Faculdade de Engenharia da Universidade do Porto

Métodos Formais em Engenharia de Software
mfes

WebSummit

Authors:
Joel Dinis (ei12064)

January 3, 2018

1 Descrição do Sistema

1.1 Descrição Informal do Sistema

O objetivo deste projeto é a modulação do evento tecnológico conhecido por WebSummit. É, pois, de esperar um modelo funcional e eficiente que represente e vá de encontro aos requisitos deste evento.

- Inscrição dos participantes nos eventos(conferências)
- Determinar as várias conferências a uma determinada hora
- Inscrever uma startup num determinado Showroom
- Atribuir um ou mais Speakers a um evento(Conferência)
- Criação de um evento(Conferência)

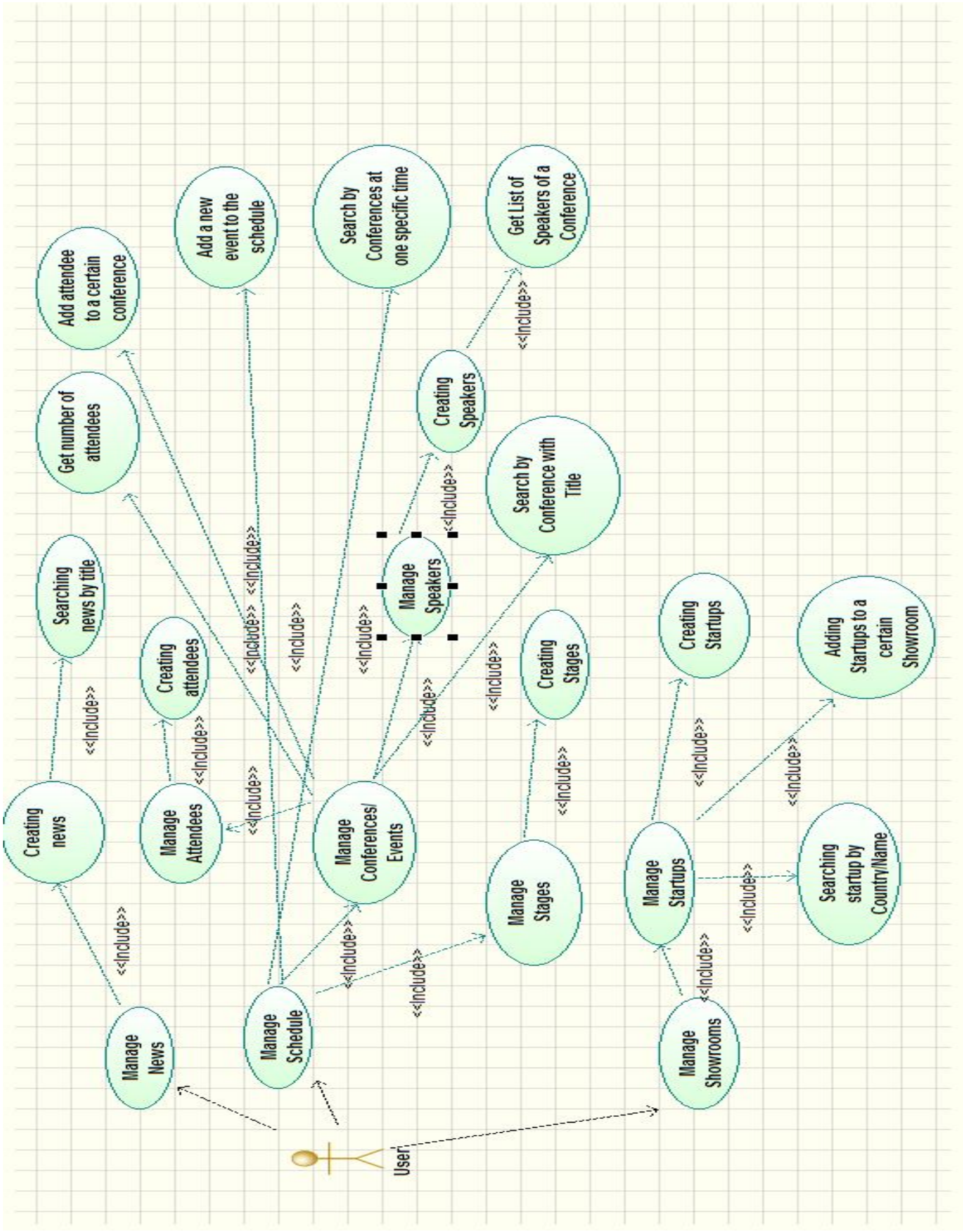
Estes são apenas um pequeno exemplo do que o nosso modelo consegue representar. Iremos, ao longo deste documento apresentar o resto.

1.2 Lista de Requisitos

Descritivo	**Prioridade**	**Descrição**
R01	Máxima	Criação de Eventos (Conferências)
R02	Máxima	Criação de Showrooms para as Startups
R03	Máxima	Criação de um jornal para as notícias
R04	Máxima	Obtenção do número de Attendees para cada Evento (Conferência)
R05	Máxima	Criação de palcos para as conferências
R06	Máxima	Criação de um horário(Schedule) geral para todos os Eventos
R07	Máxima	Adicionar Startups a uma Showroom
R08	Máxima	Criação de Speakers e Attendees
R09	Máxima	Obtenção de todos os eventos a uma determinada hora/dia
R10	Máxima	Adicionar Attendees a Eventos
R11	Máxima	Obtenção de um Evento usando como parâmetro de pesquisa o título do mesmo
R12	Máxima	Adicionar um Evento novo ao horário
R13	Máxima	Obter capacidade de um palco
R14	Máxima	Pesquisar startups por Nome/País
R15	Máxima	Pesquisar notícias por títulos
R16	Máxima	Pesquisar por conferências apresentadas por uma determinada pessoa

2 Visual UML model

2.1 Use case mode



Cenário	**Configuração**
Descrição	Adicionar notícias ao blog WebSummit
Pré-Condições	A notícia não pode já estar presente no blog
Pós-Condições	A notícia tem de estar presente no blog (final state system)
Passos	No menu "News" criar uma nova notícia, com o título, texto e data e adiciona-a ao blog.
Exceções	A notícia já está no blog.

