UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS

Leonardo de Paula Pinati

Desenvolvimento de atividades computacionais para disciplina de Sistemas Aeronáuticos de Acionamento

São Carlos

2020

Leonardo de Paula Pinati

Desenvolvimento de atividades computacionais para disciplina de Sistemas Aeronáuticos de Acionamento

Monografia apresentada ao Curso de Engenharia Aeronáutica, da Escola de Engenharia de São Carlos da Universidade de São Paulo, como parte dos requisitos para obtenção do título de Engenheiro Aeronáutico.

Orientador: Prof. Dr. Jorge Henrique Bidinotto

São Carlos

2020

Ficha catalográfica elaborada pela Biblioteca Prof. Dr. Sérgio Rodrigues Fontes da EESC/USP com os dados inseridos pelo(a) autor(a).

Pinati, Leonardo
P645d Desenvolvimento de atividades computacionais para disciplina de Sistemas Aeronáuticos de Acionamento / Leonardo Pinati; orientador Jorge Henrique Bidinotto. São Carlos, 2020.

Monografia (Graduação em Engenharia Aeronáutica) -- Escola de Engenharia de São Carlos da Universidade de São Paulo, 2020.

1. Sistemas Aeronáuticos de Acionamento. 2. OpenModelica. 3. Apostila. 4. Material de apoio. 5. Atividades computacionais. 6. Engenharia Aeronáutica. I. Título.

Eduardo Graziosi Silva - CRB - 8/8907

FOLHA DE APROVAÇÃO

Candidato: Leonardo de Paula Pinati
Título do TCC: Desenvolvimento de atividades computacionais para disciplina de Sistemas Aeronáuticos de Acionamento
Data de defesa: 18/12/2020

Comissão Julgadora	**Resultado**
Professor Doutor Álvaro Martins Abdala	Aprovado
Instituição: EESC - SAA	
Pesquisador Luís Antônio Vieira Pelegrineli	Aprovado
Instituição: EESC - SAA	

Presidente da Banca: Professor Doutor Álvaro Martins Abdala

_____

(assinatura)

Este trabalho é dedicado a todos docentes da EESC, como uma contribuição para o desenvolvimento e aperfeiçoamento da excelente didática demonstrada por todos meus anos de graduação

AGRADECIMENTOS

À toda a minha família, por todo o apoio e pela ajuda, que muito contribuíram desde o processo seletivo para entrar no curso até o desenvolvimento deste trabalho.

Aos amigos, que sempre estiveram ao meu lado, pela amizade incondicional e pelo apoio demonstrado ao longo de todo o período de tempo em que me dediquei a este trabalho.

Ao professor Bidinotto, por ter sido meu orientador e ter desempenhado tal função com dedicação e amizade. E ter me proporcionado a oportunidade de trabalhar em algo que servirá de apoio para os muitos outros alunos que ainda virão.

Aos professores, pelas correções e ensinamentos que me permitiram apresentar um melhor desempenho no meu processo de formação profissional ao longo do curso.

Às pessoas com quem convivi a longo desses anos de curso, que me incentivaram e que certamente tiveram impacto não só na minha formação acadêmica, como também em diversos aspectos do meu desenvolvimento pessoal. Em especial a todos amigos que dividiram república comigo. Àqueles que me acompanharam madrugadas a dentro em diversos momentos durante a graduação.

À Escola de Engenharia de São Carlos, essencial no meu processo de formação profissional, pela dedicação, e por tudo o que aprendi ao longo dos anos do curso. Que muito agregou através dos diversos grupos extra-curriculares, além de alguns programas de extensão.

"O estudo, a busca da verdade e da beleza são domínios em que nos é consentido sermos crianças por toda a vida."

Albert Einstein

RESUMO

Pinati, L. **Desenvolvimento de atividades computacionais para disciplina de Sistemas Aeronáuticos de Acionamento**. 2020. 61p. Monografia (Trabalho de Conclusão de Curso) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2020.

Desenvolvimento de material de apoio para auxiliar na utilização de software específico para a disciplina de Sistemas Aeronáuticos de Acionamento. O objetivo final consiste em possibilitar a aproximação entre o meio acadêmico e o ambiente prático dentro do contexto da indústria aeronáutica por meio da utilização de um software. O produto final do trabalho visa encurtar a curva de aprendizado dos alunos com relação ao software, aumentando o aproveitamento do tempo dedicado aos projetos da disciplina em questão.

Palavras-chave: Sistemas Aeronáuticos de Acionamento. OpenModelica. Apostila. Material de apoio. Atividades computacionais. Engenharia Aeronáutica.

ABSTRACT

Pinati, L. **Apostila OpenModelica**. 2020. 61p. Monografia (Trabalho de Conclusão de Curso) - Escola de Engenharia de São Carlos, Universidade de São Paulo, São Carlos, 2020.

Development of guiding material to assist in the use of specific software for the discipline of Aeronautical Drive Systems. The ultimate goal is that with the use of the software, there will be a coherence between the academic circle and a practical environment of the aeronautical industry. The final product of the work aims to shorten the students' learning curve in relation to software, increasing the efficiency of time spent on projects of the course.

Keywords: Aeronautical Drive Systems. OpenModelica. Booklet. Guiding material. Computational activities. Aeronautic engineering

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

EESC	Escola de Engenharia de São Carlos
UNESCO	Organização das Nações Unidas para a Educação, a Ciência e a Cultura
UNEVOC	O Centro Internacional de Educação e Treinamento
COBENGE	Congresso Brasileiro de Educação em Engenharia
ABENGE	Associação Brasileira de Educação em Engenharia
PBL	Aprendizagem Baseada em Projetos
TVET	Educação e Treinamento Técnico-Profissional
USP	Universidade de São Paulo
USPSC	Campus USP de São Carlos
APIs	Interfaces de Programação de Aplicativos

SUMÁRIO

1 INTRODUÇÃO

1.1 Motivação

Com os avanços tecnológicos, as ferramentas utilizadas dentro da indústria aeronáutica se tornaram cada vez mais sofisticadas e complexas. Atualmente não é possível se tornar um engenheiro aeronáutico competente sabendo somente as teorias dos livros. Existem hoje incontáveis softwares que conseguem traduzir todos estes conhecimentos teóricos em poucas linhas de comando. No entanto ainda há uma lacuna para materiais que consigam traduzir da melhor maneira possível como estes softwares devem ser utilizados.

Em busca de tornar os aprendizados da disciplina de Sistemas de Acionamentos mais próximos da indústria (menos teórico e mais prático), percebeu-se a necessidade de incorporar a utilização de softwares específicos para a análise destes sistemas para compor os projetos da disciplina. Essa prática possibilitaria aos estudantes visualizar melhor os conceitos aprendidos, como também os prepararem para o ambiente da indústria. Mas isto gerou uma nova lacuna: como os estudantes poderiam aprendem a utilizar estas ferramentas sem comprometer o tempo para o desenvolvimento do projeto em si?

Com o intuito de facilitar a curva de aprendizado dos alunos da disciplina, foi proposta a elaboração de uma apostila com exercícios básicos e conceitos essenciais da ferramenta em questão. Esta apostila servirá de base para os desenvolvimentos iniciais dos projetos das disciplinas. Otimizando assim o tempo de aula do professor, como também horas de estudo dos estudantes.

A apostila é dividida em 4 seções. A primeira dando uma breve introdução sobre o que é Modelica; passando depois pelos ambientes existentes no software; aprofundando nos conceitos da linguagem; por fim entrando na modelagem gráfica. Ao longo destas partes existem exercícios, que pretendem dar bases para que o aluno consiga projetar um sistema de acionamento aeronáutico ao final do semestre.

1.2 Objetivos

Objetivo Geral

A partir da motivação acima o trabalho em questão tem como objetivo aproximar o ensino da disciplina de Sistemas Aeronáuticos de Acionamentos do curso de Engenharia Aeronáutica das práticas[1] da indústria através de uma "capacitação técnica"com uma abordagem pedagógica respaldada.

Objetivos Específicos

Os objetivos específicos deste trabalho estão relacionados a compreender as principais abordagens para aproximar estudantes de engenharia do mercado de trabalho, demonstrar ganhos reais observados em literatura, justificar a utilização do software em questão, por fim elaborar uma apostila que busca utilizar destes conceitos para a capacitação no uso do software, através de exercícios. Em suma, os objetivos estão listados abaixo.

- Compreender as atuais discussões pedagógicas sobre o ensino técnico, explicitando estudos de casos;
- Contextualizar a relevância do software para a disciplina em questão;
- Elaborar apostila que contenha descrição básica do software, tutoriais e exercícios;

2 REVISÃO BIBLIOGRÁFICA

Para destacar a importância do desenvolvimento de materiais deste tipo, foi necessário tomar um passo atrás e entender melhor o universo de capacitação técnica para o mercado de trabalho e, partir daí, buscar referências e casos de sucesso. Com estas referências em mãos foi possível dimensionar o potencial impacto deste trabalho para dentro do contexto do mercado de trabalho na indústria aeronáutica.

A partir deste ensejo foi possível encontrar diversos trabalhos do Centro Internacional UNESCO-UNEVOC. Este instituto atua como um dos braços da UNESCO na área da educação, sendo responsável por promover o desenvolvimento da força de trabalho de seus países membros através do sistema de Educação e Treinamento Técnico-Profissional (TVET - *Technical and Vocational Education and Training*).

Resumidamente, o sistema TVET é uma alternativa ao ensino superior tradicional das universidades. Tem como principal objetivo garantir acesso a programas efetivos de educação de alta qualidade para todos os países membros. Ainda há um estigma quanto ao sistema, principalmente em países sub-desenvolvidos. Nestes lugares ainda permeia uma crença de que ensino técnico é inferior ao ensino das universidades. No entanto é possível observar aumento destes centros de ensino ao redor do mundo.

Aprofundando um pouco mais nos desafios encontrados por esta metodologia de educação profissional e tecnológica[2], muito foi discutido sobre a necessidade de aprimoramento para cumprir as demandas do mercado e as necessidades da indústria de modo geral.

Já no espectro de sucessos destaca-se o caso encontrado na Áustria[3]. Lá é possível observar algumas iniciativas que conferem a este uma grande importância à educação técnica e vocacional. Dentre elas podemos ressaltar a frequente atualização do currículo e a introdução de novos programas para lidar com as mudanças do mercado de trabalho.

