UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS RUSSAS

CURSO DE ENGENHARIA CIVIL

LUIS CARLOS DO NASCIMENTO FILHO

ANÁLISE DA VIABILIDADE DE MODELAGENS NUMÉRICAS EM SOFTWARES OPEN SOURCE

RUSSAS - CE
2022

LUIS CARLOS DO NASCIMENTO FILHO

ANÁLISE DA VIABILIDADE DE MODELAGENS NUMÉRICAS EM SOFTWARES OPEN SOURCE

Trabalho de Conclusão de Curso apresentado ao Programa de Graduação em Engenharia Civil da Universidade Federal do Ceará - Campus Russas.

Orientador: Professor Dr. Esequiel Mesquita

RUSSAS - CE
2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

F498a Filho, Luis Carlos do Nascimento.
Análise da Viabilidade de Modelagens Numéricas em Softwares Open Source / Luis Carlos do Nascimento Filho. – 2022.
51 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Russas, Curso de Curso de Engenharia Civil, Russas, 2022.
Orientação: Prof. Dr. Esequiel Fernandes Teixeira Mesquita.

1. Modelagem Numérica. 2. Software Livre. I. Título.

CDD 620

LUIS CARLOS DO NASCIMENTO FILHO

ANÁLISE DA VIABILIDADE DE MODELAGEM NUMÉRICA EM SOFTWARES OPEN SOURCE

Trabalho de Conclusão de Curso apresentado ao Programa de Graduação em Engenharia Civil da Universidade Federal do Ceará.

Orientador: Prof. Dr. Esequiel Mesquita

Aprovada em:

BANCA EXAMINADORA

_____

Prof. Dr. Esequiel Mesquita (Orientador)
Universidade Federal do Ceará (UFC)

_____

Prof. Ms. Andriele Nascimento de Souza
Universidade Federal do Ceará (UFC)

_____

Prof. Dr. Jerfson Moura Lima
Universidade Federal do Ceará (UFC)

Às dificuldades
psicológicas e financeiras,
às inseguranças,
ao racismo
e a outros fatores
que, embora não devessem,
motivaram-me.

AGRADECIMENTOS

Ao corpo docente e administrativo da Universidade Federal do Ceara - Campus de Russas, diretores, coordenadores, assistentes estudantis, dentre outros que garantem a excelência da instituição pelo trabalho prestado ao longo dos últimos 5 anos.

Ao Prof. Dr. Esequiel Mesquita, pela oportunidade de ingresso no Lareb, orientação em produções técnicas, artigos científicos, modelagens numéricas, trabalho de conclusão de curso e orientações profissionais.

À banca avaliadora pelo tempo dedicado na análise deste documento.

Todos os problemas possuem infinitas soluções.

Prof. Dr. Lindberg Lima

RESUMO

A necessidade de representar matematicamente os fenômenos físicos motivou o desenvolvimento de métodos numéricos que, juntamente com a ascensão da capacidade de processamento computacional, tornou-se algo com grande aplicabilidade em simulações físicas de engenharia. Notou-se o aparecimento das comunidades de desenvolvimento de softwares que logo se transformaram grandes instituições referências na área de simulações físicas. Em decorrência, observou-se fenômenos importantes. Primeiramente, os softwares comerciais possuem interfaces gráficas que permitem uma grande interação com os usuários, facilitando as etapas de construção de uma modelagem numérica. Contudo, ocultando a necessidade do entendimento dos cálculos envolvidos. Além disso, os elevados custos de implementação dos softwares, dado a complexidade do desenvolvimento. Por isso, este documento tem o objetivo de apresentar as etapas da modelagem numérica em *softwares Open Source* avaliando e comparando com aplicações em softwares comerciais. Para isso, foi utilizado a plataforma livre de simulação multifísica OpenFoam para a construção de uma modelagem térmica. Portanto, foi possível descrever as etapas da modelagem numérica comparativamente com modelagens comerciais e notou-se a capacidade da comunidade de desenvolvimento de softwares livres de apresentar ferramentas eficientes.

