Linguagem Modelica e Material de Estudo

Lucas Gabriel Cosmo Morais
Janeiro, 2020.

Índice

Iniciar uma nova linguagem parece ser um desafio, mas, com as dicas certas muita dor de cabeça pode ser evitada. Este arquivo contém uma breve introdução do que a linguagem representa e vários links que certamente ajudarão no estudo. Segue um breve roteiro com dicas e atalhos que serão de valia para uma iniciação no estudo e para uma melhor compreensão para modelar sistemas:

1. Leia este material para uma visão superficial dos conceitos.
2. Dê uma breve lida no Material do Professor Peter que está *linkado* aqui.
3. As video-aulas da Wolfram também são uma ótima oportunidade para começar modelar algo do básico (Tem muito material disponível sobre a linguagem. As video-aulas são em inglês).
4. O OMNotebook é o melhor amigo para se ter nesse momento. Muitos exemplos podem ser obtidos com ele. Assim que começar a criar modelos em Modelica replique os modelos do OMNotebook e veja como está funcionando a linguagem.
5. Sempre que tiver alguma dúvida mais específica da linguagem ou do OpenModelica (em especial do OMEdit) procure (em inglês) pelo tema no fórum do OpenModelica. É só *dar um google* com as palavras-chaves.

Conceitos da Linguagem Modelica

O Material foi adaptado das seguintes fontes:
Vídeo: Conceitos da Linguagem Modelica (com o ambiente Wolfram_SystemModeler)
https://www.wolfram.com/training/videos/ENG023/
Webbook: OMNotebook (Disponível na instalação do OpenModelica)

1. Modelica é uma linguagem para modelagem de sistemas dinâmicos. É baseada em equações, orientada a objeto e permite a composição gráfica de modelos de sistemas. Modelica possui bibliotecas pré-compiladas que permitem criar modelos facilmente e de modo rápido.

2. O coração da linguagem modelica está no princípio de conexão (Ilustrando com domínio elétrico – Leis de Kirchoff da Corrente e da Tensão):

KCL: A soma das correntes que chegam em um ponto (nó) é igual a zero.
KTL: A soma das tensões é um caminho fechado é igual zero.

Traduzindo a linguagem modelica tem-se a criação de uma classe ‘connector’ para o nó elétrico:

```
connector ElectricalPin
    flow Real i;          (Variável de fluxo)     => flow
    Real v;               (Variável de Esforço) => effort
end ElectricalPin;
```

Para conectar dois componentes (dois conectores criados) usa-se ‘connect(a, b)’. Connect é traduzido na linguagem como equações que descrevem os dois pontos. As equações geradas representam todas as variáveis de fluxo e esforço nos componentes: Como exemplo para as variáveis de fluxo entre a conexão a e b:

$$a.i + b.i = 0$$

E para as variáveis de esforço:

$$a.v = b.v$$

Com estes dois princípios pode-se modelar vários outros domínios. A biblioteca padrão do Modelica define conectores para vários domínios (Circuitos Elétricos, Magnéticos, Sistemas translacionais, rotaciomais, hidráulicos, transferência de calor e outros). Para Diagramas de Blocos e Máquinas de Estados não existe variáveis de fluxo pois deseja-se que as variáveis seja propagadas na modelagem (pode ser entendido como a conexão de dois nós elétricos, cuja tensão deve ser a mesma).

Os conectores seguem uma regra de orientação. Como o sentido da corrente (e o sinal que ela recebe) em um circuito elétrico, as variáveis de fluxo podem sem positivas e negativas em diferentes sentidos.

3. Os modelos descritos em Modelica são NÃO-CAUSAIS: Não se tem uma direção como em SIMULINK. Modelando em blocos com sistema CAUSAL, deve-se decidir o que são variáveis de entradas e saídas de um sistema. Em sistemas NÃO-CAUSAIS isto funciona de modo diferente, todas as variáveis do sistema são descritas pelas equações.

4. Com um abordagem de modelagem textual: Em um modo não causal usa-se equações para descrever relações entre as variáveis. Em um modo causal usa-se algoritmos (com fluxos fixos: A partir da entrada de uma corrente calcula-se uma tensão, a partir de uma tensão calcula-se uma corrente).
   Por exemplo, coma a lei de Ohm:
   Equações: u = R*i;

   Algoritmo: u := R*i;
   i := u/R;
   R := u/i;

5. Com uma abordagem NÃO-CAUSAL, pode-se fazer o reaproveitamento de modelos em diversos outros modelos. Por exemplo, uma máquina CA modelada como motor pode ser reaproveitada como gerador sem precisar fazer alterações em sua modelagem (diferente de um modelo CAUSAL).

6. Tipos de Classes em Modelica:
   6.1. connector

   6.2. model
   1. Usados para modelar sistemas que podem ser diretamente simulados;
   2. Modelos de subcomponentes não causais para reuso. Ex: **resistor**, inercia de sistema rotativos, capacitor de calor, etc;

   6.3. block
   1. Usados para construir sub-modelos reutilizáveis com abordagem causal (possui entrada e saídas definidas). Embora não seja o principio da linguagem Modelica pode atuar semelhante ao SIMULINK.
   2. Regra de conexão: não se pode conectar saída de bloco com saída de bloco.