No cenário nacional a Associação Brasileira de Educação em Engenharia (ABENGE) tem feito um extenso trabalho para melhorar a qualidade de ensino de graduação em engenharia e tecnologia no Brasil. Em diversos Congressos Brasileiros de Educação em Engenharia (COBENGE) é possível encontrar diversos artigos[4] [5] [6] [7] [8] [9] [10] [11] [12] [13] apontando maneiras de aprimorar o ensino de engenharia nas universidades. Entre as diversas publicações, muito é discutido a respeito de metodologias ativas de ensino através do *Project Base Learning* (PBL).

De modo geral é possível observar que este tipo de abordagem é capaz de melhorar o processo de ensino e aprendizado[14] além de proporcionar um maior engajamento dos estudantes de cursos de tecnologia[15].

A partir dos pontos levantados acima, é possível classificar o projeto em questão como uma metodologia ativa. Assim sendo, é de se esperar algum grau de concordância com o que foi proposto.

3 MATERIAIS E MÉTODOS

Posto a necessidade do desenvolvimento de um projeto para a disciplina em questão, resta agora dois pontos principais. A escolha do material (*software*) a ser utilizado e feita esta escolha, de que maneira é possível elaborar um documento que capacite os alunos para a utilização do mesmo.

Existem diversos *softwares* capazes de simular modelos de sistemas de modo geral. Entre eles podemos destacar:

- Dymola, desenvolvido pela Dassault systems
- OpenModelica, da OSMC
- Wolfram System, da Wolfram MathCore
- SimulationX, da ITI
- MapleSim, da MapleSoft
- AMESIM, da LMS
- Optimica Toolkit, da Modelon
- MWORKS, Tongyang Sw & Control
- IDA Simulation Env, da Equa

De todos os *softwares* listados acima, apenas um possui código aberto. Ou seja, o seu acesso é gratuito e qualquer pessoa pode sugerir alterações no código. O *software* em questão é o *OpenModelica*. E o seu caráter mais 'colaborativo' faz com que ele se sobressaia entre todos os outros da lista. Por este motivo optou-se pela sua utilização.

Dado que trata-se de um *software open-source*, de domínio público, o acesso é mais democrático, e assim a inclinação para o desenvolvimento de materiais de apoio acaba sendo maior. Além disso, partindo do premissa de que o *software* atende todos os requisitos necessários, o número de empresas e pessoas utilizando o mesmo pode ser maior, dado que não existe custo de licença. Então os verdadeiros desafios são filtrar as informações relevantes e depois compilá-las de maneira estruturada e didática.

De modo geral estes materiais podem ser divididos em duas categorias principais:

- Site oficial do OpenModelica

  - Cursos e Tutoriais;[16]
  - Artigos;[17]
  - Manuais do usuário;[18] [19]
  - Código Fonte;[20]
  - Cadernos de exercícios;[21]
  - Fóruns de discussão.[22]

- Materiais audio-visual
  - Vídeos de *YouTube*;[23] [24] [25] [26] [27]
  - Site *Spoken Tutorials*.[28]

Esta classificação é importante de ser feita pois, apesar dos objetivos serem semelhantes, há diferenças fundamentais entre as duas.

Todos os materiais encontrados no site oficial são extremamentes confiáveis. Dificilmente encontra-se informações equivocadas neste ambiente pois a manutenção do website é feita pelos mesmos desenvolvedores do *software*. O objetivo dos materiais encontrados aqui é de manutenção e atualizações quanto a novos desenvolvimentos. Pressupõe-se um conhecimento básico para a compreensão do conteúdo disponível. Grande parte dos materiais são de caráter técnico e inespecífico, apesar de servirem para qualquer tipo de aplicação, a abordagem acaba se distanciando do uso prático do *software* que é buscado para solucionar a maioria dos problemas

Para a utilização das informações vindas desta categoria o principal ponto é o de filtrar o mínimo necessário para compreender os problemas que serão propostos. Se aprofundar nestes conteúdos fará com que a pessoa se torne um desenvolvedor da ferramenta, e não um usuário, que é de fato o objetivo desta apostila.

Já quando estamos falando de materiais desenvolvidos por terceiros, o conteúdo encontrado é em sua maioria de maior relevância, pois tem como objetivo resolver casos reais e práticos. Geralmente os desenvolvedores este tipo de material são quem de fato utilizam a ferramenta no dia a dia. Em contrapartida a possibilidade de encontrar informações imprecisas também é maior.

Apesar de serem mais relevantes para o desenvolvimento da apostila, é importante destacar que existe um certo cuidado para garantir a reprodutibilidade dos conceitos apresentados nestes canais.

Feita estas considerações quanto aos materiais utilizados para a elaboração da apostila, a maneira com que estes foram compilados se torna clara. Num primeiro momento será feita uma exposição de conceitos básicos para a ambientação, a partir do que foi encontrado no site oficial.

Com a compreensão destes conceitos básicos, os mesmos serão utilizados o mais rapidamente em exercícios. Estes exercícios foram retirados em sua maioria de vídeos na língua inglesa e traduzidos para o português.

4 RESULTADOS E CONSIDERAÇÕES FINAIS

4.1 Resultados

O produto final deste trabalho foi o desenvolvimento da apostila anexada. A partir da utilização desta apostila nas atividades didáticas da disciplina de Sistemas Aeronáuticos de Acionamentos será possível o desenvolvimento de um trabalho final mais completo e conceitualmente mais acessível.

A apostila em questão também será relevante por acrescentar uma metodologia pedagógica relevante no ensino de engenharia. Por se tratar de um material idealizado dentro de uma abordagem PBL, é de se esperar que o interesse e o engajamento dos estudantes aumente a partir da resolução de seus exercícios.

Além de garantir um melhor aproveitamento no desenvolvimento do trabalho da disciplina citada acima, a utilização da apostila será importante para que os alunos tenham um certo nível de conhecimento técnico na utilização de uma ferramenta robusta na solução de problemas reais de engenharia na indústria aeronáutica.

4.2 Conclusão

Para aproximar o ensino de Sistemas Aeronáuticos de Acionamento do curso de Engenharia Aeronáutica das práticas da indústria, definiram-se três objetivos específicos. O primeiro, de compreender a importância de uma abordagem de ensino voltada a projetos no contexto de engenharia. Segundamente, contextualizar a utilização *software Open Modelica* para a indústria. Por fim, a elaboração de uma apostila para ser utilizada em sala de aula, para guiar os alunos no desenvolvimento do projeto final da disciplina.

Após levantamento de literatura, foi possível observar uma tendência geral de metodologias mais práticas no ensino técnico. Entre as principais carências observadas, foi destacado que a utilização de metodologias PBL são capaz de saná-las.

O projeto a ser desenvolvido de acordo com a abordagem citada, terá como base a utilização do *software Open Modelica*. A escolha da utilização deste *software* foi pautata principalmente por ser de código aberto. Devido a esta característica do mesmo e sua eficiência, é possível encontrar diversos casos de utilização da ferramenta no contexto abordado.

Buscando auxiliar os estudantes durante o desenvolvimento do projeto, foi elaborada uma apostila (Anexo 1) com descrições básicas sobre o funcionamento do *software* juntamente com exercícios propostos. De forma que os alunos estejam capacitados tecnicamente para esta área da indústria.

4.3 Desenvolvimentos futuros

O desenvolvimento desta apostila em questão tem como objetivo habilitar os alunos da engenharia aeronáutica para o uso do *software* para casos básicos. Apesar de cumprir este papel, é importante salientar que o aprimoramento do conhecimento para o utilização desta ferramenta ainda é necessário para a sua utilização na indústria. Dito isto, a última seção deste trabalho foca em potenciais desenvolvimentos futuros para atingir este próposito.

Um dos principais usos na indústria está relacionado a simulações constantes com alterações em variáveis de entrada. A maneira de se fazer isto envolve o desenvolvimento de códigos de programação para realizar centenas de simulações automaticamente. Atualmente Python é a linguagem de programação mais utilizada [29]. Portanto a integração entre Python e Open Modelica através de APIs seja relevante para desenvolvimentos futuros, a partir de exemplos documentados [30]

REFERÊNCIAS

1 LIND, I.; ANDERSSON, H. Model based systems engineering for aircraft systems – how does modelica based tools fit? In: **Proceedings from the 8th International Modelica Conference, Technical Univeristy, Dresden, Germany**. Linköping University Electronic Press, 2011. Disponível em: <https://doi.org/10.3384/ecp11063856>.

2 TVETJOURNAL. **9 Challenges to TVET in Developing Countries**. Disponível em: <https://www.tvetjournal.com/tvet-challenges-in-developing-countries.html>.

3 EXTERIORES, M. das R. **Mundo Afora - Educação Profissional e Tecnológica**. Disponível em: <https://sistemas.mre.gov.br/kitweb/datafiles/Oslo/pt-br/file/09_Cultural/09-10-Mundo_Afora_14.pdf>.

4 MONTEIRO, S. B. S.; QUIRINO, M. G.; ZINDEL, M. L.; OLIVEIRA, E. C.; RODRIGUES, E. C. C. **Uma Nova Abordagem De Ensino De Engenharia: Aprendizagem Baseada Em Projetos (Pjbl) Na Disciplina PSP1 Do Curso De Engenharia De Produção Da UnB**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art2028.pdf>.

5 PEREIRA, A. G.; JúNIOR, C. F. de A. **Apredizagem Baseada em Problemas e o Ensino para Dimensionamento de Sistema de Hidrantes Prediais na Graduação de Engenharia**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1567.pdf>.

6 FILHO, C. R. da S.; CABRAL, J. A. **Aprendizagem Baseada Em Problemas E a Detecção De Descargas Atmosféricas: Um Estudo De Caso No Curso De Engenharia Elétrica Do Instituto Superior TUPY**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1826.pdf>.

7 BITTENCOURT, J. C. N.; ROC, A. S.; DUARTE Ângelo A.; SANTOS, J. A. M. **Avaliando a Eficácia De Problemas Aplicados Em Uma Disciplina De Sistemas Digitais Usando a Aprendizagem Baseada Em Problemas**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1752.pdf>.

8 SILVA, L. M. **Ensino Para Nível Superior De Conceitos "PBL" Para Engenharia De Controle E Automação Industrial**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1602.pdf>.

9 SILVA, J. F. de J. **Formação Em Engenharia E Desenvolvimento De Competências a Partir Do Uso Do Método PBL: Relato De Experiência**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1755.pdf>.

10 SOARES, L. R.; LUZ, I. A. dos S.; SANTOS, D. M. B. dos; PINTO, G. R. P. R. **O Processo De Formação Dos Estudantes De Engenharia De Computação Da UEFS Para O Uso Do Método PBL: a Sexta Edição Da Oficina PBL**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1811.pdf>.

