**UNIVERSIDADE FEDERAL DE OURO PRETO – UFOP**

**Ciência da Computação**

****

**ENGENHARIA DE SOFTWARE**

**SPRINT 1 - Projeto da API do simulador de sistemas dinâmicos**

Marcus Vinícius Souza Fernandes

19.1.4046

**Ouro Preto**

**2021**

**Observação**

Esse relatório foi realizado discutindo o assunto com outros integrantes da classe para facilitar o entendimento do problema proposto, portanto o resultado obtido foi o mesmo.

**Funcionalidades**

Essa API é capaz de executar sistemas dinâmicos baseados nos preceitos da Teoria Geral de Sistemas a partir da criação de sistemas (que podem representar um estoque de energia, objetos, etc.) e fluxos (que associam os sistemas) que se relacionam através de um modelo. É possível associar quaisquer fórmulas matemáticas para os fluxos e observar os resultados após a execução de diversos sistemas dinâmicos.

**Casos de Uso**

Antes de começar a projetar a estrutura da API, é necessário realizar os estudos de caso e analisar como ela deveria se comportar em diferentes situações. Para isso foram pensadas algumas situações, nove no total, que estão listadas abaixo.

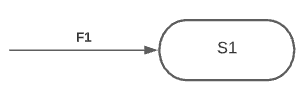
**Caso 1:** Fluxo “isolado”, que não apresenta entrada nem saída.



**Caso 2:** Sistema “isolado”, que não apresenta fluxos em sua entrada ou saída.



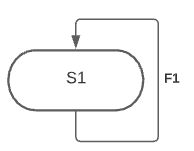
**Caso 3:** Sistema que apresenta apenas um fluxo em sua entrada.



**Caso 4:** Sistema que apresenta apenas um fluxo em sua saída.



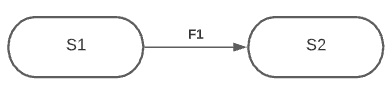
**Caso 5:** Sistema “cíclico” que apresenta o mesmo fluxo tanto em sua entrada quanto em sua saída.



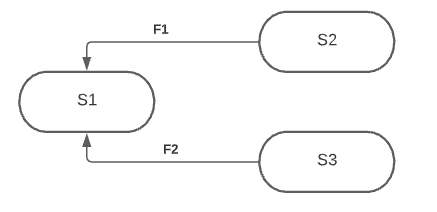
**Caso 6:** Sistema que apresenta fluxo de entrada e saída.



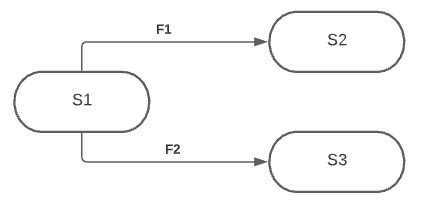
**Caso 7:** Fluxo que apresenta sistemas em sua entrada e saída.



