GetDescription());

assertEqual(date2, talk1.GetDate());

assertEqual(time2, talk1.GetTime());

assertEqual(30, talk1.GetDuration());

assertEqual("Conference 1", talk1.GetConference());

assertEqual({}, talk1.GetSpeakers());

assertEqual({}, talk1.GetAttendees());

talk1.SetName("Talk 1");

assertEqual("Talk 1", talk1.GetName());

talk1.SetDescription("Talk 1 description");

assertEqual("Talk 1 description", talk1.GetDescription());

talk1.SetDate(date1);

assertEqual(date1, talk1.GetDate());

talk1.SetTime(time1);

assertEqual(time1, talk1.GetTime());

talk1.SetDuration(40);

assertEqual(40, talk1.GetDuration());

assertEqual(talk1, conference.GetTalk(talk1.GetName()));

assertEqual(conference.GetName(), talk1.GetConference());

assertEqual(1, **card** conference.GetTalks());

assertEqual({talk1}, conference.GetTalks());

* add speaker to talk

AddSpeaker(talk1,speaker1);

assertEqual({speaker1}, talk1.GetSpeakers());

--os restantes gets e sets ja foram testados em testCreateAndAddAttendee

-- add attendee to talk

AddAttendee(talk1,attendee1);

assertEqual({attendee1}, talk1.GetAttendees());

assertEqual(1, webSummit.GetTotalAttendeesByTalk(talk1));

* remove speaker from talk

RemoveSpeaker(talk1,speaker1);

assertEqual({}, talk1.GetSpeakers());

* this test is supposed to fail (can’t add a talk that overlaps an already existing one)

/\*AddTalk(conference,talk2);\*/

* this test is supposed to fail (can’t add a talk with a date before/after the dates when websummit happens)

/\*AddTalk(conference,talk3);\*/

* this test is supposed to fail (there can’t be two talks with the same name)

/\*AddTalk(conference,talk4);\*/

* this test is supposed to fail (can’t create talks with empty name)

/\*talk2 := **new** Talk(conference, "", "Talk 2 description", date1, time2, 40);\*/

);

-- test if the construction of the schedules is working correctly

**private** testSchedules: () ==> ()testSchedules() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** conference1 : Conference := **new** Conference("Conference 1", "Conference 1 details"); **dcl** conference2 : Conference := **new** Conference("Conference 2", "Conference 2 details");

**dcl** talk1 : Talk := **new** Talk("Talk 1", "Talk 1 description", date1, time2, 40);

**dcl** talk2 : Talk := **new** Talk("Talk 2", "Talk 2 description", date1, time1, 20);

**dcl** talk3 : Talk := **new** Talk("Talk 3", "Talk 3 description", date1, time3, 20);

**dcl** talk4 : Talk := **new** Talk("Talk 4", "Talk 4 description", date2, time3, 20);

**dcl** talk5 : Talk := **new** Talk("Talk 5", "Talk 5 description", date1, time4, 20);

**dcl** talk6 : Talk := **new** Talk("Talk 6", "Talk 6 description", date2, time2, 60);

**dcl** talk7 : Talk := **new** Talk("Talk 7", "Talk 7 description", date2, time3, 20);

**dcl** talk8 : Talk := **new** Talk("Talk 8", "Talk 8 description", date2, time1, 20);

AddConference(conference1);

AddConference(conference2);

AddTalk(conference1, talk1);

AddTalk(conference1, talk2);

AddTalk(conference1, talk3);

AddTalk(conference1, talk4);

AddTalk(conference2, talk5);

AddTalk(conference2, talk6);

AddTalk(conference2, talk7);

AddTalk(conference2, talk8);

assertEqual(2, **card** webSummit.GetConferences());

assertEqual({conference1, conference2}, webSummit.GetConferences());

|  |  |  |
| --- | --- | --- |
| assertEqual(4, | **card** | conference1.GetTalks()); |
| assertEqual({talk1, | | talk2, talk3, talk4}, conference1.GetTalks()); |
| assertEqual(4, | **card** | conference2.GetTalks()); |
| assertEqual({talk5, | | talk6, talk7, talk8}, conference2.GetTalks()); |

assertEqual(2, **len** webSummit.GetSchedule(date1, mk\_Utilities‘Time(16,00)));

assertEqual([talk1, talk3], webSummit.GetSchedule(date1, mk\_Utilities‘Time(16,00)));

assertEqual(3, **len** webSummit.GetSchedule(date1, mk\_Utilities‘Time(15,00)));

assertEqual([talk2, talk5, talk1], webSummit.GetSchedule(date1, mk\_Utilities‘Time(15,00)));

assertEqual(3, **card dom** webSummit.GetSchedule(conference1));

assertEqual({date1|->[talk2, talk1, talk3], Utilities‘nextDay(date1)|->[], date2|->[talk4]}, webSummit.GetSchedule(conference1));

assertEqual(3, **card dom** webSummit.GetSchedule(conference2));

assertEqual({date1|->[talk5], Utilities‘nextDay(date1)|->[], date2|->[talk8, talk6, talk7]}, webSummit.GetSchedule(conference2));

assertEqual(3, **card dom** webSummit.GetSchedule());

assertEqual({date1|->[talk2, talk5, talk1, talk3], Utilities‘nextDay(date1)|->[], date2|->[ talk8, talk6, talk7, talk4]}, webSummit.GetSchedule());

-- this test is supposed to fail (2 first talks badly sorted, talk 2 takes place before talk5)