11 PINTO, G. R. P. R.; SENNA, C. P. P.; COSTA, R. A. da; FILHO, S. S. S.; PEREIRA, H. B. de B. **PBL-VE: Um Ambiente Virtual Para**

Apoiar a Aprendizagem Baseada Em Problemas. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1695.pdf>.

12 TORRES, R. N.; ALENCAR, R. F. M. de; N'GUESSAN, A. J. B. R. K.; VIANA, D. M.; MARANHãO, A. C. K.; GARROSSINI, D. F. **Projetos Integradores: Uma Reflexão Sobre a Aplicação De Experiências Com Base Na Aprendizagem Orientada Por Projetos**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art2077.pdf>.

13 ZANOTTI, J. S.; CARMO, C. T. do. **Simulação Como Estratégia De Ensino-aprendizagem: Estudo De Fatores Contextuais Críticos Para O Sucesso Em Sala De Aula**. 2011. Disponível em: <http://www.abenge.org.br/cobenge/arquivos/8/sessoestec/art1919.pdf>.

14 SANTOS, Y. B. I.; MENDES, S. B. R.; PELAES, T. S. **O ensino de ferramentas computacionais aplicadas a Engenharia de Produção: um método diferenciado**. 2005. Disponível em: <http://www.abepro.org.br/biblioteca/enegep2005_enegep1101_0995.pdf>.

15 MAZINI, S. R. **A Aplicação De Metodologias Ativas De Ensino E Aprendizagem: Um Estudo De Caso Da Aplicação Do Project Based Learning Em Um Curso De Engenharia De Produção**. 2018. Disponível em: <https://www.passeidireto.com/arquivo/81358738/xxv-simpep-art-286>.

16 FRITZSON, P.; POP, A. **Introduction to Object-Oriented Modeling and Simulation with Modelica and OpenModelica**. 2020. Disponível em: <https://openmodelica.org/images/M_images/200204-ModelicaTutorial-slides-PeterFritzson-AdrianPop-MODPROD2020.pdf>.

17 FRITZSON, P.; POP, A.; ABDELHAK, K.; ASHGAR, A.; BACHMANN, B.; BRAUN, W.; BOUSKELA, D.; BRAUN, R.; BUFFONI, L.; CASELLA, F.; CASTRO, R.; FRANKE, R.; FRITZSON, D.; GEBREMEDHIN, M.; HEUERMANN, A.; LIE, B.; MENGIST, A.; MIKELSONS, L.; MOUDGALYA, K.; OCHEL, L.; PALANISAMY, A.; RUGE, V.; SCHAMAI, W.; SJöLUND, M.; THIELE, B.; TINNERHOLM, J.; ÖSTLUND, P. The OpenModelica integrated environment for modeling, simulation, and model-based development. **Modeling, Identification and Control: A Norwegian Research Bulletin**, Norwegian Society of Automatic Control, v. 41, n. 4, p. 241–295, 2020. Disponível em: <https://doi.org/10.4173/mic.2020.4.1>.

18 CONSORTIUM, O. S. M. **OpenModelica User's Guide**. [S.l.]: Open Source Modelica Consortium (OSMC), c/o Linköpings universitet, Department of Computer and Information Science, 2020. (v1.17.0-dev-263-gf765c01386).

19 FRITZSON, P.; POP, A. **OpenModelica System Documentation**. 2014. Disponível em: <https://github.com/OpenModelica/OpenModelica-doc/blob/master/OpenModelicaSystem.pdf>.

20 OPENMODELICA Code Library. 2020. Disponível em: <https://github.com/OpenModelica/OpenModelica>.

21 OMWEBBOOK. 2018. Disponível em: <http://omwebbook.openmodelica.org/DrModelica>.

22 OPENMODELICA Forum Website. 2020. Disponível em: <https://openmodelica.org/forum>.

23 TECH, B. **Modelica Tutorials for Beginners playlist**. 2020. Disponível em: <https://www.youtube.com/playlist?list=PLwpmIf9ZQ3Ex9svxIo7WrWjPESFU21oDE>.

24 OPENMODELICA Tutorial. 2019. Disponível em: <https://www.youtube.com/watch?v=m0Ahs8fEN28&ab_channel=NSTUFACE>.

25 CROWLEY, M. **Hydraulic Modelling with Modelica & SimulationX**. 2017. Disponível em: <https://www.youtube.com/watch?v=cRukWyS6lXY&ab_channel=MikeCrowley>.

26 SANDROCK, C. **Simulate a feedback control system in OpenModelica**. 2016. Disponível em: <https://www.youtube.com/watch?v=Dw66ODbMS2A&ab_channel=CarlSandrock>.

27 QUICK introduction to OpenModelica in graphical mode. 2019. Disponível em: <https://www.youtube.com/watch?v=GhtBMIlO70w&ab_channel=Processdynamicsandcontrol>.

28 SPOKEN Tutorials - Open Modelica. 2020. Disponível em: <https://spoken-tutorial.org/tutorial-search/?search_foss=OpenModelica&search_language=English>.

29 COMPUTERWORLD. **As 7 linguagens de programação que você deve aprender em 2020**. 2020. Disponível em: <https://computerworld.com.br/carreira/as-7-linguagens-de-programacao-que-voce-deve-aprender-em-2020/>.

30 OPENMODELICA. **Use of Modelica + Python in Process Systems Engineering Education**. 2018. Disponível em: <https://www.openmodelica.org/images/M_images/OpenModelicaWorkshop-2018/SeborgCSTR_all_notebooks_pdf.pdf>.

31 Disponível em: <https://openmodelica.org/>.

Appendices

ANEXO A – APOSTILA OPENMODELICA

A.1 Introdução

A.1.1 O que é Modelica?

Modelica pode ser definida como uma linguagem para modelagem de sistemas cíber-físicos complexos. Suas aplicações vão desde robótica, sistemas biológicos, satélities e aeronáutica. Tem como principal função a simulação destes modelos. Entretanto, não está limitada a isto. Há também outras aplicações como a otimização de modelos, por exemplo.

É importante destacar que apesar de tratarmos Modelica como um *software* durante grande parte deste documento, o *software* que iremos utilizar é o Open Modelica, que foi desenvolvido a partir da linguagem Modelica. A partir daqui qualquer menção a "Modelica" está sendo referido ao *software* em si, mas é importante ter em mente que existem outros *softwares* que utilizam da mesma linguagem:

- Dymola (Dassault systems)
- Wolfram System Modeles (Wolfram MatchCore)
- MapleSim (MapleSoft)
- AMESIN (LSM)

A programação da linguagem é voltada a objetos, assim como Python. A estrutura da linguagem é baseada na ampla utilização de classes. Desta maneira a generalização entre diferentes tipos de domínio se torna mais fácil e intuitivo.

Entre suas principais características é possível destacar 4:

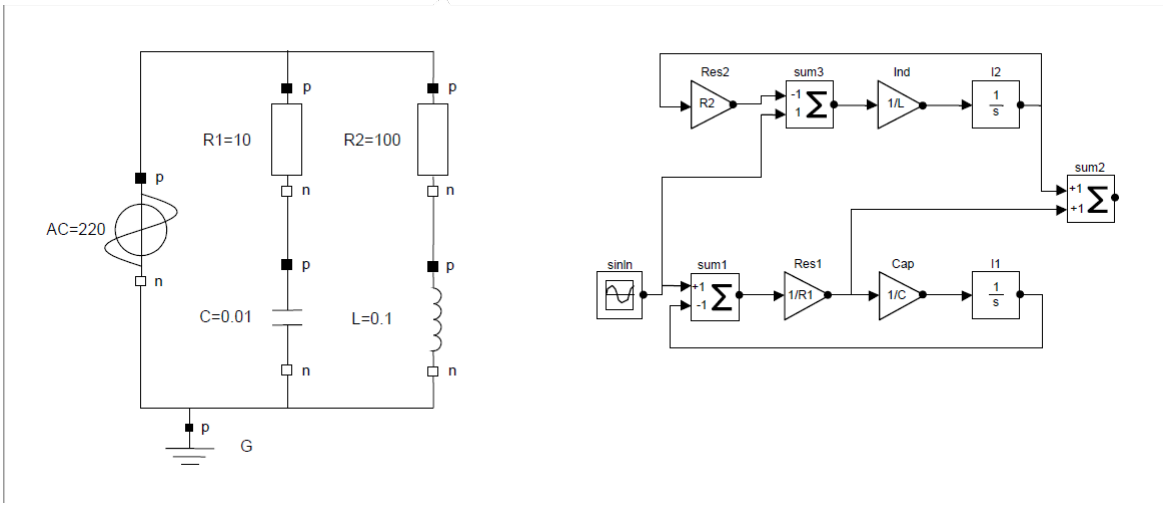
- Modelagem multi-domínio;
- Simulações e modelagens híbridas;
- Hirarquia visual acausal entre componentes de modelagem;
- Linguagem declarativa e tipada baseada em equações.

A.1.2 Modelica x Outros *Softwares*

É possível montar e simular modelos tanto pela interface gráfica quanto por linhas de comando. Além disso a equivalência entre ambos é de um para um.

Uma das principais vantagens da Modelica é sua interface gráfica intuitiva que preserva as relações físicas entre os objetos da simulação. Ao olhar um modelo dentro do ambiente gráfico da Modelica, é fácil imaginá-lo na vida real. Enquanto outros *softwares*, como Simulink, demandam um certo nível de abstração e conhecimento prévio da ferramenta para atingirmos o mesmo objetivo.