**Caso 8:** Sistema que é conectado a dois ou mais sistemas por fluxos de entrada.



**Caso 9:** Sistema que é conectado a dois ou mais sistemas por fluxos de saída.

****

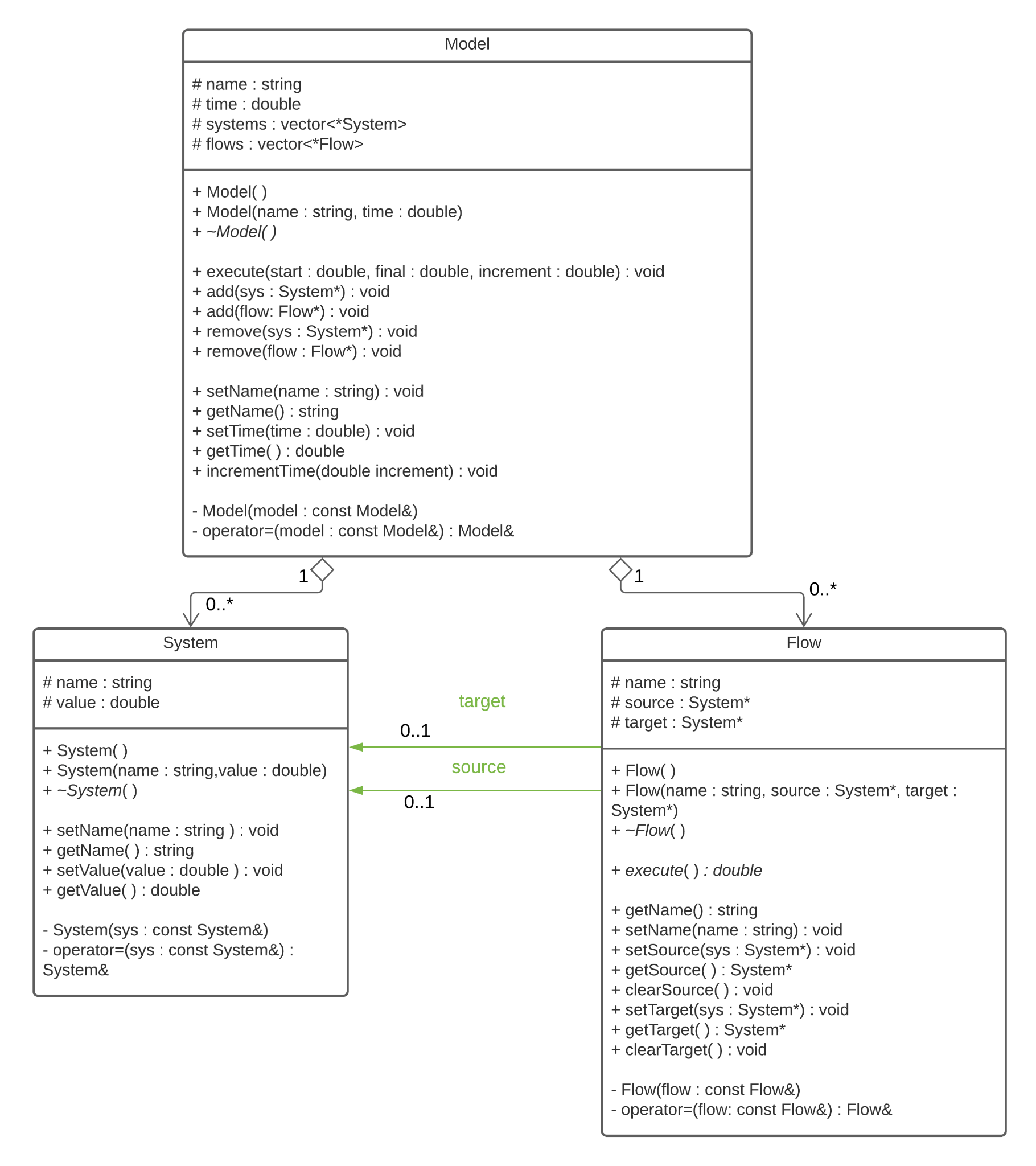
**Codificação dos casos de uso**

Nessa parte é pensado como seria o uso da API através de códigos escritos pelo usuário. Para cada um dos casos de teste apresentados anteriormente foi imaginado como seria a implementação deles em código.

|  |
| --- |
| **// Teste referente ao caso de uso 1 Model m; Flow f1("F1", NULL, NULL); m.add(f1); m.execute();  // Teste referente ao caso de uso 2 Model m; System s1("S1", 0); m.add(s1); m.execute();  // Teste referente ao caso de uso 3 Model m; System s1("S1", 0); Flow f1("F1", NULL, &f1); m.add(s1); m.add(f1); m.execute();  // Teste referente ao caso de uso 4 Model m; System s1("S1", 0); Flow f1("F1", &s1, NULL); m.add(s1); m.add(f1); m.execute();  // Teste referente ao caso de uso 5 Model m; System s1("S1", 0); Flow f1("F1", &s1, &s1); m.add(s1); m.add(f1); m.execute();  // Teste referente ao caso de uso 6 Model m; System s1("S1", 0); Flow f1("F1", NULL, &s1); Flow f2("F2", &s1, NULL); m.add(s1); m.add(f1); m.add(f2); m.execute();  // Teste referente ao caso de uso 7 Model m; System s1("S1", 0); System s2("S2", 0); Flow f1("F1", &s1, &s2); m.add(s1); m.add(s2); m.add(f1); m.execute();  // Teste referente ao caso de uso 8 Model m; System s1("S1",0); System s2("S2",0); System s3("S3",0); Flow f1("F1",&s2,&s1); Flow f2("F2",&s3,&s1); m.add(s1); m.add(s2); m.add(s3); m.add(f1); m.add(f2); m.execute();  // Teste referente ao caso de uso 9 Model m; System s1("S1",0); System s2("S2",0); System s3("S3",0); Flow f1("F1",&s1,&s2); Flow f2("F2",&s1,&s3); m.add(s1); m.add(s2); m.add(s3); m.add(f1); m.add(f2); m.execute();** |

**Diagrama UML**

Por fim, foi criado o diagrama UML contendo os métodos, classes e relações da API de acordo com o que foi ministrado nas aulas da disciplina. Nesse diagrama é apresentado como a API deveria funcionar e tem como finalidade guiar o desenvolvimento do projeto.

****