/\*assertEqual({date1|->[talk5, talk2, talk1, talk3], Utilities‘nextDay(date1)|->[], date2|->[ talk8, talk6, talk7, talk4]}, webSummit.GetSchedule());

\*/

-- this test is supposed to fail (talk that starts at 15:20 and finishes at 15:40)

/\*assertEqual(3, **len** webSummit.GetSchedule(date1, mk\_Utilities‘Time(16,00))); assertEqual([talk2, talk1, talk3], webSummit.GetSchedule(date1, mk\_Utilities‘Time(16,00)));\*/

);

-- test if the creation of companies is working correctly

**private** testCreateAndAddCompany: () ==> ()testCreateAndAddCompany() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** conference1 : Conference := **new** Conference("Conference 1", "Conference 1 details");

**dcl** company1 : Company := **new** Company("Comp1");

-- for tests supposed to fail

/\***dcl** company2 : Company := **new** Company("Company 1");\*/

AddConference(conference1);

AddCompany(conference1,company1);

-- tests gets and sets

assertEqual("Comp1", company1.GetName()); company1.SetName("Company 1");

assertEqual("Company 1", company1.GetName());

assertEqual(1, **card** conference1.GetCompanies()); assertEqual({company1}, conference1.GetCompanies());

* this test is supposed to fail (can’t create companies with empty name)

/\*AddCompany(conference1,company2);\*/

* this test is supposed to fail (there can’t be two companies with the same name)

/\*AddCompany(conference1, **new** Company("Company 1"));\*/

);

-- test if the removal of companies from conferences is working correctly

**private** testRemoveCompany: () ==> ()testRemoveCompany() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** conference1 : Conference := **new** Conference("Conference 1", "Conference 1 details");

**dcl** company1 : Company := **new** Company("Company 1");

-- for tests supposed to fail

/\***dcl** company2 : Company := **new** Company("Company 2");\*/

AddConference(conference1);

AddCompany(conference1,company1);

assertEqual(1, **card** conference1.GetCompanies()); assertEqual({company1}, conference1.GetCompanies());

RemoveCompany(conference1,company1);

assertEqual(0, **card** conference1.GetCompanies()); assertEqual({}, conference1.GetCompanies());

-- this test is supposed to fail (can’t remove companies from a conference if they are not attending it)

/\*RemoveCompany(conference1,company2); \*/

);

-- test if the removal of talks from conferences is working correctly

**private** testRemoveTalk: () ==> ()testRemoveTalk() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** conference1 : Conference := **new** Conference("Conference 1", "Conference 1 details");

**dcl** talk1 : Talk := **new** Talk("Talk 1", "Talk 1 description", date1, time2, 40);

-- for tests supposed to fail

/\***dcl** talk2 : Talk := **new** Talk("Talk 2", "Talk 2 description", date1, time1, 20);\*/

AddConference(conference1);

AddTalk(conference1,talk1);

assertEqual(1, **card** conference1.GetTalks()); assertEqual({talk1}, conference1.GetTalks());

RemoveTalk(conference1,talk1);

assertEqual(0, **card** conference1.GetTalks());

assertEqual({}, conference1.GetTalks());

-- this test is supposed to fail (can’t remove talks from a conference if they don’t exist)

/\*RemoveTalk(conference1,talk2); \*/

);

-- test if the creation of attendees is working correctly

**private** testCreateAndAddAttendee: () ==> ()testCreateAndAddAttendee() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** attendee1 : Common := **new** Common("Ines");

**dcl** attendee2 : Influential := **new** Influential("Andreia Rodrigues", "STUDENT", **new** Company("feup"), "STUDENT AT FEUP", "PT");

-- tests gets and sets

assertEqual("Ines", attendee1.GetName()); attendee1.SetName("Ines Gomes");

assertEqual("Ines Gomes", attendee1.GetName());

AddAttendee(attendee1);

assertEqual(1, GetTotalAttendees());

assertEqual({attendee1}, webSummit.GetAttendees());

assertEqual(attendee1, webSummit.GetAttendee("Ines Gomes"));

AddAttendee(attendee2);

-- tests gets and sets

assertEqual("STUDENT", attendee2.GetJobPosition()); attendee2.SetJobPosition("Student");

assertEqual("Student", attendee2.GetJobPosition());

assertEqual("feup", attendee2.GetCompany().GetName()); attendee2.SetCompany(**new** Company("FEUP"));

assertEqual("FEUP", attendee2.GetCompany().GetName());

assertEqual("STUDENT AT FEUP", attendee2.GetDescription()); attendee2.SetDescription("Student At FEUP");

assertEqual("Student At FEUP", attendee2.GetDescription());

assertEqual("PT", attendee2.GetCountry());

attendee2.SetCountry("Portugal");

assertEqual("Portugal", attendee2.GetCountry());

* get total attendees

assertEqual(2, GetTotalAttendees());

* this test is supposed to fail (can’t add an already existing attendee to a talk)

/\*AddAttendee(attendee2);\*/

);

**private** testCreateAndAddStartup : () ==> ()testCreateAndAddStartup() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** startup1: Startup := **new** Startup("Startup Name","Startup Description","www.startup.pt","PT");

-- for tests supposed to fail

/\***dcl** startup2: Startup := **new** Startup("Emitu","Startup Description","www.startup.pt","PT");\*/