Cenário	**Configuração**
Descrição	Procurar notícias por Título
Pré-Condições	O input do utilizador deve ter pelo menos 1 caráter.
Pós-Condições	Nenhum
Passos	Escolher a opção "pesquisar por notícia com título" e inserir o título.
Exceções	O input do utilizador é vazio.

Cenário	**Configuração**
Descrição	Criação de Attendees
Pré-Condições	O Attendee não pode existir
Pós-Condições	Nenhuma
Passos	No menu "Attendees" escolher a opção "Criar um novo attendee".
Exceções	O Attendee já existe.

Cenário	**Configuração**
Descrição	Obter número de Attendees de uma dado Evento/Conferência
Pré-Condições	Nenhuma
Pós-Condições	O número de Attendees é igual ou superior a 0.
Passos	No menu "Conferências" escolher a opção "Ver número de Inscrições".
Exceções	Nenhuma.

Cenário	**Configuração**
Descrição	Adicionar Attendees a uma Conferência
Pré-Condições	O Attendee não pode já lá estar inscrito
Pós-Condições	É preciso verificar, se com a inscrição deste novo Attendee, a capacidade do Stage não é ultrapassada.
Passos	No menu "Conferências" escolher a opção "Criar um novo attendee".
Exceções	A capacidade do palco (Stage) é ultrapassada.

Cenário	**Configuração**
Descrição	Inserir um Evento (Conference) no horário (Schedule)
Pré-Condições	É preciso verificar se dos Attendees/Speakers que vão participar deste novo evento, já não estão inscritos noutro evento, pois não podem estar em 2 sítios ao mesmo tempo.
Pós-Condições	É preciso verificar se já não existe um evento, no mesmo palco (Stage), ao mesmo tempo.
Passos	No menu "Schedule" escolher a opção "Criar um novo evento".
Exceções	Se os attendees/speakers a inserir já estiverem inscritos noutro evento ao mesmo tempo, ou se o palco já estiver reservado para um outro evento à mesma hora.

Cenário	**Configuração**
Descrição	Pesquisar por eventos a uma determinada hora/dia
Pré-Condições	Nenhuma
Pós-Condições	Nenhuma.
Passos	No menu "Schedule" escolher a opção "Pesquisar eventos por dia/hora".
Exceções	Nenhuma.

Cenário	**Configuração**
Descrição	Obter Lista de Speakers para uma Conferência
Pré-Condições	Nenhuma
Pós-Condições	Nenhuma.
Passos	No menu "Schedule" escolher o evento desejado e clicar em "Ver Speakers".
Exceções	Nenhuma.

Cenário	**Configuração**
Descrição	Pesquisar por uma Conferência usando o Título
Pré-Condições	Nenhuma
Pós-Condições	Nenhuma.
Passos	No menu "Schedule" escolher a opção "Procurar Conferência" e inserir o título.
Exceções	A conferência não é encontrada.

Cenário	**Configuração**
Descrição	Criar palcos (Stages)
Pré-Condições	O palco não pode já estar criado
Pós-Condições	Nenhuma.
Passos	No menu "Stage" escolher a opção "Criar Stage" e inserir o nome e capacidade do mesmo.
Exceções	O palco já foi criado.

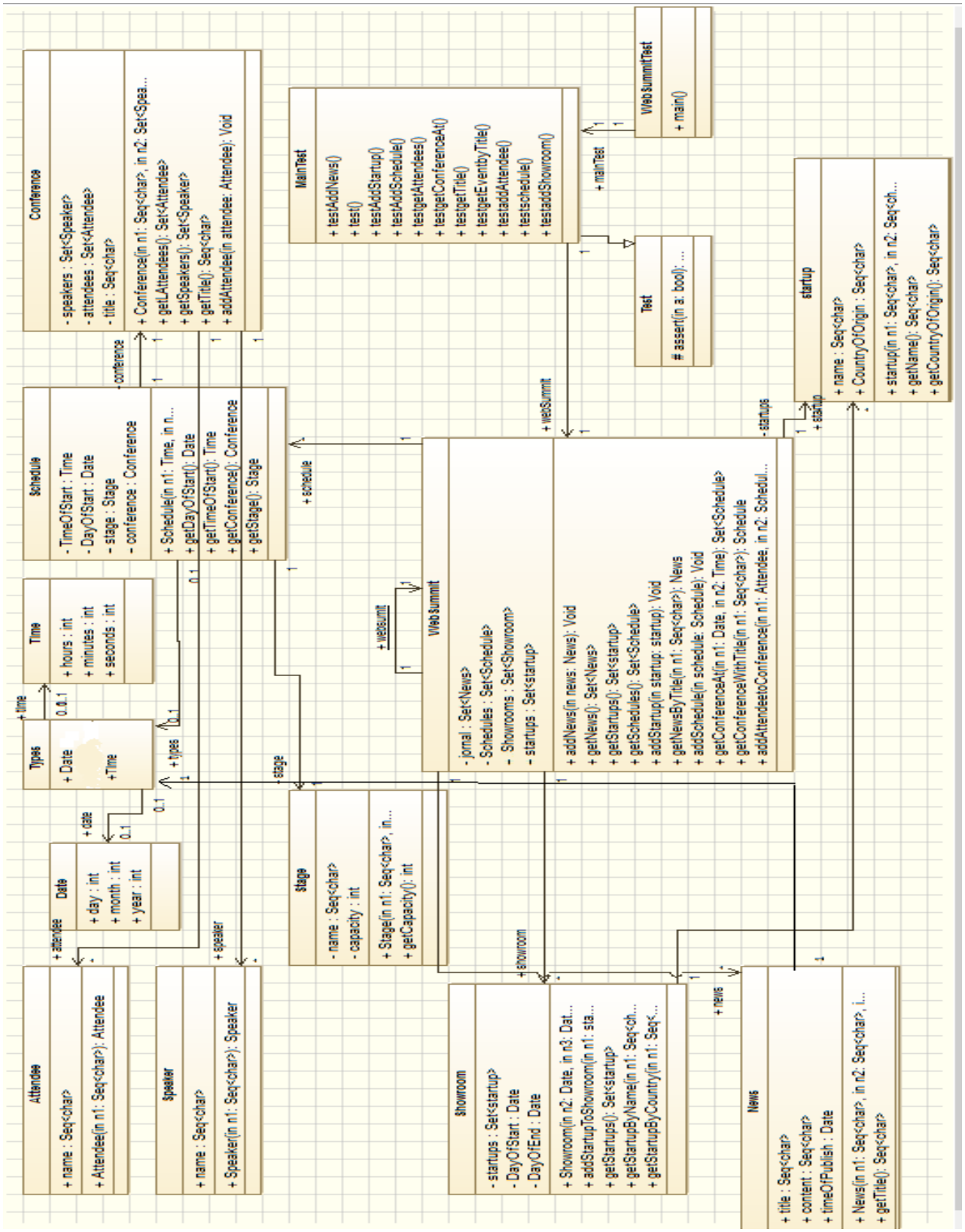
Cenário	**Configuração**
Descrição	Criar startups
Pré-Condições	A startup não pode já estar criada
Pós-Condições	Nenhuma.
Passos	No menu "Startup" escolher a opção "Criar nova startup" e inserir o nome e país de origem da mesma.
Exceções	Já existe uma startup com as configurações indicadas.