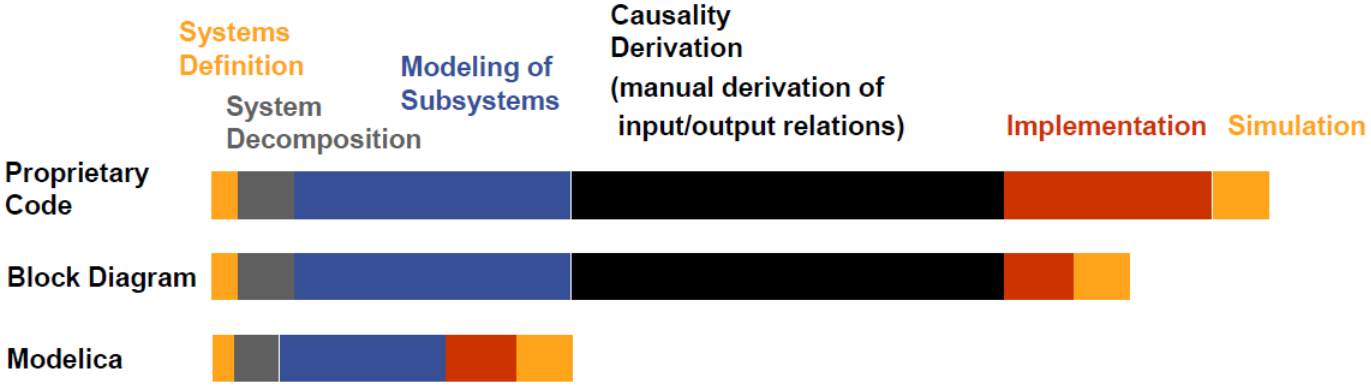
Figura 1 – Equivalência entre modelo Modelica x Simulink



Outro ponto que coloca Modelica a frente de seus competidores é o fato de que suas equações são definidas de fato como equações, e não declarações de variáveis. Ou seja, a manipulação das variáveis se dá mais facilmente.

Quando comparamos o esforço despendido do usuários para traçar a relação entre as variáveis para configurar o modelo e enfim gerar resultados, temos a seguinte cenário:

Figura 2 – Esforços desprendidos para simular um novo modelo



A.1.3 Instalação do *software*

O *software* Open Modelica está disponível tanto para o sistema operacional Windows, quanto para Linux. O suporte para MAC foi descontinuado, então para instalação do *software* nestes dispositivos é necessário utilizar uma máquina virtual Linux. Para guiar a instalação em todas plataformas é possível encontrar instruções no *website*[31] do Open Modelica. Para encontrar o guia referente a cada dispositivo basta encontrá-lo no seguinte menu:

Figura 3 – Aba de instalação do site

Para a instalação no Windows é necessário fazer o *download* do arquivo executável e seguir com a interface padrão de instalação de *software*. Para sistemas Linux é possível fazer a instalação direto do *prompt* de comando. Por fim, usuários MAC devem primeiro ter uma máquina virtual Linux, onde farão a instalação do *software* de maneira análoga ao Linux.

A.2 Primeiros passos

A.2.1 Ambiente OpenModelica

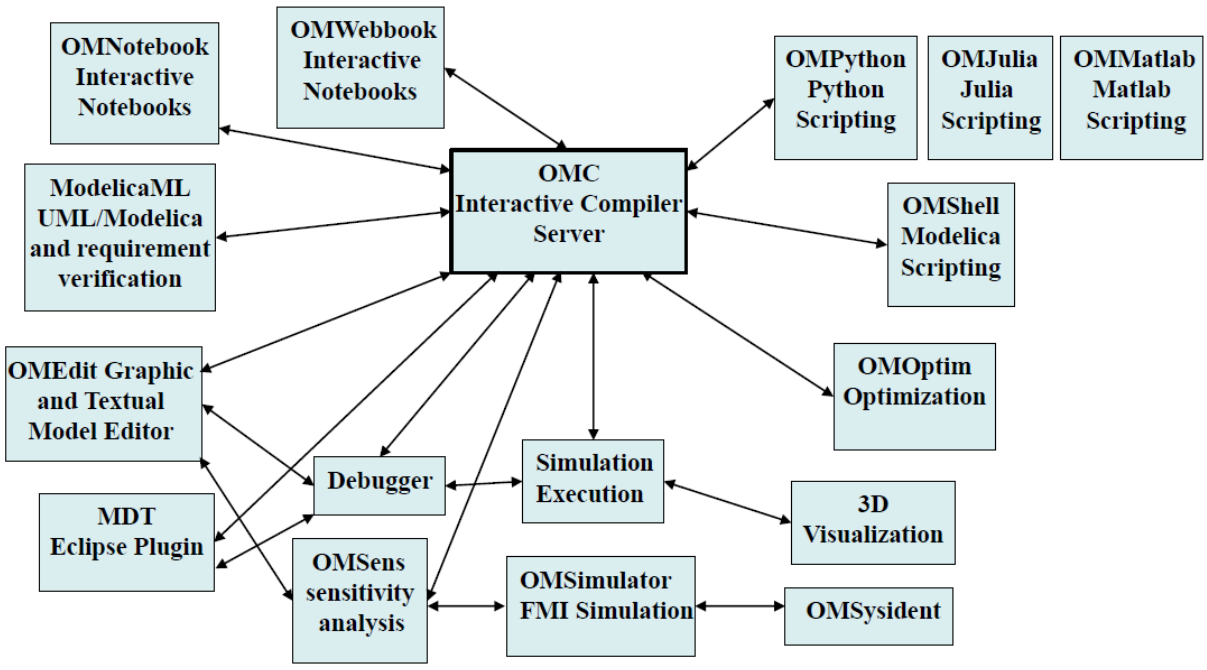
Agora que o *software* já está instalado, é importante observar que existem diversas ferramentas que compreendem Open Modelica. A lista de todas as ferramentas disponibilizadas durante a instalação é a seguinte:

- Advanced Interactive OpenModelica Compiler (OMC)
- Interactive OpenModelica Shell (OMShell)

- OpenModelica Notebook (OMNotebook)
- DrControl Under OMNotebook
- OpenModelica Connection Editor (OMEdit)
- OMEdit with Interactive Simulation
- OpenModelica Equation Model Debugger
- OMOptim
- Modelica Development Tooling (MDT)
- Modelica Modeling Language (ModelicaML)
- OpenModelica Python Interface (OMPython)
- OpenModelica Simulator (OMSimulator)

A arquitetura como estas ferramentas estão organizadas pode ser observada na seguinte imagem:

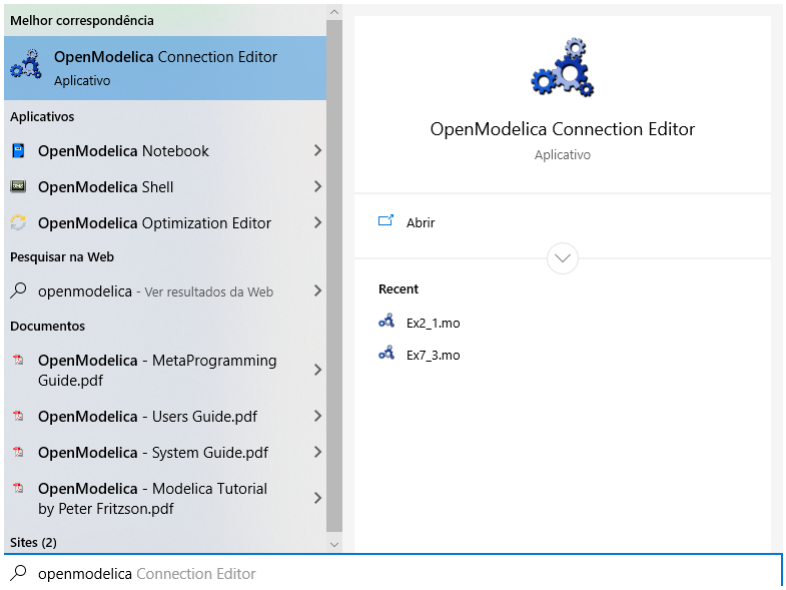
Figura 4 – Arquitetura de ferramentas [extraído de tutorial do website]



Dentre as ferramentas listadas, a de maior importância e que vamos abordar nesta apostila é a OMEdit, a interface de modelagem gráfica. É a partir desta interface que vamos montar os modelos a partir de blocos separados em diferentes bibliotecas.

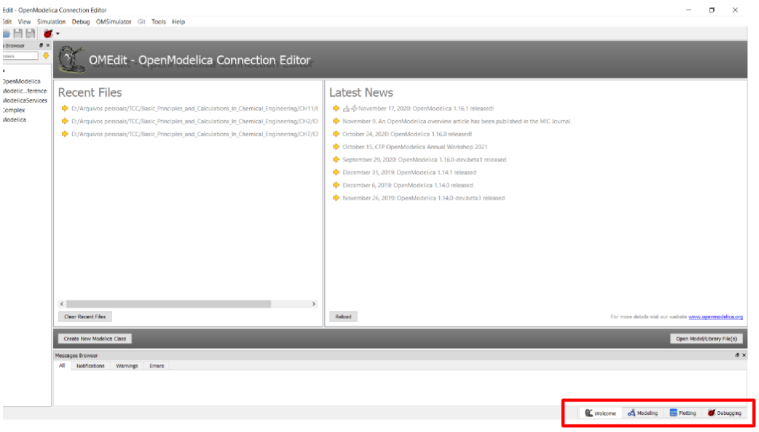
Para inicialização do programa, basta procurar por openmodelica no computador e selecionar o programa com nome "OpenModelica Connection Editor", conforme a imagem 5.

Figura 5 – Programa OMEdit



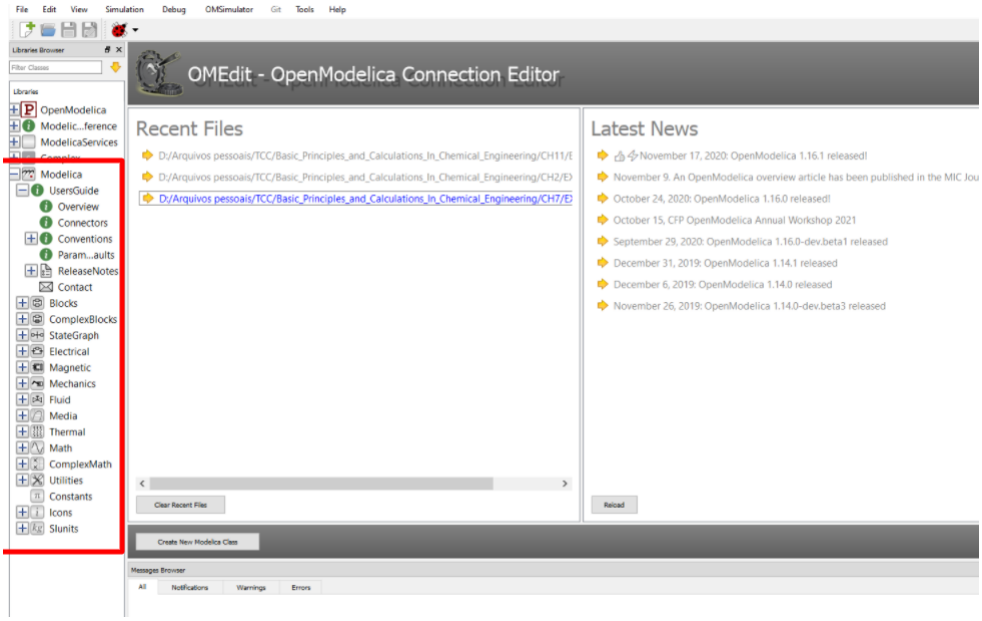
Dentro desta interface gráfica é necessário destacar 3 regiões. No canto inferior esquerdo é possível observar 4 diferentes abas que podemos alternar conforme necessidade. Em "modeling" iremos construir os nossos modelos. Com uma simulação já realizada, podemos observar seus resultados na aba "plotting". Quando precisamos Corrigir algum tipo de erro, principalmente com relação a equações, utilizamos a aba debugging.