-- tests gets and sets

assertEqual("Startup Name", startup1.GetName()); startup1.SetName("Emitu");

assertEqual("Emitu", startup1.GetName());

assertEqual("Startup Description", startup1.GetDescription()); startup1.SetDescription("New Description");

assertEqual("New Description", startup1.GetDescription());

assertEqual("www.startup.pt", startup1.GetWebsite()); startup1.SetWebsite("www.startup.en"); assertEqual("www.startup.en", startup1.GetWebsite());

assertEqual("PT", startup1.GetCountry()); startup1.SetCountry("EN");

assertEqual("EN", startup1.GetCountry());

-- test add

assertEqual({}, GetStartups()); AddStartup(startup1);

assertEqual({startup1}, GetStartups());

assertEqual(startup1, webSummit.GetStartup("Emitu"));

-- test to fail : add startup with the same name

/\*AddStartup(startup2);\*/

);

**private** testRemoveStartup : () ==> ()testRemoveStartup() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** startup1: Startup := **new** Startup("Startup Name","Startup Description","www.startup.pt","PT");

--prepare test

AddStartup(startup1);

* test remove

RemoveStartup(startup1); assertEqual({}, GetStartups());

* test to fail : remove startup that doesn’t exist

/\*RemoveStartup(startup1);\*/

);

**private** testCreateAndAddInvestors : () ==> ()testCreateAndAddInvestors() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** investor : Influential := **new** Influential("Mark Zuckerberg","CEO",**new** Company("facebook"),"Speaker Description","EN");

assertEqual({}, GetInvestors()); AddInvestor(investor);

assertEqual({investor}, GetInvestors());

);

**private** testCreateAndAddNews: () ==> ()testCreateAndAddNews() == (

**dcl** webSummit: WebSummit := WebSummit‘ClearInstance(date1,date2);

**dcl** new1: New := **new** New("Title","Body");

-- tests gets and sets

assertEqual("Title", new1.GetTitle()); new1.SetTitle("New Title");

assertEqual("New Title", new1.GetTitle());

assertEqual("Body", new1.GetBody()); new1.SetBody("New Body");

assertEqual("New Body", new1.GetBody());

assertEqual({}, GetNews()); AddNew(new1);

assertEqual({new1}, GetNews());

);

**private** AddConference: Conference ==> ()AddConference(c) == ( WebSummit‘GetInstance().AddConference(c); );

**private** AddTalk: Conference\*Talk ==> ()AddTalk(c, t) == ( WebSummit‘GetInstance().AddTalk(c, t); );

**private** RemoveTalk: Conference\*Talk ==> ()RemoveTalk(c, t) == ( WebSummit‘GetInstance().RemoveTalk(c, t); );

**private** AddCompany: Conference\*Company ==> ()AddCompany(conf, c) == ( WebSummit‘GetInstance().AddCompany(conf, c); );

**private** RemoveCompany: Conference\*Company ==> ()RemoveCompany(conf, c) == ( WebSummit‘GetInstance().RemoveCompany(conf, c); );

**private** AddAttendee: Attendee ==> ()

AddAttendee(a) == ( WebSummit‘GetInstance().AddAttendee(a); );

**private** AddAttendee: Talk\*Attendee ==> ()

AddAttendee(t, a) == ( WebSummit‘GetInstance().AddAttendee(t, a); );

**private** AddSpeaker: Talk\*Influential ==> ()

AddSpeaker(talk, s) == ( WebSummit‘GetInstance().AddSpeaker(talk, s); );

**private** RemoveSpeaker: Talk\*Influential ==> ()

RemoveSpeaker(talk, s) == ( WebSummit‘GetInstance().RemoveSpeaker(talk, s); );

**private** AddStartup: Startup ==> ()

AddStartup(s) == ( WebSummit‘GetInstance().AddStartup(s); );

**private** RemoveStartup: Startup ==> ()

RemoveStartup(s) == ( WebSummit‘GetInstance().RemoveStartup(s); );

**private** GetStartups: () ==> **set of** Startup

GetStartups() == ( WebSummit‘GetInstance().GetStartups(); );

**private** AddInvestor: Influential ==> ()

AddInvestor(i) == ( WebSummit‘GetInstance().AddInvestor(i); );

**private** GetInvestors: () ==> **set of** Influential

GetInvestors() == ( WebSummit‘GetInstance().GetInvestors(););

**private** AddNew: New ==> ()

AddNew(n) == ( WebSummit‘GetInstance().AddNew(n); );

**private** GetNews: () ==> **set of** New

GetNews() == ( WebSummit‘GetInstance().GetNews(); );

**private** GetTotalAttendees: () ==> **int**

GetTotalAttendees() == ( WebSummit‘GetInstance().GetTotalAttendees(); );

**private** GetTotalAttendeesByTalk: Talk ==> **int**

GetTotalAttendeesByTalk(talk) == (WebSummit‘GetInstance().GetTotalAttendeesByTalk(talk); );