Cenário	**Configuração**
Descrição	Adicionar Startups a Showrooms
Pré-Condições	Nenhuma
Pós-Condições	Nenhuma.
Passos	No menu "Startup" escolher a opção "Criar nova startup" e inserir o nome e país de origem da mesma.
Exceções	Nenhuma.

Cenário	**Configuração**
Descrição	Pesquisar startups por Nome/País
Pré-Condições	Nenhuma
Pós-Condições	Nenhuma.
Passos	No menu "Startup" escolher a opção "Criar nova startup" e inserir o nome e país de origem da mesma.
Exceções	Nenhuma.

Cenário	**Configuração**
Descrição	Pesquisar por conferências apresentadas por uma determinada pessoa
Pré-Condições	Nenhuma
Pós-Condições	Nenhuma.
Passos	No menu "Schedule" escolher a opção "Pesquisar eventos por orador" e inserir o nome do mesmo.
Exceções	Nenhuma.

2.2 Class model



Cenário	**Configuração**
Attendee	Define um participante de uma Conferência/Evento
Speaker	Define um palestrante de uma Conferência/Evento
Types	Classe usada para definir vários tipos úteis ao projeto
Schedule	Define uma parte do horário e atividades do evento
Conference	Define a palestra, os participantes, os palestrantes e o título da palestre/conferência
Showroom	Classe que representa um evento de mostra de startups.
Stage	Classe que representa um palco
News	Classe que representa uma notícia
startup	Classe que representa uma startup
Test	Classe que define o Assert
WebSummitTest	Classe que recebe o entry point de execução e chama o MainTest
MainTest	Classe principal de execução de testes

3 Formal VDM++ model

3.1 Class Attendee

```
class Attendee -- Define um participante de uma palestra
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
    public name: seq of char;
operations
-- TODO Define operations here
-- Construtor do participante
    public Attendee: seq of char ==> Attendee
  Attendee(n1) == (name := n1; return self)
  post name = n1;

functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Attendee
```

3.2 Class startup

```
class startup
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
-- Representa uma startup
  public name: seq of char;
  public CountryOfOrigin: seq of char;

operations
-- TODO Define operations here
  public startup: seq of char * seq of char ==> startup
  startup(n1,n2) == (name := n1; CountryOfOrigin := n2; return self)
  post name = n1 and CountryOfOrigin = n2;

  public getName:() ==> seq of char
  getName() == return name;

  public getCountryOfOrigin:() ==> seq of char
  getCountryOfOrigin() == return CountryOfOrigin;

functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end startup
```

3.3 Class Conference

```
class Conference -- Define um Evento/Conferência presente na Schedule
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
    private speakers: set of Speaker :={};
    private attendees: set of Attendee :={};
    private title: seq of char;

operations
-- TODO Define operations here
    public Conference:  seq of char*set of Speaker * set of Attendee ==> Conference
    Conference(n1,n2,n3) == (title := n1;speakers := n2; attendees := n3; return self)
    post title = n1 and speakers = n2 and attendees = n3;

    public Conference:  seq of char*set of Speaker ==> Conference
    Conference(n1,n2) == (title := n1;speakers := n2; return self)
    post title = n1 and speakers = n2;

    --Obtém participantes de uma dada conferência
    public pure getLAttendees:() ==> set of Attendee
    getLAttendees() == return attendees;

    --Obtém palestrantes de uma dada conferência
    public getSpeakers:() ==> set of Speaker
    getSpeakers() == return speakers;

     --Retorna número de participantes de uma conferência
    public pure getAttendees:() ==> int
    getAttendees() ==(
          dcl count: real := 0;

          for all attendee in set attendees
                do count := count + 1;

          return count;
    )
    post RESULT >= 0;


    public getTitle:() ==> seq of char
    getTitle() == return title;
```

```
    -- Adiciona um participante a uma conferência
    public addAttendee: Attendee ==> ()
    addAttendee(attendee) == (
  --  dcl numberofattendees:int := getAttendees;
    attendees := attendees union {attendee}
    )
    pre attendee not in set attendees
    post attendee in set attendees;


functions
-- TODO Define functiones here

traces
-- TODO Define Combinatorial Test Traces here
end Conference
```

3.4 Class News

```
class News
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
public title: seq of char;
public content: seq of char;
public timeOfPublish: Types`Date;

operations
-- TODO Define operations here
public News: seq of char * seq of char * Types`Date ==> News
News(n1,n2,n3) == (title := n1;content := n2; timeOfPublish := n3; return self)
post title = n1 and content = n2 and timeOfPublish = n3;

--Título da notícia
public getTitle: () ==> seq of char
getTitle() == return title;

functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end News
```

3.5 Class Schedule

```
class Schedule
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
-- Cada Evento/Schedule tem um dia de começo, uma hora, um palco e uma conferência
    private TimeOfStart: Types`Time;
    private DayOfStart: Types`Date;
    private stage: Stage;
    private conference: Conference;

operations
-- TODO Define operations here
    public Schedule: Types`Time * Types`Date * Stage * Conference ==> Schedule
    Schedule(n1,n2,n3,n4) == (TimeOfStart := n1; DayOfStart := n2; stage := n3; conference := n4; return self)
  --  pre n3 <> stage or (n3 = stage and DayOfStart = n2 and TimeOfStart <> n1)
    post TimeOfStart = n1 and DayOfStart = n2 and stage = n3 and conference = n4;

    public getDayOfStart: () ==>Types`Date
    getDayOfStart() == return DayOfStart;

    public getTimeOfStart: () ==>Types`Time
    getTimeOfStart() == return TimeOfStart;

    public pure getConference: () ==> Conference
    getConference() == return conference;

    public pure getStage: () ==> Stage
    getStage() == return stage;



functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Schedule
```