Figura 6 – Programa OMEdit destaque de janelas



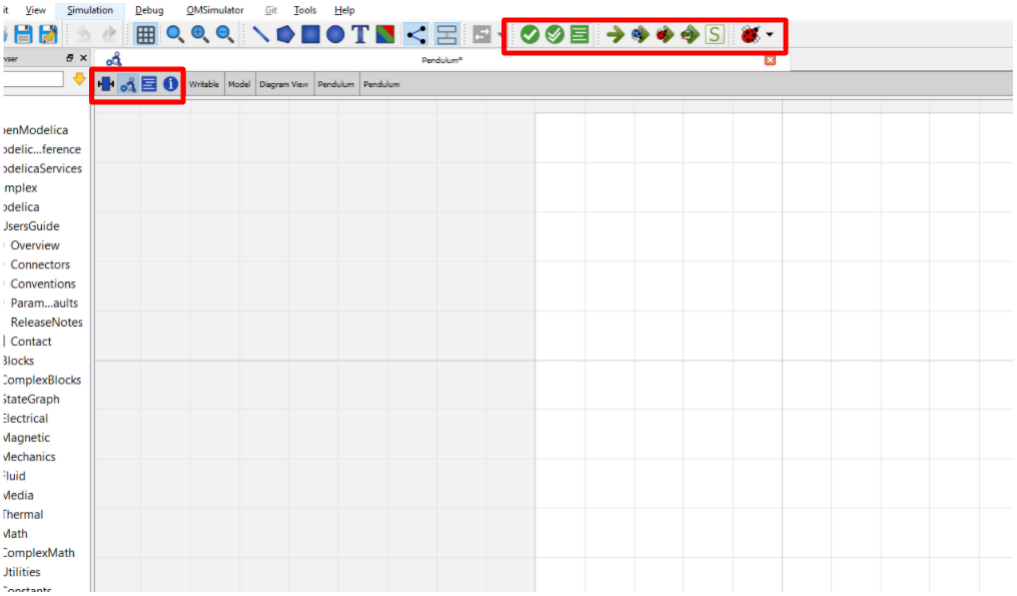
Já na parte esquerda do programa é possível notar a biblioteca de módulos. Mais especificamente, dentro de Modelica é que pegaremos os principais blocos para serem utilizados nos exercícios.

Figura 7 – Programa OMEdit destaque bibliotecas



Ao acessarmos um novo modelo, iremos nos deparar com botões referentes às simulações, assim como algumas abas referentes a visualização deste modelo.

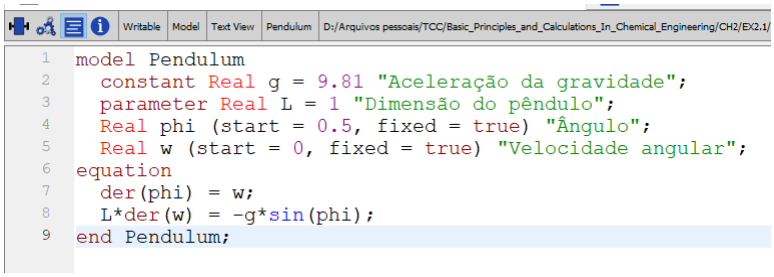
Figura 8 – Programa OMEdit botões de simulação e abas



A.2.2 Simulação inicial

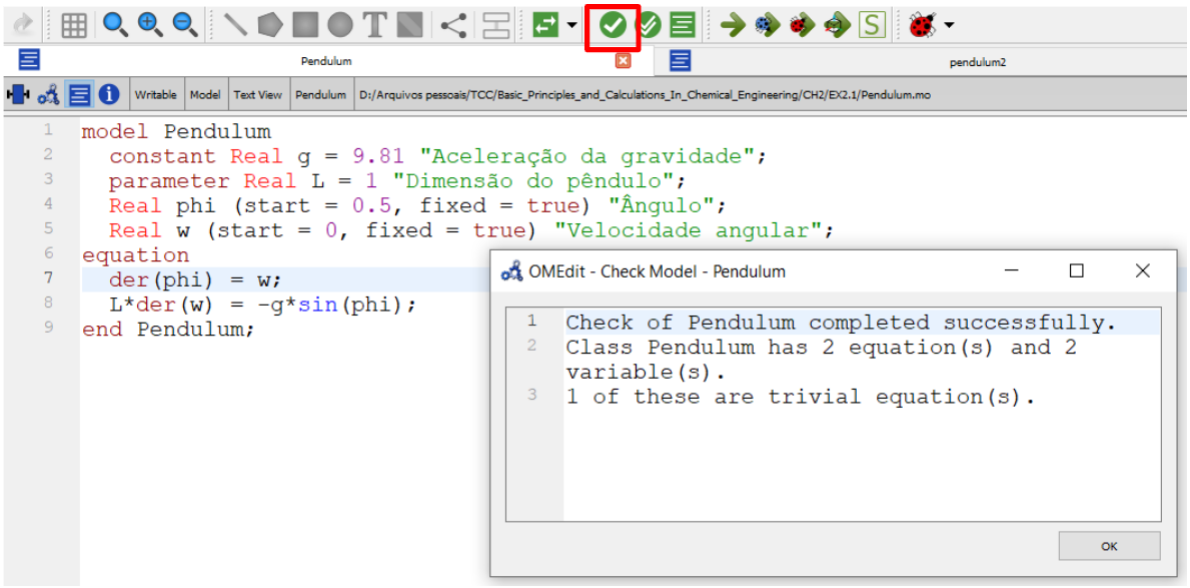
Com o intuito de ilustrar a as diferentes abas e um processo de simulação, vamos criar um modelo simples de um pêndulo a partir de suas equações e definições de variáveis como a seguir:

Figura 9 – Parâmetros de simulação pêndulo



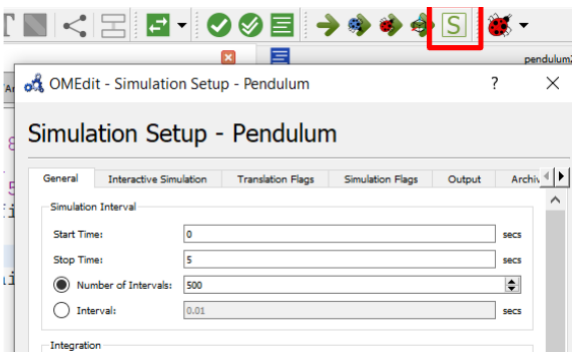
Definindo estes parâmetros na aba de visualização por texto, podemos verificar a integridade das equações definidas a partir do botão "check model", conforme imagem a seguir:

Figura 10 – Checagem modelo Pêndulo



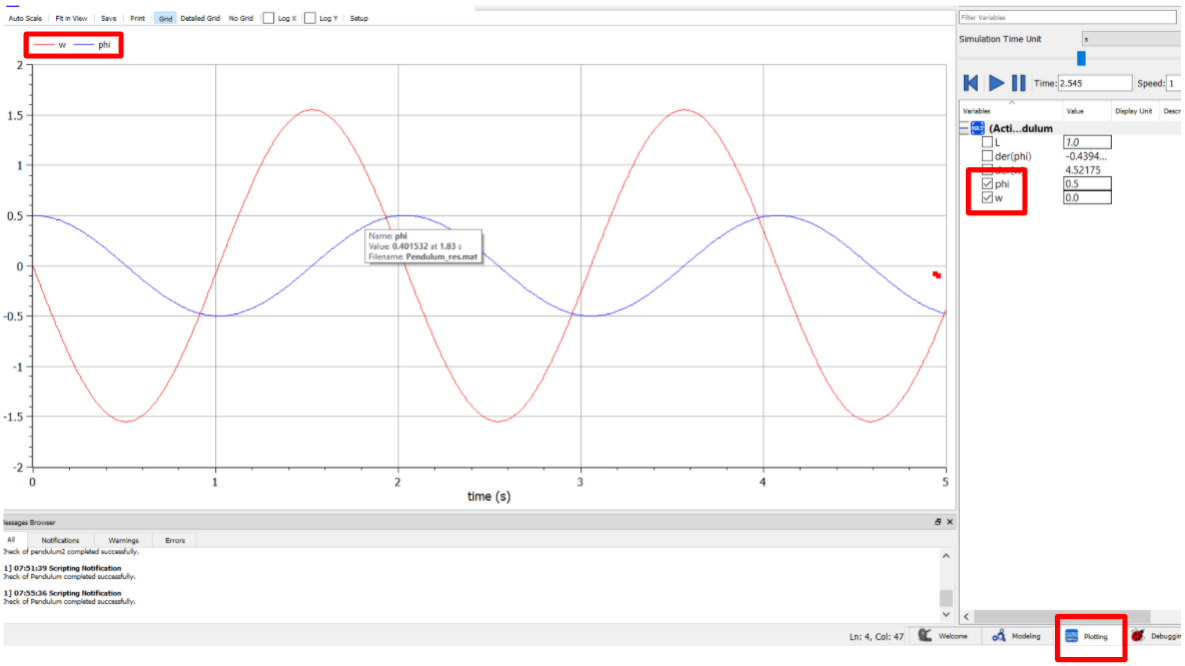
Então podemos enfim fazer a nossa primeira simulação para os primeiros 5 segundos, utitilizando a função simular:

Figura 11 – Simulação modelo Pêndulo



Para a análise de resultados iremos observar a aba "Plotting" onde podemos escolher quais variáveis queremos plotar no gráfico temporal.