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **functions** | | |  | |  | |  |  | | |  |
| **traces** | | |  | |  | |  |  | | |  |
| **end** WebSummitTest | | |  | |  | |  |  | | |  |
|  | Function or operation | Line | | Coverage | | Calls | | |  |
|  |  |  | |  | |  | | |  |  | |
|  | [AddAttende](about:blank)e | 473 | | 0.0% | | 0 | | |  |  | |
|  | [AddCompan](about:blank)y | 463 | | 100.0% | | 2 | | |  |  | |
|  | [AddConferenc](about:blank)e | 448 | | 100.0% | | 7 | | |  |  | |
|  | [AddInvesto](about:blank)r | 508 | | 100.0% | | 1 | | |  |  | |
|  | [AddNe](about:blank)w | 518 | | 100.0% | | 1 | | |  |  | |
|  | [AddSpeake](about:blank)r | 483 | | 100.0% | | 1 | | |  |  | |
|  | [AddStartu](about:blank)p | 493 | | 100.0% | | 2 | | |  |  | |
|  | [AddTal](about:blank)k | 453 | | 100.0% | | 10 | | |  |  | |
|  | [GetInvestor](about:blank)s | 513 | | 100.0% | | 2 | | |  |  | |
|  | [GetNew](about:blank)s | 523 | | 100.0% | | 2 | | |  |  | |
|  | [GetStartup](about:blank)s | 503 | | 100.0% | | 3 | | |  |  | |
|  | [GetTotalAttendee](about:blank)s | 528 | | 100.0% | | 2 | | |  |  | |
|  | [GetTotalAttendeesByTal](about:blank)k | 533 | | 0.0% | | 0 | | |  |  | |
|  | [RemoveCompan](about:blank)y | 468 | | 100.0% | | 1 | | |  |  | |
|  | [RemoveSpeake](about:blank)r | 488 | | 100.0% | | 1 | | |  |  | |
|  | [RemoveStartu](about:blank)p | 498 | | 100.0% | | 1 | | |  |  | |
|  | [RemoveTal](about:blank)k | 458 | | 100.0% | | 1 | | |  |  | |
|  | [mai](about:blank)n | 17 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddAttende](about:blank)e | 324 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddCompan](about:blank)y | 243 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddConferenc](about:blank)e | 68 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddInvestor](about:blank)s | 419 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddNew](about:blank)s | 429 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddStartu](about:blank)p | 367 | | 100.0% | | 1 | | |  |  | |
|  | [testCreateAndAddTal](about:blank)k | 107 | | 100.0% | | 1 | | |  |  | |
|  | [testRemoveCompan](about:blank)y | 272 | | 100.0% | | 3 | | |  |  | |
|  | [testRemoveStartu](about:blank)p | 403 | | 100.0% | | 1 | | |  |  | |
|  | [testRemoveTal](about:blank)k | 298 | | 100.0% | | 1 | | |  |  | |
|  | [testSchedule](about:blank)s | 179 | | 100.0% | | 1 | | |  |  | |
|  |  |  | |  | |  | | |  |  | |
|  | WebSummitTest.vdmpp |  | | 99.2% | | 51 | | |  |  | |

*Tabela 21 : Cobertura da classe WebSummitTest*

## **Resultados**

|  |  |
| --- | --- |
| **Teste** | **Descrição** |
| **testCreateAndAddConference** | Verifica se o requisito **R02** está a funcionar corretamente. |
| **testCreateAndAddTalk** | Verifica se os requisitos **R03**, **R05** e **R13** estão a funcionar corretamente. |
| **testRemoveTalk** | Verifica se o requisito **R14** está a funcionar corretamente. |
| **testSchedules** | Verifica se os requisitos **R10**, **R11** e **R12** estão a funcionar corretamente. |
| **testCreateAndAddCompany** | Verifica se o requisito **R04** está a funcionar corretamente. |
| **testRemoveCompany** | Verifica se o requisito **R17** está a funcionar corretamente. |
| **testCreateAndAddAttendee** | Verifica se os requisitos **R01** e **R06** estão a funcionar corretamente. |
| **testCreateAndAddStartup** | Verifica se o requisito **R07** está a funcionar corretamente. |
| **testRemoveStartup** | Verifica se o requisito **R16** está a funcionar corretamente. |
| **testCreateAndAddInvestors** | Verifica se o requisito **R08** está a funcionar corretamente. |
| **testRemoveInvestors** | Verifica se os requisitos **R15** e **R09** estão a funcionar corretamente. |
| **testCreateAndAddNews** | Verifica se os requisitos **R18** e **R19** estão a funcionar corretamente |

*Tabela 22 : Tabela que relaciona os testes com os requisitos observados na* [*tabela de requisitos*](#_3dy6vkm)

# **Verificação do Modelo**

## **Exemplo de uma verificação de domínio**

Uma das *proof obligation* geradas pelo Overture foi:

|  |  |  |
| --- | --- | --- |
| **No.** | **PO Name** | **Type** |
| 60 | WebSummit`GetSchedule(Utilities`Date, Utilities`Time) | legal map application |

*Tabela 23 : Exemplo de uma proof obligation*

O código apresentado com as partes relevantes da map application sublinhadas:

-- returns the event schedule by date/time

**public** GetSchedule : Utilities‘Date\*Utilities‘Time ==> **seq of** Talk

GetSchedule (date, time) == (

**dcl** temp: **seq of** Talk := [];

-- joins all talks from that day starting or occuring at the given time

**for all** conference **in set** conferences **do** (

**for all** talk **in set elems** conference.GetSchedule()(date) **do**(

**if**(talk.GetTime().hour = time.hour)

**then** temp := temp ˆ [talk]

**else**

**if**(talk.GetTime().hour + 1 = time.hour)

**then if**((talk.GetDuration()) >= (60 - talk.GetTime().minute))

**then if**((talk.GetTime().minute + talk.GetDuration() - 60) <= 60)

**then** temp := temp ˆ [talk]

);

);

-- orders them by time

temp := Utilities‘mergeSortTalks(temp);

**return** temp;

)

**pre forall** conference **in set** conferences & date **in set** (**dom** (conference.GetSchedule)()) ;

Neste caso podemos comprovar facilmente utilizando a pré-condição

**‘ pre forall** conference **in set** conferences & date **in set** (**dom** (conference.GetSchedule)()); ‘ asseguramos que o map é acedido apenas dentro do seu domínio.