3.6 Class Showroom

```
class Showroom
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
-- Um showroom é basicamente um evento de mostra de startups, tendo um dia de início e outro de fim
    private startups: set of startup :={};
    private DayOfStart: Types`Date;
    private DayOfEnd: Types`Date;
operations
-- TODO Define operations here
    public Showroom:  Types`Date * Types`Date ==> Showroom
  Showroom(n2,n3) == (DayOfStart := n2; DayOfEnd := n3; return self)
  post DayOfStart = n2 and DayOfEnd = n3;

  public addStartupToShowroom: startup ==> ()
  addStartupToShowroom(n1) == startups := startups union {n1}
  pre n1 not in set startups;

  public getStartups: () ==> set of startup
  getStartups() == return startups;

  public getDayOfStart: () ==> Types`Date
  getDayOfStart() == return DayOfStart;

  public getStartupByName: seq of char ==> startup
  getStartupByName(n1) == (
  dcl temp :startup;
        for all companies in set startups do(
        if companies.getName() = n1
        then temp := companies;


        );

        return temp;
  );

    public getStartupByCountry: seq of char ==> set of startup
  getStartupByCountry(n1) == (
        dcl temp:set of startup:={};



         for all companies in set startups do(
         if companies.getCountryOfOrigin() = n1
         then temp:= temp union {companies}

         );
         return temp;
   );

functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Showroom
```

3.7 Class Speaker

```
class Speaker
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
    public name: seq of char;
operations
-- TODO Define operations here
-- Representa um speaker
    public Speaker: seq of char ==> Speaker
  Speaker(n1) == (name := n1; return self)
  post name = n1;

  public getName: () ==> seq of char
getName() == return name;
functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Speaker
```

3.8 Class Stage

```
class Stage
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
private name: seq of char;
private capacity: int;
--Representa um palco
inv capacity >= 0;

operations
-- TODO Define operations here
    public Stage: seq of char * int ==> Stage
    Stage(n1,n2) == (name := n1;capacity := n2;return self)
    post name = n1 and capacity = n2;

    public pure getCapacity: () ==> int
    getCapacity() == return capacity;
functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Stage
```

3.9 Class Types

```
class Types
types
      -- Define Tipos Novos a usar
  public Time :: hours : int minutes: int seconds: int

  inv tempo == tempo.minutes >= 0 and tempo.minutes < 60 and tempo.seconds >= 0 and tempo.seconds < 60 and
  tempo.hours >= 0;

  public Date :: day : int  month: int year: int
  inv datas == datas.year >= 0 and datas.month > 0 and datas.month <= 12 and datas.day > 0 and
   datas.day <= 31;

values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
operations
-- TODO Define operations here
functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Types
```

3.10 Class WebSummit

```
class WebSummit
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
    private jornal: set of News :={};
    private startups: set of startup :={};
    private Schedules: set of Schedule :={};
    private Showrooms: set of Showroom :={};
    public static websumit: WebSummit := new WebSummit();

operations

public addNews: News ==> ()
addNews(news) == jornal := jornal union {news}
pre news not in set jornal
post news in set jornal;

public addShowroom: Showroom ==> ()
addShowroom(show) == Showrooms := Showrooms union {show}
pre show not in set Showrooms
post show in set Showrooms;

public getNews: () ==> set of News
getNews() == return jornal;

public getStartups: () ==> set of startup
getStartups() == return startups;

public getSchedules:() ==> set of Schedule
getSchedules() == return Schedules;

public addStartup: startup ==> ()
addStartup(startup) == startups := startups union {startup}
pre startup not in set startups
post startup in set startups;

public getNewsByTitle: seq of char ==>News
getNewsByTitle(n1) ==(
-- Obtém notícias pelo título
        dcl temp:News;
        for all noticias in set jornal do
```

```
        (
            if noticias.getTitle() = n1
            then temp := noticias;

        );
        return temp;

);

public getTalksByName: seq of char ==>set of Schedule
getTalksByName(n1) ==(
-- Obtém Schedules nas quais o n1 é orador
        dcl temp:set of Schedule :={};
        for all eventos in set Schedules do
        (
            for all speaker in set eventos.getConference().getSpeakers() do
            (
                if n1 = speaker.getName()
                then temp := temp union {eventos}
            );


        );
        return temp;

);

public addSchedule: Schedule ==> ()
addSchedule(schedule) == (
-- Adiciona um evento ao horário geral, tendo em conta múltiplas resrtrições
      dcl exist:int := 0;

      -- Não pode haver 2 eventos no mesmo palco ao mesmo tempo
    for all schedulez in set Schedules do(
        if (schedulez.getStage() = schedule.getStage() and schedulez.getDayOfStart() = schedule.getDayOfStart() and sch
        then
        (IO`println("Cannot have two events at the same stage on the same time");
        exist := 1;)

    );

    -- Um participante não pode participar em 2 palestras ao mesmo tempo
for all schedulez in set Schedules do(
for all name in set schedulez.getConference().getLAttendees() do
```

```
(
            for all name2 in set schedule.getConference().getLAttendees() do
            (
 if schedule.getDayOfStart() = schedulez.getDayOfStart() and schedule.getTimeOfStart() = schedulez.getTimeOfStart() an
 and name2 = name
               then (IO`println("One Cannot Attend 2 events in the same time");
                exist := 1;)


            );
        );


    );
        -- 1 Speaker não pode dar 2 palestras ao mesmo tempo
for all schedulez in set Schedules do(
for all name in set schedulez.getConference().getSpeakers() do
(
for all name2 in set schedule.getConference().getSpeakers() do
(
if schedule.getDayOfStart() = schedulez.getDayOfStart() and schedule.getTimeOfStart() = schedulez.getTimeOfStart() and
and name2 = name
then (IO`println("One Cannot Attend 2 events in the same time");
exist := 1;)


);
);


    );
    if exist = 0 then
    Schedules := Schedules union {schedule}

);

-- Retorna as conferências nesse tempo
public getConferenceAt: Types`Date * Types`Time ==> set of Schedule
getConferenceAt(n1,n2) == (
             dcl events: set of Schedule :={};

         for all schedule in set Schedules do
            (
            if schedule.getDayOfStart() = n1 and schedule.getTimeOfStart() = n2
```

```
            then events := events union {schedule}
              );

              return events;

);
-- Retorna a Schedule com uma conferência com o título especificado
public getConferenceWithTitle: seq of char ==> Schedule
getConferenceWithTitle(n1) == (
            dcl event: Schedule;

          for all schedule in set Schedules do
            (
            if schedule.getConference().getTitle() = n1
            then event := schedule
              );

              return event;
);

public getShowroomByDate: Types`Date ==> Showroom
getShowroomByDate(n1) == (
            dcl show2: Showroom;

          for all show in set Showrooms do
            (
            if show.getDayOfStart() = n1
            then show2 := show
              );

              return show2;
);

-- Adiciona um participante dado um Schedule, que possui um Evento
public addAttendeetoConference: Attendee * Schedule ==> ()
addAttendeetoConference(n1,n2) ==(

       dcl exist:int := 0;

  for all schedulez in set Schedules do
  (

  if (schedulez.getStage() <> n2.getStage() and schedulez.getDayOfStart() = n2.getDayOfStart() and schedulez.getTimeOf
  then


    (
    IO`print("Cannot add Attendee that already is signed up on another event");
    exist := 1;
    );

          );

          if exist = 0
          then n2.getConference().addAttendee(n1);


 )
 pre n1 not in set n2.getConference().getLAttendees() -- To add a new Attendee to a Conference we need to make sure he
 post  n2.getConference().getAttendees() + 1 < n2.getStage().getCapacity();  -- To add a new attendee we also need to


   functions
 -- TODO Define functiones here
 traces
 -- TODO Define Combinatorial Test Traces here
 end WebSummit
```