   6.4. type
   1. As variáveis nos modelos são declaradas usando um dos tipos definidos (ou algum tipo derivado/personalizado usando type class): **Real , Integer, String, Boolean.**
   2. Diferentes tipos de variáveis: **constant (**Nunca muda de valor**)**, **parameter (**Não muda de valor durante a simulação, pode ser alterado entre simulações**)**, **discrete (**Muda de valor durante certos eventos**).**
   3. Variáveis não declarados como constant, parameter ou discrete podem mudar o valor durante todo o tempo.
   4. **type class** é usado para criar tipos personalizáveis derivados de tipos definidos (como um novo tipo com especificador de unidade):
      type Voltage = Real (unit = “V”);

type Length = Real (unit = “m”, min = 0);

6.5. package

1. Usado para empacotar (ou colecionar) modelos e exemplos em uma estrutura de reuso, também chamado de biblioteca.

6.6. Function

1. Usado como funções como em qualquer linguagem. Permitem entradas e saídas de dados.

7. Estrutura Hierárquica: Em Modelica pode-se estruturar hierarquicamente os modelos. Pode-se modelar um sistema com sub componentes diferentes. (Ex: Sistema PI de Controle de Velocidade de um motor ==> modelo do motor DC ==> resistor, indutor, parte eletromecânica, massa de inercia rotativa). Usa-se notação de ponto para se referir a modelos dentro de pacotes: *IntroductoryExamples.Hierarquical.TankSystem.*

8. Herança: Modelica contém heranças de dados e comportamentos de modelos.

Material para estudo da Linguagem Modelica e Ferramentas de Modelagem e Análise

Cursos em Geral ----------

A empresa Wolfram oferece aulas gratuitas sobre diversos assuntos, Machine Learning, Data Science, Wolfram Language, Modelagem e Simulação.
https://www.wolfram.com/wolfram-u/

OpenModelica ----------

Guia de Usuário do OpenModelica 1.14
https://openmodelica.org/useresresources/userdocumentation

Linguagem ----------

Modelica Cursos (longos e curtos)
https://openmodelica.org/useresresources/modelica-courses

Slides de Peter Fritzson (desenvolvedor da ferramenta OpenModelica), da linguagem e acerca de outras ferramentas comerciais que utilizam Modelica.
https://openmodelica.org/images/docs/userdocs/modprod2012-tutorial1-Peter-Fritzson-ModelicaTutorial.pdf

Wolfram SystemModeler contém:
https://reference.wolfram.com/system-modeler/

1. Tutoriais com passos iniciais na ferramenta (se assemelha ao OpenModelica):
   https://reference.wolfram.com/system-modeler/GettingStarted/Introduction.html
2. Exemplos de Modelos
3. Bibliotecas

Wolfram U contém:
https://www.wolfram.com/wolfram-u/catalog/modeling-simulation/

1. **Modelagem de Sistemas** (modelica + Wolfram SystemModeler):
   Aula sobre conceitos da linguagem Modelica e sobre a ferramenta que se assemelha ao OpenModelica
2. **Análise de Sistemas**
3. **Controle de Sistemas**
   Aula sobre conceitos de controle, espaço de estados, e dominio da frequência.

Exemplos de Modelos ----------

Wolfram System Modeler oferece alguns exemplos com código Modelica que podem ser aproveitados.
https://www.wolfram.com/system-modeler/examples/

Modelos Acadêmicos

1. LEGO Segway: Controlando Pêndulo Invertido
2. Planejamento de Caminho e Controle de um Robô Industrial
3. Ball and Beam: implantar o controlador no hardware (Arduino UNO)
4. Conversor Buck-Booster
5. Somador de 8 bits
6. Bouncing Balls: Lide com eventos e descontinuidades
7. BunggeJump
8. Pêndulo de Newton
9. Precessão giroscópica

Modelos Industriais

1. Planejamento de Caminho e Controle de um Robô Industrial
2. Retificador de Onda Completa a diodo
3. Confiabilidade ininterrupta da fonte de alimentação
4. Controle de Caminho de um satélite
5. Transmissão de um Carro (Driveline)
6. Projeto de Amortecimento da Suspensão de Caminhão
7. Arrefecimento de Motor DCPM
8. Roda com fricção a seco
9. Embreagens Acopladas
10. Amortecimento ativo: Controle as vibrações do motor diesel
11. Controle ideal para um reator de tanque continuamente agitado
12. Sistema de Tanques com OPC UA
13. Avaliação de tensões causadas por gases em expansão
14. Design de Joystick
15. Chaleira Elétrica: Processo de Aquecimento Controlado
16. Relógio de Pêndulo
17. Aquecimento doméstico: compare o consumo de energia
18. Bateria: Modelar um sistema eletroquímico