## **Exemplo de uma verificação de invariante**

Uma das *proof obligation* geradas pelo Overture foi:

|  |  |  |
| --- | --- | --- |
| **No.** | **PO Name** | **Type** |
| 68 | WebSummit`AddConference(Conference) | state invariant holds |

*Tabela 24 : Exemplo de uma proof obligation para a verificação de invariante*

O código apresentado com as mudanças de estado relevantes sublinhadas:

-- creates a new conference

**public** AddConference : Conference ==> ()

AddConference (conference) == (

conferences := conferences **union** {conference};

)

**pre** conference **not in set** conferences **and** notAlreadyExistent(conference) = **true**

**post** conferences = conferences˜ **union** {conference};

A invariante em análise é a seguinte:

**‘ inv not exists** c1, c2 **in set** conferences & c1 <> c2 **and** c1.GetName() = c2.GetName(); **‘**

A mudança de estado em análise é relativa à adição de uma nova conferência, não previamente existente , ao conjunto de conferências que fazem parte do evento WebSummit:

conferences = conferences **union** {conference};

Temos de provar que depois da execução deste pedaço de código que a invariante se mantém, ou seja, que no conjunto de conferências que fazem parte do evento WebSummit, não existem 2 conferências iguais ou com nomes iguais. Como na pré-condição definimos que para esta função ser executada a conferência tem de ser diferente das já existentes e que não pode existir uma conferência com o mesmo nome da que queremos adicionar

(**forall** conference:Conference & (((conference **not in set** conferences)

**and** (notAlreadyExistent(conference) = **true**)) =>

(((**not** (**exists** c1, c2 **in set** conferences & ((c1 <> c2) **and** ((c1.GetName)() = (c2.GetName)()))))

, podemos concluir que a invariante se mantém ao inserir uma nova conferência no conjunto de conferências do evento.

# **Geração de Código**

A geração de código Java decorreu sem problemas, não sendo necessária nenhuma alteração ao código gerado. Para facilitar a interpretação do mesmo, criamos a classe *Interface.java* descrita no tópico seguinte. Para usar a interface apenas é necessário criar o ficheiro *Interface.java*, copiar o código do tópico seguinte e adicionar a linha de código

‘ new Interface(); ‘

à função *Run()* do ficheiro *Main.java*.

## **Interface.java**

package MFES;

import java.util.ArrayList;

import java.util.HashSet;

import java.util.Map;

import java.util.Scanner;

import java.util.Set;

import org.overture.codegen.runtime.VDMSeq;

public class Interface {

private WebSummit websummit = WebSummit.GetInstance();

private Scanner scanner = new Scanner(System.in);

private Utilities.Date startDate = new Utilities.Date(2017L, 9L, 1L);

private Utilities.Date endDate = new Utilities.Date(2017L, 9L, 3L);

public Interface()

{

Conference c1 = new Conference("ROBOTICS","This conference is just about robots");

Conference c2 = new Conference("Cyber Security","This conference is just about Cyber Security");

Talk t1 = new Talk("Sophia Robot","Talk about Sophia Robot",new Utilities.Date(2017L, 9L, 1L),new Utilities.Time(15L, 30L),30L);

Talk t2 = new Talk("IOT","Internet Of Things",new Utilities.Date(2017L, 9L, 1L),new Utilities.Time(16L, 30L),30L);

Talk t3 = new Talk("Is Our Home safe?","Complex discussion about the dangers of internet related with our homes.",new Utilities.Date(2017L, 9L, 2L),new Utilities.Time(16L, 30L),30L);

Influential s1 = new Influential("Mark Zuckemberg","CEO",new Company("Facebook"),"Social Networks lover","USA");

Influential s2 = new Influential("Bill Gates","CEO",new Company("Microsoft"),"My description","USA");

//some data added

websummit.SetDates(startDate, endDate);

websummit.AddConference(c1);

websummit.AddConference(c2);

websummit.AddTalk(c1, t1);

websummit.AddTalk(c1, t2);

websummit.AddTalk(c2, t3);

websummit.AddSpeaker(t1, s1);

websummit.AddSpeaker(t2, s1);

websummit.AddSpeaker(t3, s1);

websummit.AddSpeaker(t3, s2);

websummit.AddAttendee(t1, new Common("Ines"));

websummit.AddAttendee(t1, new Common("Andreia"));

printMainMenu();

}

public void printMainMenu()

{

ArrayList<String> options = new ArrayList<String>();

options.add("Administration");

options.add("Register Common Attendee");

options.add("Register Influential Attendee");

options.add("Attend Event");

options.add("Startups");

options.add("EXIT");

System.out.println(" ====================== ");

System.out.println(" === WebSummit Menu === ");

System.out.println(" ====================== \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printAdministrationMenu();

break;

case 2:

printCommonRegister();

break;

case 3:

printInfluentialRegister();

break;

case 4:

isParticipant();

break;

case 5:

printStartupMenu();

break;

case 6:

return;

default:

break;

}

}

public void printAdministrationMenu()

{

ArrayList<String> options = new ArrayList<String>();

options.add("News");

options.add("Conferences");

options.add("Total Attendees");

options.add("BACK");

System.out.println(" === ADMINISTRATION === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printNewsMenu();

break;

case 2:

printConferencesMenu();

break;

case 3:

printTotalAttendees();

break;

case 4:

printMainMenu();

break;

default:

break;

}

}

public void printNewsMenu()

{

ArrayList<String> options = new ArrayList<String>();

options.add("Add New");

options.add("Get All News");

options.add("BACK");

