**UFOP (Universidade Federal de Ouro Preto)**

****

**PROFESSOR:** Tiago Garcia de Senna Carneiro

**RELATÓRIO - BCC 322:**

**Gabriel Catizani Faria Oliveira (20.1.4004)**

**Trabalho Prático 2 – Sprint 1**

Construção de Simulações Baseadas na Dinâmica de Sistemas

**Ouro Preto, Minas Gerais**

**14 de dezembro de 2021**

***PARTE 1 – Casos de uso***

**Caso 1: Apenas um estoque**



*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* est(*value*);

m.add(&est);

OU

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* est();

est.SetInitialValue(value);

m.add(&est);

**Caso 2: Apenas um Flow**

****

**OBS: os nulls seriam de onde o Flow sai e aonde ele esta entrando, respectivamente**

**Exemplo: Flow flu(origem, destino);**

*Model* m(*inicialTime*, *finalTime*, *id*);

*Flow* flu(*stock* \**ori* = null, *stock* \**des* = null);

m.add(&flu, null, null);

OU

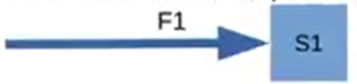
*Model* m(*inicialTime*, *finalTime*, *id*);

*Flow* flu();

flu.connection(stock \*ori = null, stock \*des = null);

m.add(&flu, null, null);

**Caso 3: Existe apenas um Flow e tem apenas uma entrada. A saida não existe.**

****

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* destiny(*value*);

*Flow* flu(*stock* \**ori* = null, &*destiny*);

m.add(&flu, null, &destiny);

OU

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* destiny();

destiny.SetInitialValue(value);

*Flow* flu();

flu.connection(stock \*ori = null, &destiny);

m.add(&flu, null, &destiny);

**Caso 4: Um fluxo conectado à apenas um estoque na origem**

****

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* source(*value*);

*Flow* flu(&*source*, *stock* \**des* = null);

m.add(&flu, &source, null);

OU

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* source();

source.SetInitialValue(value);

*Flow* flu();

flu.connection(&source, stock \*des = null);

m.add(&flu, &source, null);

**Caso 5: Um fluxo conectando dois estoques**

****

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* source(*value1*);

*Stock* destiny(*value2*);

*Flow* flu(*source*, *destiny*);

m.add(&flu, &source, &destiny);

OU

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* source();

*Stock* destiny();

source.SetInitialValue(value1);

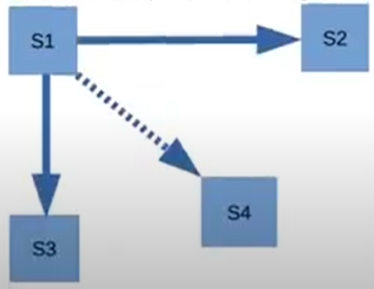
destiny.SetInitialValue(value2);

*Flow* flu();

Flu.connection(&source, &destiny);

m.add(&flu, &source, &destiny);

**Caso 6: Um estoque que serve de origem a outros dois ou mais sistemas de destino**

****

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* source(*value1*);

*Stock* destinyA(*value2*);

*Stock* destinyB(*value3*);

*Stock* destinyC(*value4*);

*Flow* flu1(&*source*, &*destinyA*);

*Flow* flu2(&*source*, &*destinyB*);

*Flow* flu3(&*source*, &*destinyC*);

m.add(&flu1, &source, &destinyA);

m.add(&flu2, &source, &destinyB);

m.add(&flu3, &source, &destinyC);

OU

*Model* m(*inicialTime*, *finalTime*, *id*)

*Stock* source();

*Stock* destinyA(), destinyB(), destinyC();

source.SetInitialValue(value1);

destinyA.SetInitialValue(value2);

destinyB.SetInitialValue(value3);

destinyC.SetInitialValue(value4);

*Flow* flu1(), flu2(), flu3();

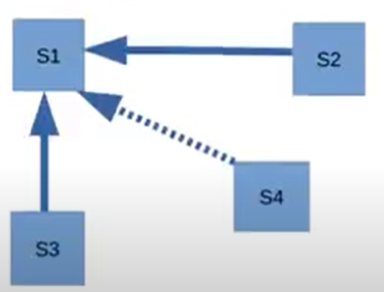
flu1.connection(&source, &destinyA), flu2.connection(&source, &destinyB), flu3.connection(&source, &destinyC);

m.add(&flu1, &source, &destinyA);

m.add(&flu2, &source, &destinyB);

m.add(&flu3, &source, &destinyC);

**Caso 7: um estoque que serve de destino para dois outros sistemas ou mais**

****

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* destiny(*value1*);

*Stock* sourceA(*value2*);

*Stock* sourceB(*value3*);

*Stock* sourceC(*value4*);