Figura 12 – Resultados modelo Pêndulo



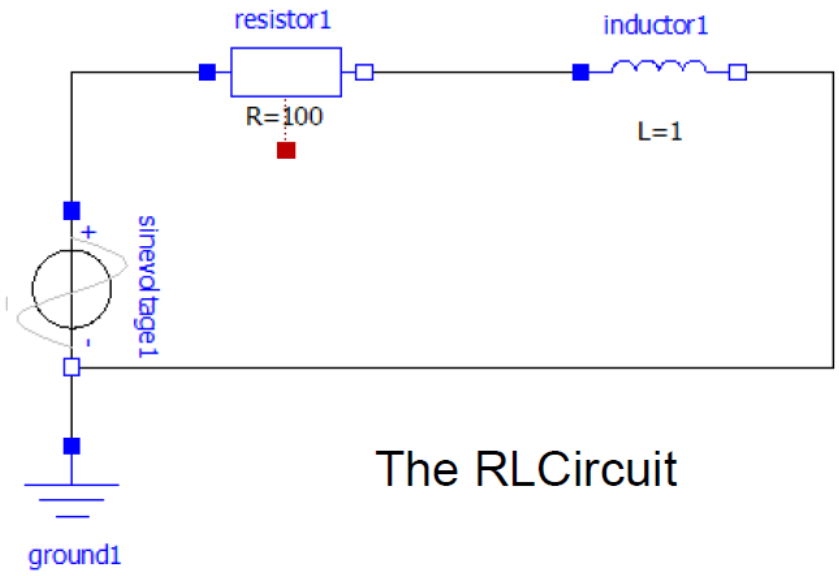
E isto encerra nossa demonstração da ferramenta, restando agora a resolução de exercícios.

A.3 Exercícios

A.3.1 Exercício 1 - Sistema Elétrico Básico

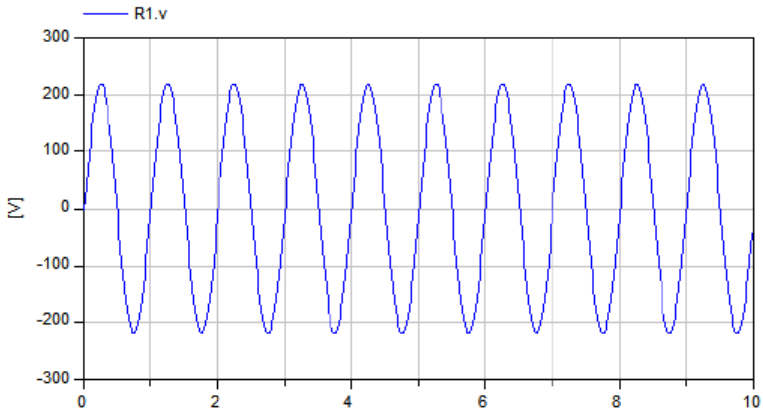
O objetivo deste exercício é montar um modelo básico de sistema elétrico com gerador, resistor e indutor, como a figura abaixo:

Figura 13 – Circuito elétrico básico



A partir deste modelo, iremos fazer uma simulação do sistema, chegando ao seguinte resultado:

Figura 14 – Resultado Simulação Circuito RLC



- Abrir programa OpenModelica Connection Editor;
- Criar um nova classe do tipo modelo;

- Definir um nome para o modelo. Sugestão: *RLCircuit*;

- Adicionar componentes indicados;

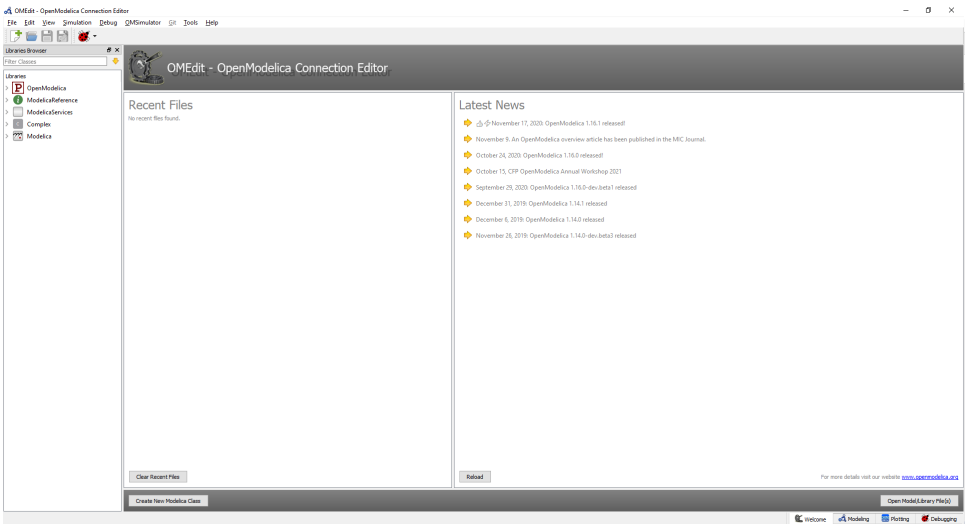
- Conectar componentes;

- Realizar simulação;

- Plotar variáveis.

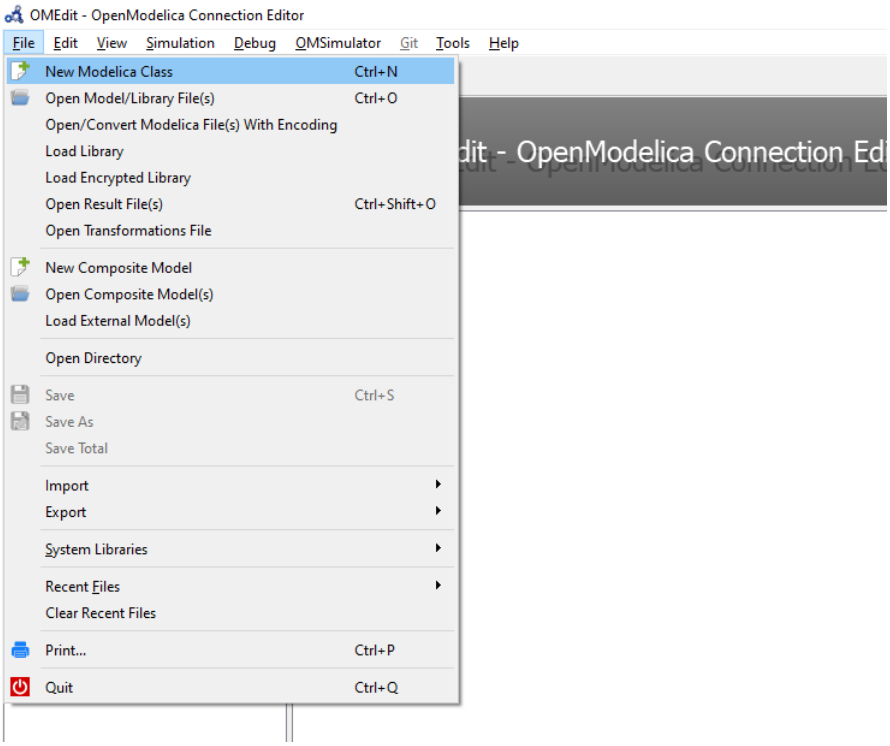
Certificando que o processo de instalação do programa foi feito de maneira adequada, abra o aplicativo "OpenModelica Connection Editor" (OMEdit). A interface do programa pode ser observada segundo imagem:

Figura 15 – Interface Gráfica



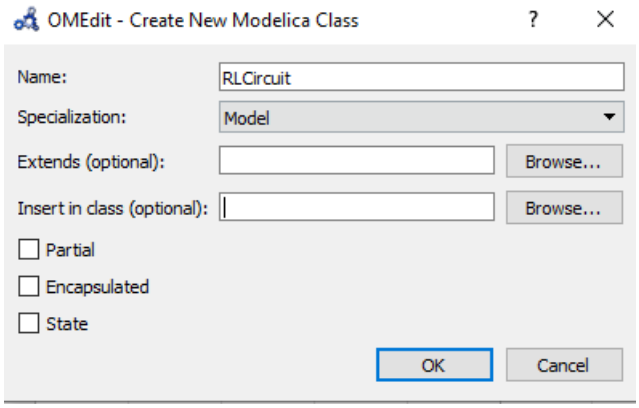
Para criar um novo modelo, é necessário acessar a opção "New Modelica Class", que pode ser feita através do menu "File" no canto superior da tela, ou com o atalho de teclado Crtl+N.

Figura 16 – Criação de novo modelo



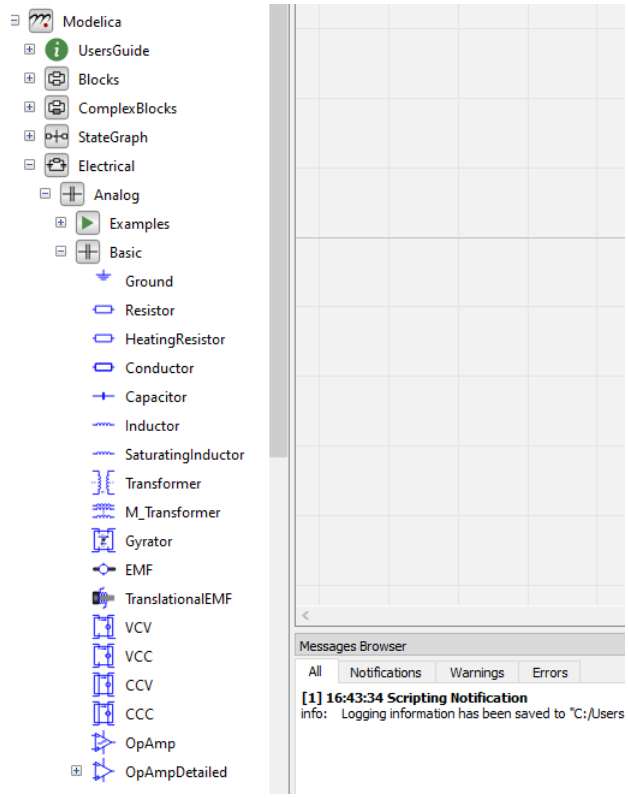
Então deve-se preencher os campos "*Name*" e "*Specialization*" conforme exemplo abaixo.

Figura 17 – Campos a serem preenchidos



Agora que já temos um novo modelo criado, basta adicionar os componentes ilustrados inicialmente. Para este exercício iremos utilizar componentes da biblioteca *Modelica.Electrical.Analog*. Resistor, indutor e terra estão dentro do pacote *Basic*, enquanto o sinal do gerador está dentro de *Sources*.