System.out.println(" === ADMINISTRATION / NEWS === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printAddNew();

break;

case 2:

printAllNews();

break;

case 3:

printAdministrationMenu();

break;

default:

break;

}

}

public void printAddNew()

{

System.out.println(" === ADMINISTRATION / NEWS / ADD NEW === \n");

System.out.println("Title : ");

String title = scanner.nextLine();

System.out.println("Body : ");

String body = scanner.nextLine();

New myNew = new New(title,body);

websummit.AddNew(myNew);

waitOk();

printNewsMenu();

}

public void printAllNews()

{

System.out.println(" === ADMINISTRATION / NEWS / GET ALL NEWS === \n");

Set<New> news = websummit.GetNews();

for(New myNew: news)

{

System.out.println(myNew.toString());

System.out.println("\n ~~~ \n");

}

waitOk();

printNewsMenu();

}

public void printConferencesMenu()

{

ArrayList<String> options = new ArrayList<String>();

options.add("Add New Conference");

options.add("Organize Conferences");

options.add("BACK");

System.out.println(" === ADMINISTRATION / CONFERENCES === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printAddConference();

break;

case 2:

printAllConferences();

break;

case 3:

printAdministrationMenu();

break;

default:

break;

}

}

public void printAddConference()

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ADD CONFERENCE === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

System.out.println("Description : ");

String des = scanner.nextLine();

Conference c = new Conference(name,des);

websummit.AddConference(c);

waitOk();

printConferencesMenu();

}

public void printAllConferences()

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE === \n");

Set<Conference> conferences = websummit.GetConferences();

if(conferences.size() == 0){

System.out.println("No conferences to show.");

waitOk();

printConferencesMenu();

}

else

{

int i = 0;

for(Conference c : conferences){

i++;

System.out.println(" "+i+". "+c.toString());

}

//user input

int number = getUserInput(conferences.size());

//get conference

i = 0;

Conference conference = null;

for(Conference c : conferences){

i++;

if(i == number){

conference = c;

break;

}

}

printConferenceMenu(conference);

}

}

public void printConferenceMenu(Conference conference)

{

ArrayList<String> options = new ArrayList<String>();

options.add("Add New Talk");

options.add("Organize Talks");

options.add("Add Company");

options.add("Remove Company");

options.add("BACK");

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printAddTalk(conference);

break;

case 2:

printAllTalks(conference);

break;

case 3:

printAddCompany(conference);

break;

case 4:

printRemoveCompany(conference);

break;

case 5:

printConferencesMenu();

break;

default:

break;

}

}

public void printAddTalk(Conference conference)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ADD TALK === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

System.out.println("Description : ");

String des = scanner.nextLine();

System.out.println("Date (day): ");

String day = scanner.nextLine();

long dayn = Long.parseLong(day);

System.out.println("Date (month): ");

String mon = scanner.nextLine();

long monn = Long.parseLong(mon);

System.out.println("Date (year): ");

String year = scanner.nextLine();

long yearn = Long.parseLong(year);

Utilities.Date date = new Utilities.Date(yearn, monn, dayn);

System.out.println("Start Time (Hour): ");

String hour = scanner.nextLine();

long hourn = Long.parseLong(hour);

System.out.println("Start Time (Minutes): ");

String min = scanner.nextLine();

long minn = Long.parseLong(min);

Utilities.Time time = new Utilities.Time(hourn, minn);

System.out.println("Duration");

String dur = scanner.nextLine();

long durn = Long.parseLong(dur);

Talk t = new Talk(name,des,date,time,durn);

websummit.AddTalk(conference, t);

waitOk();

printConferenceMenu(conference);

}

public void printAllTalks(Conference conference)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ORGANIZE TALKS === \n");

Set<Talk> talks = conference.GetTalks();

if(talks.size() == 0){

System.out.println("No talks to show.");

waitOk();

printConferencesMenu();

}

else{

//display menu

int i = 0;

for(Talk t : talks){

i++;

System.out.println(" "+i+". "+t.toString());

}

//user input

int number = getUserInput(talks.size());

//get conference

i = 0;

Talk talk = null;

for(Talk t : talks){

i++;

if(i == number){

talk = t;

break;

}

}

printTalkMenu(talk);

}

}

public void printTalkMenu(Talk talk)

{

ArrayList<String> options = new ArrayList<String>();

options.add("Add New Speaker");

options.add("Remove Speaker");

options.add("Get Total Attendees");

options.add("Remove Talk");

options.add("BACK");

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ORGANIZE TALK === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printAddSpeaker(talk);

break;

case 2:

printRemoveSpeaker(talk);

break;

case 3:

printTotalAttendees(talk);

break;

case 4:

printRemoveTalk(talk);

break;

case 5:

Conference c = websummit.GetConference(talk.GetConference());

printConferenceMenu(c);

break;

default:

break;

}

}

public void printAddSpeaker(Talk talk)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ORGANIZE TALK / ADD SPEAKER === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

System.out.println("Description : ");

String des = scanner.nextLine();

System.out.println("Company Name : ");

String compName = scanner.nextLine();

Company company = new Company(compName);

System.out.println("Job Position : ");

String jobPos = scanner.nextLine();

System.out.println("Country : ");

String country = scanner.nextLine();

Influential speaker = new Influential(name,jobPos,company,des,country);

websummit.AddSpeaker(talk, speaker);

waitOk();

printTalkMenu(talk);

}

public void printRemoveSpeaker(Talk talk)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ORGANIZE TALKS / REMOVE SPEAKER === \n");