4 Model validation

4.1 Class Test

```
class Test
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
operations

  protected assert : bool ==> ()
  assert(a) == return
  pre a

functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end Test
```

4.2 Class MainTest

```
class MainTest is subclass of Test
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here

operations
-- TODO Define operations here
public test() == (
    IO`print("testAddNews -> ");
    testAddNews();
    IO`println("Success");

    IO`print("testAddStartup -> ");
    testAddStartup();
    IO`println("Success");

     IO`print("testAddEvent -> ");
    testgetAttendees();
    IO`println("Success");

    IO`print("testAddSchedule -> ");
    testAddSchedule();
    IO`println("Success");

     IO`print("testgetConferencesAt -> ");
    testgetConferenceAt();
    IO`println("Success");

    IO`print("testgetTitle -> ");
    testgetTitle();
    IO`println("Success");

    IO`print("testgetEventbyTitle -> ");
    testgetTitle();
    IO`println("Success");

    IO`print("testcreateShowroom -> ");
    testaddShowroom();
    IO`println("Success");

    IO`print("testgetEventbyTitle -> ");
```

Requisito	Teste que o comprova
R01	testAddSchedule()
R02	testaddShowroom()
R03	testAddNews()
R04	testgetAttendees()
R05	testAddSchedule()
R06	testAddSchedule()
R07	testaddShowroom()
R08	testAddSchedule()
R09	testgetConferenceAt()
R10	testaddAttendee()
R11	testgetEventbyTitle()
R12	testAddSchedule()
R13	Não está especificado um teste, mas a função existe
R14	testaddShowroom()
R15	testAddNews()
R16	testgetTalks()

```
  testgetEventbyTitle();
  IO`println("Success");

  IO`print("testaddAttendee -> ");
  testaddAttendee();
  IO`println("Success");

    IO`print("testSchedules -> ");
  testschedule();
  IO`println("Success");

  IO`print("testgetTalks-> ");
  testgetTalks();
  IO`println("Success");



);

public testAddNews() == (
-- Testa se consegue adicionar uma notícia
  dcl summit1: WebSummit := new WebSummit();
  dcl n2:News;
  dcl n1: News := new News("Inscriptions for WebSummit Open","Inscriptions are open from 20th February to 20th March",
  summit1.addNews(n1);
  n2 := summit1.getNewsByTitle("Inscriptions for WebSummit Open");
  assert(n1 in set summit1.getNews());
  assert(n1 = n2);

);

public testgetTalks() ==(

  dcl summit1: WebSummit := new WebSummit();
  dcl s1:Speaker := new Speaker("Maria Laurinda");
  dcl s2:Speaker := new Speaker("Elon Musk");
  dcl s3:Speaker := new Speaker("yuyuyu");
  dcl a1:Attendee := new Attendee("Bill Gates");
  dcl a2:Attendee := new Attendee("Miguel");


--  dcl c1:Conference := new Conference(mk_Types`Time(10,30,00),{s1,s2},{a1,a2},mk_Types`Date(21,5,2010));
  dcl c1:Conference := new Conference("Reneweable energies",{s1},{a1,a2});
  dcl c3:Conference := new Conference("Reneweable pol",{s1});
```

```
  dcl stage1:Stage := new Stage("Main Stage",100);
  dcl stage2:Stage := new Stage("Main Stage2",100);
  dcl schedule1:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c1);
  dcl schedule2:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage2,c3);
  dcl temp: set of Schedule;

  summit1.addSchedule(schedule1);

  temp := summit1.getTalksByName("Maria Laurinda");
  assert(schedule1 in set temp);


);

public testAddStartup() ==(
  dcl summit1: WebSummit := new WebSummit();
  dcl s1:startup := new startup("OsCorp","USA");
  summit1.addStartup(s1);
assert(s1 in set summit1.getStartups());

);

public testAddSchedule() ==(
--Testa se consegue adicionar um evento ao horário geral de atividades
  dcl summit1: WebSummit := new WebSummit();
  dcl s1:Speaker := new Speaker("Maria Laurinda");
  dcl s2:Speaker := new Speaker("Elon Musk");
  dcl s3:Speaker := new Speaker("yuyuyu");
  dcl a1:Attendee := new Attendee("Bill Gates");
  dcl a2:Attendee := new Attendee("Miguel");


--  dcl c1:Conference := new Conference(mk_Types`Time(10,30,00),{s1,s2},{a1,a2},mk_Types`Date(21,5,2010));
  dcl c1:Conference := new Conference("Reneweable energies",{s1},{a1,a2});
  dcl c3:Conference := new Conference("Reneweable pol",{s1});
  dcl stage1:Stage := new Stage("Main Stage",100);
  dcl stage2:Stage := new Stage("Main Stage2",100);
  dcl schedule1:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c1);
  dcl schedule2:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage2,c3);
  dcl num:int := c1.getAttendees();

  summit1.addSchedule(schedule1);
  assert(schedule1 in set summit1.getSchedules());
```

```
  assert(2 = num);
--  summit1.addSchedule(schedule2); -- Testa se não é possível ter 2 eventos com o mesmo orador em lugares
                                    -- distintos. Tem de falhar.
  --IO`println(summit1.getSchedules());

  --summit1.addConference(c1);
  assert(schedule2 not in set summit1.getSchedules());


);

  public testgetAttendees() ==(
  --Testa se consegue consegue obter o número de participantes corretamente

  dcl s1:Speaker := new Speaker("Maria Laurinda");
  dcl s2:Speaker := new Speaker("Elon Musk");
  dcl a1:Attendee := new Attendee("Bill Gates");
  dcl a2:Attendee := new Attendee("Miguel");

  dcl c1:Conference := new Conference("Renewable energies",{s1,s2},{a1,a2});