*Flow* flu1(&*sourceA*, &*destiny*);

*Flow* flu2(&*sourceB*, &*destiny*);

*Flow* flu3(&*sourceC*, &*destiny*);

m.add(&flu1, &sourceA, &destiny);

m.add(&flu2, &sourceB, &destiny);

m.add(&flu3, &sourceC, &destiny);

OU

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* destiny();

*Stock* sourceA(), sourceB(), sourceC();

destiny.SetInitialValue(value1);

sourceA.SetInitialValue(value2);

sourceB.SetInitialValue(value3);

sourceC.SetInitialValue(value4);

*Flow* flu1(), flu2(), flu3();

flu1.connection(&sourceA, &destiny), flu2.connection(&sourceB, &destiny), flu3.connection(&sourceC, &destiny);

m.add(&flu1, &sourceA, &destiny);

m.add(&flu2, &sourceB, &destiny);

m.add(&flu3, &sourceC, &destiny);

**Caso 8: Um estoque "ponte" que serve de origem para um determinado fluxo e destino para outro(s)**

****

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* DesOri(*value*);

*Flow* flu1(*stock* \**ori* = null, &*DesOri*);

*Flow* flu2(&*DesOri*, *stock* \**des* = null);

m.add(&flu1, null, &DesOri)

m.add(&flu2, &DesOri, null);

**OU**

*Model* m(*inicialTime*, *finalTime*, *id*);

*Stock* DesOri();

DesOri.SetInitialValue(value);

*Flow* flu1();

*Flow* flu2();

flu1.connection(stock \*ori = null, &DesOri);

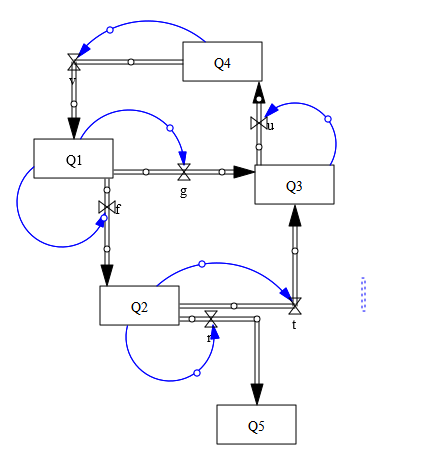
flu2.connection(&DesOri, stock \*des = null);

m.add(&flu1, null, &DesOri)

m.add(&flu2, &DesOri, null);

***PARTE 2 – Criterio de Aceitação***

**1º critério:**

****

double value1, value2, value3, value4, value5;

double inicialTime, finalTime;

*model* m (*inicialTime*, *finalTime*, 1);

*Stock* Q1(*value1*, "Q1");

*Stock* Q2(*value2*, "Q2");

*Stock* Q3(*value3*, "Q3");

*Stock* Q4(*value4*, "Q4");

*Stock* Q5(*value5*, "Q5");

*Flow* fluF (&*Q1*, &*Q2*, 1);

*Flow* fluG (&*Q1*, &*Q3*, 2);

*Flow* fluR (&*Q2*, &*Q5*, 3);

*Flow* fluT (&*Q2*, &*Q3*, 4);

*Flow* fluU (&*Q3*, &*Q4*, 5);

*Flow* fluV (&*Q4*, &*Q1*, 6);

m.add (&fluF, &Q1, &Q2);

m.add (&fluG, &Q1, &Q3);

m.add (&fluR, &Q2, &Q5);

m.add (&fluT, &Q2, &Q3);

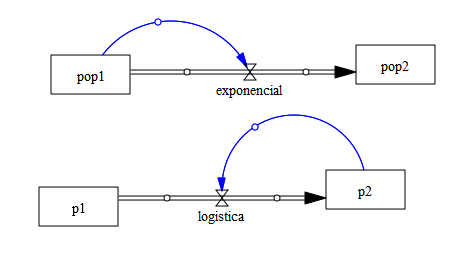
m.add (&fluU, &Q3, &Q4);

m.add (&fluV, &Q4, &Q1);

m.run();

m.print\_results();

**2º Critério:**

****

double value1, value2;

double inicialTime, finalTime;

*Model* m (*inicialTime*, *finalTime*, 2);

*Stock* pop1(*value1*, "pop1");

*Stock* pop2(*value2*, "pop2");

*Flow* expo (&*pop1*, &*pop2*);

m.add(&expo, &pop1, &pop2);

m.run()

m.print\_results();

**OU**

double value1, value2;

double inicialTime, finalTime;

*Model* m (*inicialTime*, *finalTime*, 2);

*Stock* p1(*value1*, "p1");

*Stock* p2(*value2*, "p2");

*Flow* logi (&*p1*, &*p2*);

m.add(&logi, &p1, &p2);

m.run();

m.print\_results();

***PARTE 3 – UML***