Set<Influential> speakers = talk.GetSpeakers();

if(speakers.size() == 0){

System.out.println("No speakers to show.");

waitOk();

printTalkMenu(talk);

}

else{

int i = 0;

for(Influential s : speakers){

i++;

System.out.println(" "+i+". "+s.toString());

}

//user input

int number = getUserInput(speakers.size());

//get conference

i = 0;

Influential speaker = null;

for(Influential s : speakers){

i++;

if(i == number){

speaker = s;

break;

}

}

websummit.RemoveSpeaker(talk, speaker);

printTalkMenu(talk);

}

}

public void printTotalAttendees(Talk talk)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ORGANIZE TALKS / GET TOTAL ATTENDEES === \n");

System.out.println("\nTotal Attendees : "+websummit.GetTotalAttendeesByTalk(talk));

waitOk();

printTalkMenu(talk);

}

public void printRemoveTalk(Talk talk)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ORGANIZE TALKS / REMOVE TALK === \n");

Conference c = websummit.GetConference(talk.GetConference());

websummit.RemoveTalk(c, talk);

waitOk();

printConferenceMenu(c);

}

public void printAddCompany(Conference conference)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / ADD COMPANY === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

Company company = new Company(name);

websummit.AddCompany(conference, company);

waitOk();

printConferenceMenu(conference);

}

public void printRemoveCompany(Conference conference)

{

System.out.println(" === ADMINISTRATION / CONFERENCES / ORGANIZE CONFERENCE / REMOVE COMPANY === \n");

Set<Company> companies = conference.GetCompanies();

if(companies.size() == 0)

{

System.out.println("No companies to show.");

waitOk();

printConferenceMenu(conference);

}

else

{

int i = 0;

for(Company c : companies){

i++;

System.out.println(" "+i+". "+c.toString());

}

//user input

int number = getUserInput(companies.size());

//get conference

i = 0;

Company company = null;

for(Company c : companies){

i++;

if(i == number){

company = c;

break;

}

}

websummit.RemoveCompany(conference, company);

printConferenceMenu(conference);

}

}

public void printTotalAttendees()

{

System.out.println(" === ADMINISTRATION / GET TOTAL ATTENDEES === \n");

Set<Attendee> attendees = websummit.GetAttendees();

for(Attendee a : attendees)

System.out.println(a.toString());

System.out.println("\n Total : "+attendees.size()+ " attendees");

waitOk();

printAdministrationMenu();

}

public void printCommonRegister()

{

System.out.println(" === COMMON ATTENDEE / REGISTER === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

Common c = new Common(name);

websummit.AddAttendee(c);

waitOk();

printMainMenu();

}

public void printInfluentialRegister()

{

System.out.println(" === INFLUENTIAL ATTENDEE / REGISTER === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

System.out.println("Description : ");

String des = scanner.nextLine();

System.out.println("Company Name : ");

String compName = scanner.nextLine();

Company company = new Company(compName);

System.out.println("Job Position : ");

String jobPos = scanner.nextLine();

System.out.println("Country : ");

String country = scanner.nextLine();

Influential i = new Influential(name,jobPos,company,des,country);

websummit.AddAttendee(i);

waitOk();

printMainMenu();

}

public void isParticipant()

{

System.out.println(" === ATTENDEE / ATTEND CONFERENCES === \n");

System.out.println("Participant name : ");

String name = scanner.nextLine();

Set<Attendee> attendees = websummit.GetAttendees();

Attendee a = websummit.GetAttendee(name);

if(a != null){

System.out.println("Success!");

waitOk();

printParticipantMenu(a);

}

else{

System.out.println("Invalid Participant Name!");

waitOk();

printMainMenu();

}

}

public void printParticipantMenu(Attendee attendee)

{

ArrayList<String> options = new ArrayList<String>();

options.add("Register to talk");

options.add("Get WebSummit Schedule");

options.add("Get Conference Schedule");

options.add("Get Talks By Hour");

options.add("Get Exhibit");

options.add("BACK");

System.out.println(" === ATTENDEE / ATTEND CONFERENCES === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printTalksToRegister(attendee);

break;

case 2:

printWebSummitSchedule(attendee);

break;

case 3:

printConferencesToSchedule(attendee);

break;

case 4:

printScheduleByHour(attendee);

break;

case 5:

printExhibit(attendee);

break;

case 6:

printMainMenu();

default:

break;

}

}

public void printTalksToRegister(Attendee attendee)

{

System.out.println(" === ATTENDEE / ATTEND CONFERENCES / REGISTER TO TALK === \n");

Set<Conference> conferences = websummit.GetConferences();

if(conferences.size() == 0){

System.out.println("No talks to show.");

waitOk();

printParticipantMenu(attendee);

}

else

{

Set<Talk> allTalks = new HashSet<Talk>();

int i = 0;

for(Conference c : conferences)

{

Set<Talk> talks = c.GetTalks();

for(Talk t : talks)

{

i++;

System.out.println(" "+i+". "+t.toString());

allTalks.add(t);

}

}

//user input

int number = getUserInput(allTalks.size()+1);

//get talk

Talk talk = null;

i = 0;

for(Talk t : allTalks){

i++;

if(i == number){

talk = t;

break;

}

}

websummit.AddAttendee(talk, attendee);

System.out.println("Successfully registered to talk!");

waitOk();

printParticipantMenu(attendee);

}

}

public void printWebSummitSchedule(Attendee attendee)