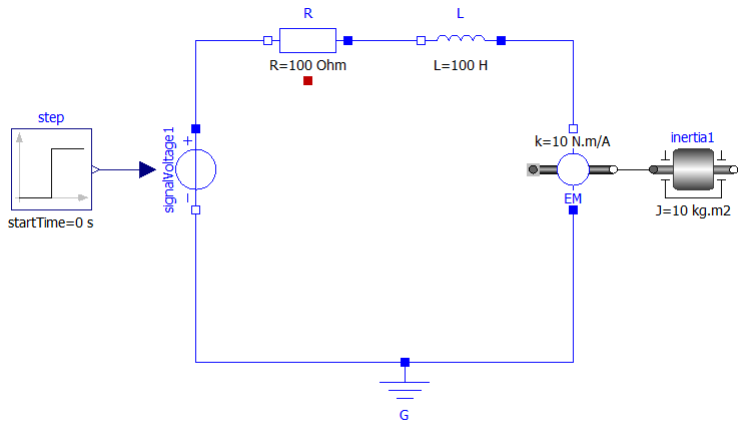
Figura 18 – Localização de componentes



A.3.2 Exercício 2 - Motor DC

Para a simulação de um motor DC vamos utilizar elementos elétricos e mecânicos. A composição destes elementos num mesmo moledo é feita da mesma maneira que o exemplo anterior, onde tratamos de um sistema exclusivamente elétrico. O sistema que iremos montar é ilustrado na imagem abaixo.

Figura 19 – Sistema Motor DC

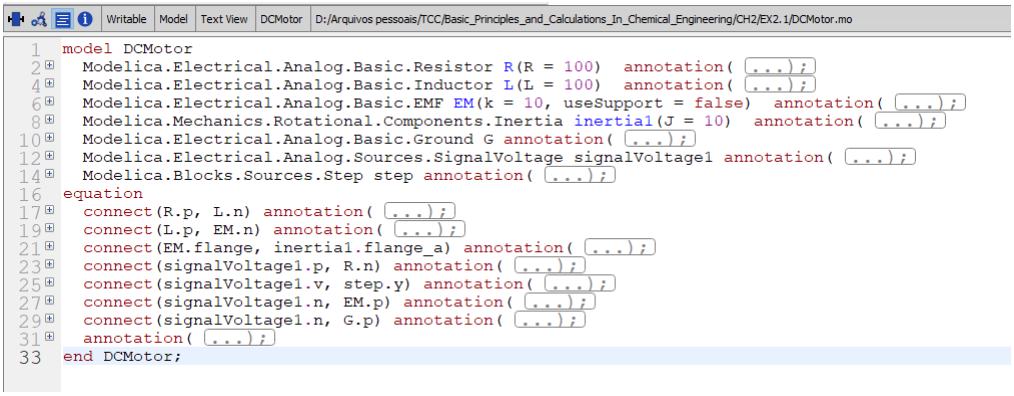


A lista de elementos utilizados é a seguinte:

- Modelica.Blocks.Sources.Step
- Modelica.Electrical.Analog.Basic.Ground
- Modelica.Electrical.Analog.Basic.Resistor
- Modelica.Electrical.Analog.Basic.Inductor
- Modelica.Electrical.Analog.Basic.EMF
- Modelica.Electrical.Analog.Sources.SignalVoltage
- Modelica.Mechanics.Rotational.Components.Inertia

As conexões entre os elementos podem ser observadas na figura, mas para garantirmos que estamos fazendo as conexões necessárias, é possível alterar o tipo de visualização de diagrama para texto. Ao acessar o visual de texto, é possível observar com detalhes as conexões entre os componentes e a "terra", além de comparar os valores dos parâmetros que foram definidos para esta simulação.

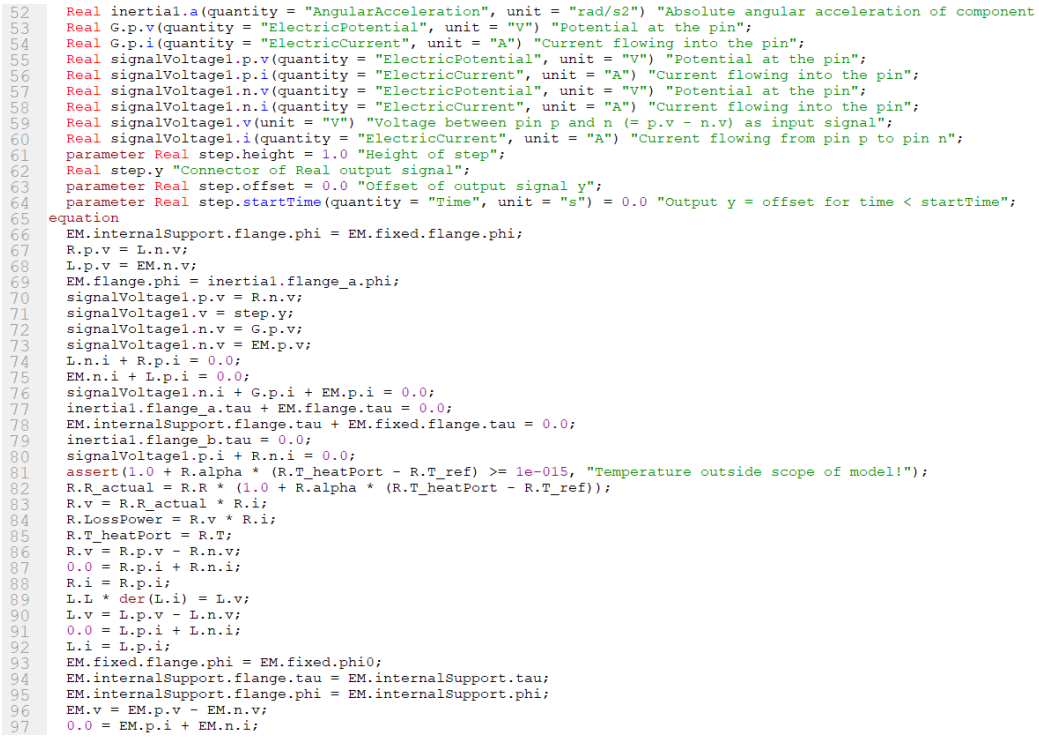
Figura 20 – Conexões entre Componentes



Com todos os parâmetros definidos, conexões, basta checar o modelo para confirmar a relação entre variáveis e equações e então realizar a simulação.

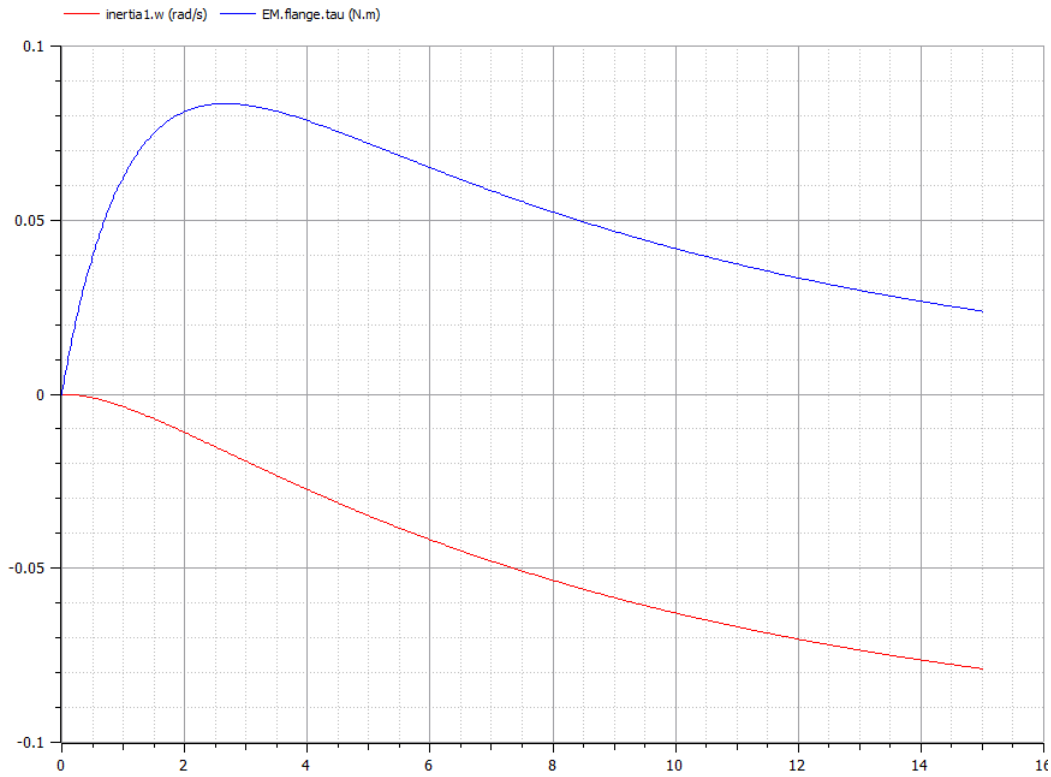
Antes de simular o modelo, uma boa prática é a confirmação de quais equações estão sendo utilizadas para os cálculos da simulação. Para fazer isto basta clicar no botão de "Instantiate model", que uma janela semelhante a esta abrirá:

Figura 21 – Lista de Equações Motor DC



Depois que a simulação foi computada, é possível observar variáveis como a velocidade angular do componente e o torque na janela de gráficos.

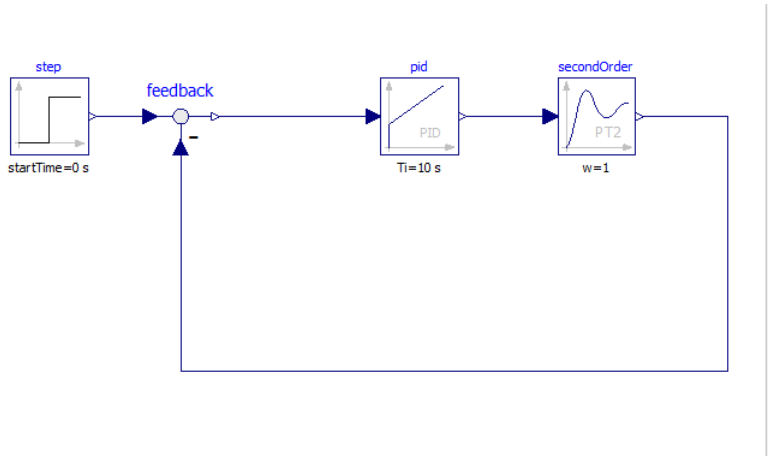
Figura 22 – Resultados da Simulação Motor DC



A.3.3 Exercício 3 - Controle PID

Neste exercício vamos elaborar um sistema de controle PID simples. Para isto vamos utilizar um sinal de segunda ordem, uma entrada do tipo degrau, o controlador PID sendo alimentado por um feedback. Conforme ilustra a imagem a seguir.