  dcl num:int := c1.getAttendees();
  dcl t1:seq of char;
  t1 := c1.getTitle();

  assert(2 = num);

);

public testgetConferenceAt() ==(

--Devolve as conferências disponíveis a um determinado tempo

  dcl summit1: WebSummit := new WebSummit();
  dcl s1:Speaker := new Speaker("Maria Laurinda");
  dcl s2:Speaker := new Speaker("Elon Musk");
  dcl a1:Attendee := new Attendee("Bill Gates");
  dcl a2:Attendee := new Attendee("Miguel");

  dcl c1:Conference := new Conference("Renewable energies",{s1,s2},{a1,a2});
  dcl stage1:Stage := new Stage("Main Stage",100);
  dcl schedule1:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c1);
  dcl schedules: set of Schedule;

  summit1.addSchedule(schedule1);
```

```
  schedules := summit1.getConferenceAt(mk_Types`Date(21,5,2010),mk_Types`Time(10,30,00));
  assert(schedule1 in set schedules);


);

  public testgetTitle() ==(
  --Testa se a função get funciona
  dcl s1:Speaker := new Speaker("Maria Laurinda");
  dcl s2:Speaker := new Speaker("Elon Musk");
  dcl a1:Attendee := new Attendee("Bill Gates");
  dcl a2:Attendee := new Attendee("Miguel");
  dcl t1:seq of char;
  dcl c1:Conference := new Conference("Renewable energies",{s1,s2},{a1,a2});
  t1 := c1.getTitle();
  assert(t1 = "Renewable energies");


);

public testgetEventbyTitle() ==(
-- Testa se consegue obter uma conferência pelo título
dcl summit1: WebSummit := new WebSummit();
  dcl s1:Speaker := new Speaker("Maria Laurinda");
  dcl s2:Speaker := new Speaker("Elon Musk");
  dcl a1:Attendee := new Attendee("Bill Gates");
  dcl a2:Attendee := new Attendee("Miguel");

  dcl c1:Conference := new Conference("Renewable energies",{s1,s2},{a1,a2});
  dcl stage1:Stage := new Stage("Main Stage",100);
  dcl schedule1:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c1);
  dcl schedules: Schedule;

  summit1.addSchedule(schedule1);

  schedules := summit1.getConferenceWithTitle("Renewable energies");
  assert(schedules.getConference().getTitle() = "Renewable energies");


);

public testaddAttendee() ==(
-- Adiciona um Participante a uma conferência
-- Testa ainda se pode adicionar o mesmo participante a uma conferência a decorrer ao mesmo tempo
```

```
-- noutro local

dcl summit1: WebSummit := new WebSummit();
    dcl stage1:Stage := new Stage("Main Stage",100);
    dcl stage2:Stage := new Stage("Main Stage2",100);
    dcl a1:Attendee := new Attendee("Bill Gatorade");
    dcl c1:Conference := new Conference("Renewable energies",{},{});
    dcl c2:Conference := new Conference("Renewable gasolines",{},{});
    dcl schedule1:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c1);
    dcl schedule2:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage2,c2);

  summit1.addSchedule(schedule1);
  summit1.addSchedule(schedule2);
  summit1.addAttendeetoConference(a1,schedule1);
  summit1.addAttendeetoConference(a1,schedule2);



);

public testschedule() ==(
-- Testa todos os testes na Schedule, até os para falhar daí não assuster por aparecerem
-- erros na consola

dcl summit1: WebSummit := new WebSummit();
    dcl stage1:Stage := new Stage("Main Stage",100);
    dcl stage2:Stage := new Stage("Main Stage2",100);
    dcl a1:Attendee := new Attendee("Bill Gatorade");
    dcl s1:Speaker := new Speaker("Elon Musk");
    dcl c1:Conference := new Conference("Renewable energies",{},{a1});
    dcl c2:Conference := new Conference("Renewable gasolines",{},{a1});
    dcl c3:Conference := new Conference("Renewable energies",{s1},{});
    dcl c4:Conference := new Conference("Renewable gasolines",{s1},{});
    dcl schedule1:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c1);
    dcl schedule2:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage2,c2);
    dcl schedule3:Schedule := new Schedule(mk_Types`Time(10,30,00),mk_Types`Date(21,5,2010),stage1,c3);

    dcl schedule5:Schedule := new Schedule(mk_Types`Time(11,30,00),mk_Types`Date(21,5,2010),stage1,c3);
    dcl schedule4:Schedule := new Schedule(mk_Types`Time(11,30,00),mk_Types`Date(21,5,2010),stage2,c4);


  -- Tenta implementar várias opções erradas, para verificar que as condições estão bem

  summit1.addSchedule(schedule1);
```

```
    summit1.addSchedule(schedule2);
    summit1.addSchedule(schedule3);
    summit1.addSchedule(schedule5);
    summit1.addSchedule(schedule4);



  );

  public testaddShowroom() ==(
  dcl summit1: WebSummit := new WebSummit();
  dcl startup2:startup := new startup("Tesla","USA");

  dcl showroom1:Showroom := new Showroom(mk_Types`Date(20,5,2010),mk_Types`Date(21,5,2010));
  dcl list:set of startup;
  dcl tempstartup:startup;
  dcl tempstartup1:set of startup;

  showroom1.addStartupToShowroom(startup2);
  tempstartup:=showroom1.getStartupByName("Tesla");
  tempstartup1:=showroom1.getStartupByCountry("USA");
  assert(startup2 in set tempstartup1);

  assert(tempstartup = startup2);
  list := showroom1.getStartups();
  assert(startup2 in set list);
  summit1.addShowroom(showroom1);
  assert(summit1.getShowroomByDate(mk_Types`Date(20,5,2010)) = showroom1);

  );
functions
-- TODO Define functiones here
--
traces
-- TODO Define Combinatorial Test Traces here
end MainTest
```