{

System.out.println(" === ATTENDEE / ATTEND CONFERENCES / WEBSUMMIT SCHEDULE === \n");

printSchedule(websummit.GetSchedule());

waitOk();

printParticipantMenu(attendee);

}

public void printConferencesToSchedule(Attendee attendee)

{

System.out.println(" === ATTENDEE / ATTEND CONFERENCES / CONFERENCE SCHEDULE === \n");

Set<Conference> conferences = websummit.GetConferences();

if(conferences.size() == 0){

System.out.println("No conferences to show.");

waitOk();

printParticipantMenu(attendee);

}

else

{

int i = 0;

for(Conference c : conferences){

i++;

System.out.println(" "+i+". "+c.toString());

}

//user input

int number = getUserInput(conferences.size());

//get conference

i = 0;

Conference conference = null;

for(Conference c : conferences){

i++;

if(i == number){

conference = c;

break;

}

}

printSchedule(websummit.GetSchedule(conference));

waitOk();

printParticipantMenu(attendee);

}

}

public void printScheduleByHour(Attendee attendee)

{

System.out.println(" === ATTENDEE / ATTEND CONFERENCES / TALKS BY HOUR === \n");

System.out.println("Date (day): ");

String day = scanner.nextLine();

long dayn = Long.parseLong(day);

System.out.println("Date (month): ");

String mon = scanner.nextLine();

long monn = Long.parseLong(mon);

System.out.println("Date (year): ");

String year = scanner.nextLine();

long yearn = Long.parseLong(year);

Utilities.Date date = new Utilities.Date(yearn, monn, dayn);

System.out.println("Start Time (Hour): ");

String hour = scanner.nextLine();

long hourn = Long.parseLong(hour);

System.out.println("Start Time (Minutes): ");

String min = scanner.nextLine();

long minn = Long.parseLong(min);

Utilities.Time time = new Utilities.Time(hourn, minn);

VDMSeq schedule = websummit.GetSchedule(date, time);

System.out.println("Results : \n");

for(int i = 0; i < schedule.size();i++){

System.out.println(schedule.get(i));

}

waitOk();

printParticipantMenu(attendee);

}

public void printSchedule(Map<Utilities.Date,VDMSeq> schedule)

{

for (Map.Entry<Utilities.Date,VDMSeq> entry : schedule.entrySet())

{

System.out.println(entry.getKey().day+" / "+entry.getKey().month+" / "+entry.getKey().year);

VDMSeq talks = entry.getValue();

for(int i = 0; i < talks.size(); i++){

System.out.println(" "+talks.get(i));

}

System.out.println();

}

}

public void printExhibit(Attendee attendee)

{

System.out.println(" === ATTENDEE / EXHIBIT === \n");

Set<Startup> startups = websummit.GetStartups();

for(Startup s : startups)

System.out.println(s.toString());

waitOk();

printParticipantMenu(attendee);

}

// --- STARTUPS ---

public void printStartupMenu()

{

ArrayList<String> options = new ArrayList<String>();

options.add("Register");

options.add("Cancel Registration");

options.add("BACK");

System.out.println(" === STARTUPS === \n");

for(int i = 1; i<= options.size(); i++){

System.out.println(" "+i+". "+options.get(i-1));

}

//user input

int number = getUserInput(options.size());

switch (number) {

case 1:

printStartupRegister();

break;

case 2:

printStartupRemove();

break;

case 3:

printMainMenu();

break;

default:

break;

}

}

public void printStartupRegister()

{

System.out.println(" === STARTUPS / REGISTER === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

System.out.println("Description : ");

String des = scanner.nextLine();

System.out.println("Website : ");

String website = scanner.nextLine();

System.out.println("Country : ");

String country = scanner.nextLine();

Startup s = new Startup(name,des,website,country);

websummit.AddStartup(s);

waitOk();

printMainMenu();

}

public void printStartupRemove()

{

System.out.println(" === STARTUPS / REMOVE REGISTRATION === \n");

System.out.println("Name : ");

String name = scanner.nextLine();

Startup s = websummit.GetStartup(name);

websummit.RemoveStartup(s);

waitOk();

printMainMenu();

}

public int getUserInput(int max)

{

boolean valid = false;

String input = "";

int number = 0;

while(!valid){

System.out.println("\nSelect your option :");

input = scanner.nextLine();

number = Integer.parseInt(input);

if(! (number < 1 || number > max) )

valid = true;

}

return number;

}

public void waitOk()

{

boolean valid = false;

String input = "";

while(!valid){

System.out.println("\nType 'ok' to continue :");

input = scanner.nextLine();

if(input.equals("ok"))

valid = true;

}

}

}

# **Conclusões**

O modelo desenvolvido cobre todos os requisitos especificados no enunciado.

Foram implementadas todas as principais características que definem o evento WebSummit: conferências onde estão agendadas palestras, montra de exibição e notícias acerca do evento.

Apesar de acharmos que o modelo está bastante completo, um exemplo de melhorias a fazer seria desenvolver mais a parte das notícias, associando empresas ou investidores aos assuntos abordados para o utilizador poder ter um acesso mais direto ás noticias que pretende ver.

O grupo trabalhou em conjunto e todos os elementos tiveram uma participação significativa no desenvolvimento do projeto:

Andreia Rodrigues - 50%

Inês Gomes - 50%

# **Referências**

1. Apontamentos da unidade curricular Métodos Formais de Engenharia de Software
2. Overture tool web site, <http://overturetool.org>
3. VDM-10 Language Manual, Peter Gorm Larsen et al, Overture Technical Report Series No. TR-001, March 2014
4. WebSummit web site, <https://websummit.com/>