Figura 23 – Sistema controlador PID



Todos elementos utilizados podem ser encontrados na biblioteca "Blocks".

- Modelica.Blocks.Continuos.PID
- Modelica.Blocks.Continuos.SecondOrder
- Modelica.Blocks.Math.Feedback
- Modelica.Blocks.Sources.Step

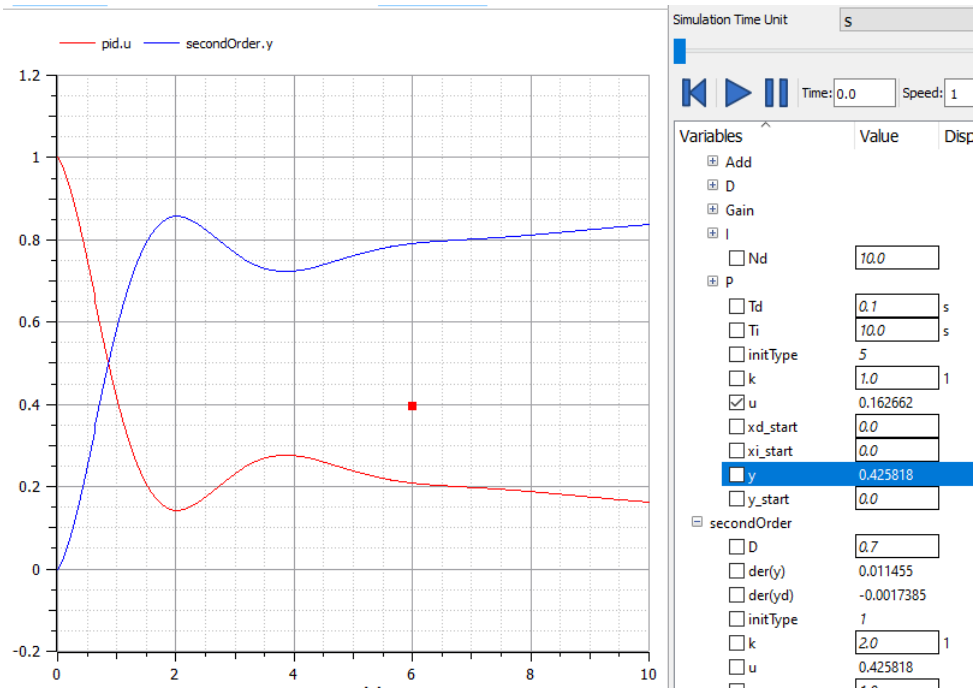
Para este sistema é preciso definir os parâmetros tanto do controlador quanto da função transferência. Para o controlador: ganho (k) e a constante de tempo do integrador (Ti). Para a função de segunda ordem: ganho (k), frequência angular (w), o fator de amortecimento (D).

Após a definição de valores para estes parâmetros, confira se as equações estão corretas através do botão "Check Model". O número de equações deve ser igual ao número de variáveis. Todas as equações definidas por estes blocos, assim como os parâmetros e o tipo de conexão entre eles, podem ser encontradas no menu "Instantiate Model", como mostra a figura a seguir.

Figura 24 – Lista de Equações Sistema PID

Com todas as definições citadas acima, podemos simular o modelo em questão. Após o término da simulação, a aba de plotagem dos gráficos deve aparecer automaticamente, mas caso isto não aconteça, basta acessá-la no canto inferior direito. Os resultados da simulação devem se assemelhar com a seguinte imagem:

Figura 25 – Resultados Sistema PID



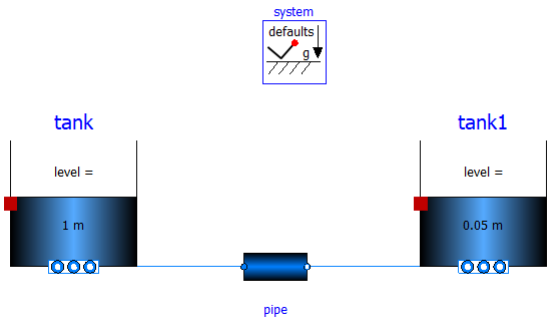
Dentro deste menu podemos selecionar quais variáveis queremos colocar no gráfico.

Além disso, é possível alterar valores dos parâmetros nesta mesma janela e simular o modelo novamente.

A.3.4 Exercício 4 - Sistema Hidráulico

O último exercício a ser proposto corresponde a um sistema hidráulico. Neste exemplo vamos simular a trasferência de um fluido entre dois tanques através de um tubo. Apesar de se tratar de um exemplo básico, devemos notar que para este tipo de ambiente são necessárias algumas definições adicionais que não são tão intuitivas quanto nos exercícios anteriores.

Figura 26 – Sistema Hidráulico



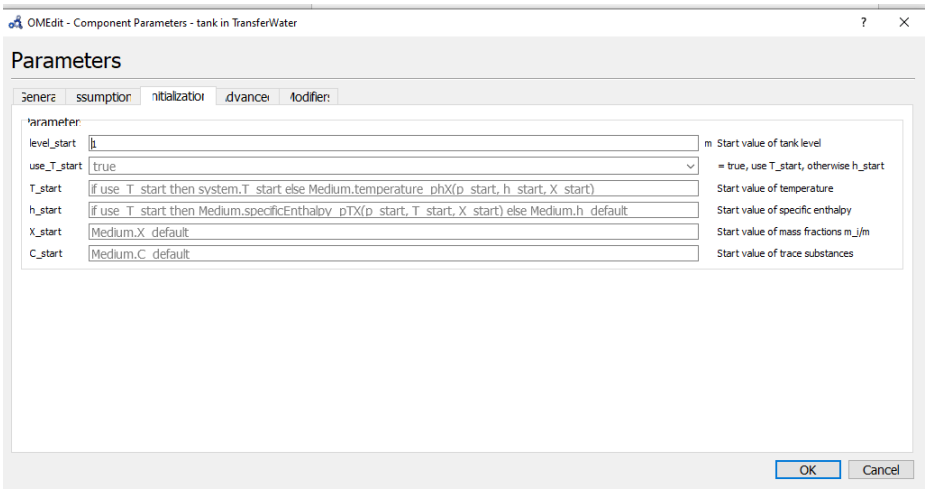
Para este exercício vamos utilizar 3 elementos distintos da biblioteca de fúidos.

- Fluid.Vessels.OpenTank
- Fluid.Pipes.StaticPipe
- Fluid.System

Aqui vamos definir o volume de líquido em cada um dos tanques e avaliar o tempo necessário para que o volume de líquido em ambos se iguale.

Para cada um dos tanques devemos definir o nível inicial de fluido, área de sua seção, altura total. Estes parâmetros podem ser encontrados nas definições dos blocos, nas abas geral e inicialização. Para o tubo definimos o seu comprimento e diâmetro.

Figura 27 – Parâmetros Tanque



Como mencionado, neste tipo de simulação, além dos parâmetros usuais dos blocos, preciamos definir qual fluido estamos utilizando, além de definir o diâmetro das portas de saída dos tanques. Diferente dos parâmetros utilizados até o momento, estes serão definidos na aba de texto do modelo.

Nesta aba vamos definir os valores através de comandos conforme figuras a seguir.

Figura 28 – Difinições do Modelo em Texto

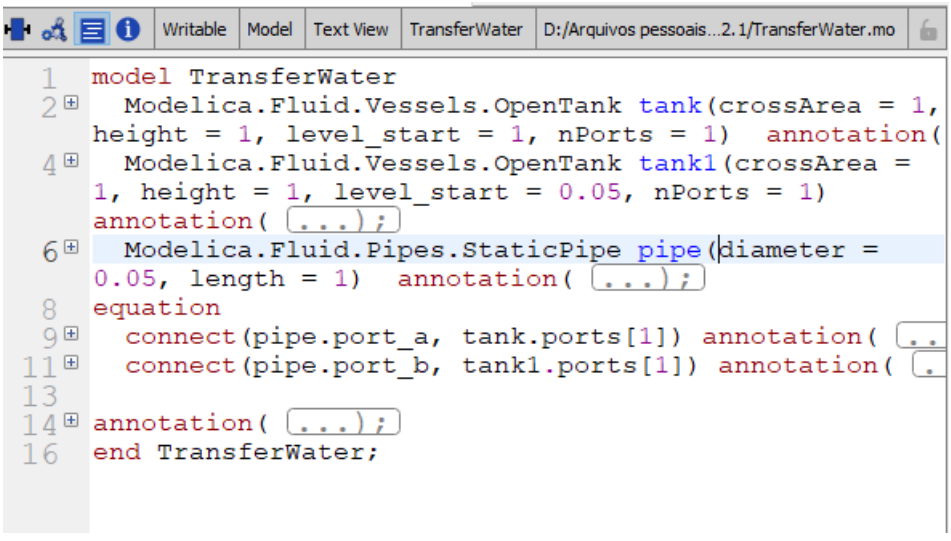


Figura 29 – Difinições do Modelo em Texto Ajustado

Writable | Model | Text View | TransferWater | D:/Arquivos pessoais/TCC/Basic_Pr...eering/CH2/EX2.1/TransferWater.mo

```
model TransferWater
  Modelica.Fluid.Vessels.OpenTank tank(redeclare package Medium =
Modelica.Media.Water.StandardWater, crossArea = 1, height = 1,
level_start = 1, nPorts = 1, portsData =
{Modelica.Fluid.Vessels.BaseClasses.VesselPortsData(diameter =
0.1)})  annotation( ...);
  Modelica.Fluid.Vessels.OpenTank tank1(redeclare package Medium =
Modelica.Media.Water.StandardWater, crossArea = 1, height = 1,
level_start = 0.05, nPorts = 1, portsData =
{Modelica.Fluid.Vessels.BaseClasses.VesselPortsData(diameter =
0.1)})  annotation( ...);
  Modelica.Fluid.Pipes.StaticPipe pipe(redeclare package Medium =
Modelica.Media.Water.StandardWater, diameter = 0.05, length = 1)
annotation( ...);
equation
  connect(pipe.port_a, tank.ports[1]) annotation( ...);
  connect(pipe.port_b, tank1.ports[1]) annotation( ...);

annotation( ...);
end TransferWater;
```

Com os parâmetros que definidos, é possível observar que os tanques terão o mesmo volume após 70 segundos.

Figura 30 – Resultado Sistema Hidráulico