4.3 Class WebSummitTest

```
class WebSummitTest
types
-- TODO Define types here
values
-- TODO Define values here
instance variables
-- TODO Define instance variables here
operations
-- TODO Define operations here
-- Chama os testes a executar
    public static main() == (
    new MainTest().test();
  );
functions
-- TODO Define functiones here
traces
-- TODO Define Combinatorial Test Traces here
end WebSummitTest
```

5 Verificação do modelo

5.1 Exemplo de verificação de domínio

Um exemplo, no nosso caso, de uma *proof obligation* poderá ser:

nº	PO Name	Type
5	Conference'addAttendee(Attendee)	operation establishes postcondition

O código em causa é o seguinte:

```
-- Adiciona um participante a uma conferencia
public addAttendee: Attendee ==> ()
addAttendee(attendee) == (
-- dcl numberofattendees:int := getAttendees;
attendees := attendees union {attendee}
)
pre attendee not in set attendees
post attendee in set attendees;
```

Vista *Proof Obligation*:

```
(forall attendee:Attendee & ((attendee not in set attendees) =>
(attendee in set (attendees union {attendee}))))
```

Neste caso, e porque no nosso sistema, não queremos elementos repetidos, partindo deste princípio, a única maneira de unirmos 2 elementos e garantirmos que não há elementos repetidos, é assumindo que o elemento que vamos juntar não estava colocado já anteriormente no set, estando assim provada a pré-condição.

5.2 Exemplo de verificação de invariante

Um exemplo, no nosso caso, de uma *proof obligation* poderá ser:

nº	PO Name	Type
43	Stage'Stage(seq of (char),int)	state invariant holds

O código em causa é o seguinte:

```
public Stage: seq of char * int ==> Stage
Stage(n1,n2) == (name := n1;capacity := n2;return self)
post name = n1 and capacity = n2;
```

sendo que o invariante = *inv capacity >= 0;* Neste caso, no nosso sistema, não é possível ter palcos com capacidade inferior a 0. Como tal, e partindo desta perspetiva, que o nosso sistema só será possível se verificar esta condição, o código respeita o invariante. Caso o nosso sistema quisesse palcos com capacidade negativa, aí o invariante já não se mantinha, o que causaria um falha no sistema. Contudo, como não é o caso, o invariante mantém-se.

6 Geração de código JAVA

A geração de código JAVA a partir de código VDM++ é bastante simples, bastante apenas clicar no projeto e gerar. Contudo, no meu caso, houveram certos problemas a converter os testes. Sendo assim, e por assumir que os testes não fariam mais falta, comentei esta parte e escrevi uma interface mais amigável, que desse para perceber que o trabalho havia tido sucesso. O código JAVA está em anexo, pelo que é só correr e verificar assim da implementação do modelo pedido.

```
package MFES;

import java.util.*;
import org.overture.codegen.runtime.*;

public class WebSummitTest {

    public static void main(String[] args) {
    WebSummit web = new WebSummit();
    int n=0;
    while(n !=  10) {
    System.out.println("======================");
    System.out.println("=Welcome to WebSummit=");
    System.out.println("======================");
    System.out.println("1. Create News");
    System.out.println("2. Check Journal");
    System.out.println("3. Create Showroom");
    System.out.println("4. Add Startup to Showroom");
    System.out.println("5. Add New Event to the Schedule");
    System.out.println("6. Add Attendee to a Conference");
    System.out.println("7. Check Schedule at a specific time");
    System.out.println("8. Get Number of Attendees of a Schedule");
    System.out.println("9. Get Talks given by Person X");
    System.out.println("10. Exit");
    Scanner reader = new Scanner(System.in);
    n = reader.nextInt();

    Scanner reader2,reader3,reader4,reader5,reader6,reader7,reader8,reader9,reader10,reader11;

    if(n == 1) {
        System.out.println("Insert Title of News");
        reader2 = new Scanner(System.in);
        String n2 = reader2.nextLine();
        System.out.println("Insert Content of News");
        reader3 = new Scanner(System.in);
        String n3 = reader3.nextLine();
        System.out.println("Insert Day");
        reader4 = new Scanner(System.in);
        int n4 = reader4.nextInt();
        System.out.println("Insert Month");
        reader5 = new Scanner(System.in);
        int n5 = reader5.nextInt();
        System.out.println("Insert Year");
        reader6 = new Scanner(System.in);
```

```
    int n6 = reader6.nextInt();
    Types.Date temp = new Types.Date(n4,n5,n6);
    News news1 = new News(n2,n3,temp);
    web.addNews(news1);
    System.out.println("News inserted");
}
if(n == 2) {
    System.out.println(web.getNews());
}
if(n == 3) {
    System.out.println("Insert Day of Start");
    reader4 = new Scanner(System.in);
    int n4 = reader4.nextInt();
    System.out.println("Insert Month of Start");
    reader5 = new Scanner(System.in);
    int n5 = reader5.nextInt();
    System.out.println("Insert Year of Start");
    reader6 = new Scanner(System.in);
    int n6 = reader6.nextInt();
    System.out.println("Insert Day of End");
    reader7 = new Scanner(System.in);
    int n7 = reader4.nextInt();
    System.out.println("Insert Month of End");
    reader8 = new Scanner(System.in);
    int n8 = reader5.nextInt();
    System.out.println("Insert Year of End");
    reader9 = new Scanner(System.in);
    int n9 = reader6.nextInt();
    Types.Date temp = new Types.Date(n4,n5,n6);
    Types.Date temp2 = new Types.Date(n7,n8,n9);

    Showroom show = new Showroom(temp,temp2);
    web.addShowroom(show);

}
if(n == 4) {
    System.out.println("Insert Day of Start");
    reader4 = new Scanner(System.in);
    int n4 = reader4.nextInt();
    System.out.println("Insert Month of Start");
    reader5 = new Scanner(System.in);
    int n5 = reader5.nextInt();
    System.out.println("Insert Year of Start");
    reader6 = new Scanner(System.in);
```

```
    int n6 = reader6.nextInt();
    Types.Date temp = new Types.Date(n4,n5,n6);

    System.out.println("Insert Name of Startup");
    reader7 = new Scanner(System.in);
    String n7 = reader7.nextLine();
    System.out.println("Insert Country of Origin of Startup");
    reader8 = new Scanner(System.in);
    String n8 = reader8.nextLine();
    startup star = new startup(n7,n8);
    web.getShowroomByDate(temp).addStartupToShowroom(star);
    System.out.println(web.getShowroomByDate(temp));

}
if(n == 5) {

    System.out.println("Insert Speaker");
    reader2 = new Scanner(System.in);
    String n2 = reader2.nextLine();
    System.out.println("Insert Title of Conference");
    reader3 = new Scanner(System.in);
    String n3 = reader3.nextLine();
    Speaker speak = new Speaker(n2);
    VDMSet speakers = SetUtil.set();
    speakers.add(speak);
    Conference conf = new Conference(n3,speakers);
    System.out.println("Insert Day of Start");
    reader4 = new Scanner(System.in);
    int n4 = reader4.nextInt();
    System.out.println("Insert Month of Start");
    reader5 = new Scanner(System.in);
    int n5 = reader5.nextInt();
    System.out.println("Insert Year of Start");
    reader6 = new Scanner(System.in);
    int n6 = reader6.nextInt();
    System.out.println("Insert Hour of Start");
    reader7 = new Scanner(System.in);
    int n7 = reader4.nextInt();
    System.out.println("Insert Minute of Start");
    reader8 = new Scanner(System.in);
    int n8 = reader5.nextInt();
    System.out.println("Insert Second of Start");
    reader9 = new Scanner(System.in);
    int n9 = reader6.nextInt();
```

```
        System.out.println("Insert Stage Name");
        reader10 = new Scanner(System.in);
        String n10 = reader7.nextLine();
        System.out.println("Insert Capacity of Stage");
        reader11 = new Scanner(System.in);
        int n11 = reader6.nextInt();

        Stage stage = new Stage(n10,n11);

        Types.Date temp = new Types.Date(n4,n5,n6);
        Types.Time temp2 = new Types.Time(n7,n8,n9);

        Schedule schedule = new Schedule(temp2,temp,stage,conf);

        web.addSchedule(schedule);
        System.out.println(web.getSchedules());



    }

    if(n == 6) {

        System.out.println("Insert Conference Title");
        reader4 = new Scanner(System.in);
        String n4 = reader4.nextLine();
        Schedule c1 = web.getConferenceWithTitle(n4);
        System.out.println("Insert new Attendee Name");
        reader5 = new Scanner(System.in);
        String n5 = reader5.nextLine();
        Attendee n1 = new Attendee(n5);
        web.addAttendeetoConference(n1,c1);
        System.out.println(web.getSchedules());



    }
    if(n == 7) {

        System.out.println("Insert Day of Start");
        reader4 = new Scanner(System.in);
        int n4 = reader4.nextInt();
        System.out.println("Insert Month of Start");
```

```
        reader5 = new Scanner(System.in);
        int n5 = reader5.nextInt();
        System.out.println("Insert Year of Start");
        reader6 = new Scanner(System.in);
        int n6 = reader6.nextInt();
        System.out.println("Insert Hour of Start");
        reader7 = new Scanner(System.in);
        int n7 = reader4.nextInt();
        System.out.println("Insert Minute of Start");
        reader8 = new Scanner(System.in);
        int n8 = reader5.nextInt();
        System.out.println("Insert Second of Start");
        reader9 = new Scanner(System.in);
        int n9 = reader6.nextInt();
        Types.Date temp = new Types.Date(n4,n5,n6);
        Types.Time temp2 = new Types.Time(n7,n8,n9);
        System.out.println(web.getConferenceAt(temp, temp2));


    }
    if(n == 8) {
        System.out.println("Insert Conference Title");
        reader4 = new Scanner(System.in);
        String n4 = reader4.nextLine();
        Schedule c1 = web.getConferenceWithTitle(n4);
        System.out.println(c1.getConference().getAttendees());


    }
    if(n == 9) {
        System.out.println("Insert Speaker Name");
        reader4 = new Scanner(System.in);
        String n4 = reader4.nextLine();
        VDMSet c1 = web.getTalksByName(n4);
        System.out.println(c1);


    }


    }
```

7 Conclusões

É, preciso pois concluir o projeto bem como afirmar umas palavras de relevo relativas ao conteúdo produzido. Creio que em geral o projeto foi bem conseguido, havendo ainda espaço para melhoria no modelo, aumentando a complexidade, por exemplo criando filas de espera para os eventos, enfim uma grande variedade de coisas que poderíamos incorporar. É também, com o meu pesar que afirmo que o trabalho elaborado neste projeto foi 100% por mim, Joel Dinis. De resto, é assim concluído o projeto.

8 Referências

https://www.google.pt/url?sa=t&rct=j&q=&esrc=s&source=web&cd=8&cad=rja&uact=8&ved=0ahUKEwiV8aux7LDYAhXK
https3A%2F%2Fraw.githubusercontent.com%2Foverturetool%2Fdocumentation%2Fmaster%2Fdocumentation%
2FUserGuideOvertureIDE%2FOvertureIDEUserGuide.pdf&usg=AOvVaw3-q-R5WZD30jLjeEGOF6fv https://moodle
.up.pt/pluginfile.php/165034/mod_resource/content/0/MFES-TrabVDM-1718.pdf