Palavras-chave: Modelagem Numérica; Software Livre; Análise Térmica**;**

ABSTRACT

The need to mathematically represent physical phenomena motivated the development of numerical methods along with the rise of computational processing capacity has become something with great applicability in physical engineering simulations. So, it was noticed the emergence of software development communities that soon became major reference institutions in the area of physical simulations. Consequently, there were important phenomena. First, commercial software has graphical interfaces that allows a great interaction with users, facilitating the stages of construction of a numerical modeling, however, hiding the need to understand the calculations involved. It is also noted, the high costs of implementing the software, given the complexity of the development. Therefore, this document aims to present the steps of numerical modeling in Open Source software evaluating and comparing with applications in commercial software. The openfoam multiphysics simulation platform was used to build a thermal modeling. Therefore, it was possible to describe the stages of numerical modeling compared to commercial modeling and the ability of the free software development community to present efficient tools was noted.

key words: Numerical Modeling; Open Source; Thermal Analysis;

LISTA DE FIGURAS

LISTA DE GRÁFICOS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

MEF	Métodos dos Elementos Finitos
MVF	Métodos dos Volumes Finitos
VC	Volume de Controle
CAD	Computer-Aided Design
®	Marca Registrada
Φ	Centroide do Volume discretizado no MVF

SUMÁRIO

1 INTRODUÇÃO

A complexidade de calcular os fenômenos físicos de maneira analítica fez com que os engenheiros e matemáticos buscassem novas ferramentas para a simulação de domínios físicos através de métodos numéricos. Esses métodos, com o início datado no século XVII, são essenciais em projetos de engenharia uma vez que conseguem descrever sistemas físicos e são integrados ao cálculo computacional, o que eleva a capacidade de processamento e possibilita a resolução de sistemas de equações de fenômenos complexos (AZEVEDO, 2018).

Dessa forma, as inúmeras possibilidades de simulações fazem a modelagem numérica presente em diversas áreas do conhecimento. Como exemplificação, o estudo desenvolvido por Lotti (2004), na área da odontologia. Ver-se também o estudo de Mesquita (2019), o qual apresenta a modelagem numérica através do Método dos Elementos Finitos (MEF) como uma das ferramentas de análise não destrutiva da Engenharia do Patrimônio.

Dada a importância multidisciplinar dos métodos numéricos, juntamente com o avanço da ciência da computação, surgiram no mercado inúmeros *softwares* que através da aplicação desses métodos simplificam e apresentam resultados de simulações físicas de maneira eficiente. Esses fatos transformaram as instituições desenvolvedoras de software em corporações multinacionais, como os exemplos da ANSYS e COMSOL *Multiphysics*.

Com esse evento de ascensão dos desenvolvedores verificam-se fenômenos importantes no processo de construção de um modelo numérico, sendo, conforme Castro (2019) o elevado preço de se desenvolver um *software* de modelagem computacional o que aumenta o custo das versões no mercado. Outro fenômeno observável é que nem sempre os modelos numéricos possuem considerações que descrevem o problema físico em questão, quando se nota a facilidade de se extrair resultados através de interfaces gráficas e funções pré-configuradas.

Observa-se no trabalho de Silva (2020), a calibragem no processo de modelagem numérica, uma vez que o sistema virtual não descreveu corretamente o sistema real, houve a necessidade de modificar os parâmetros e calibrar o modelo para a obtenção de resultados satisfatórios.

Paralelo ao progresso do mercado de *softwares* de engenharia, com foco na área de simulações numérica, existem as comunidades de softwares *Open Source* que apresentam ferramentas gratuitas e de grande eficiência. Essas ferramentas baseadas em linguagens de programação descrevem as etapas dos métodos numéricos e os apresentam com resultados precisos e aplicáveis, como a plataforma de simulação numérica multifísica OpenFoam.

Assim sendo, o objetivo deste trabalho é realizar uma análise de viabilidade de modelagens numéricas no software livre *OpenFoam* e os *softwares* comerciais, descrevendo os processos de uma análise física termodinâmica. Além disso, busca abordar a modelagem numérica através de um *software* de código aberto demonstrando todas as etapas do processo, desde a geometria até a visualização de resultados.

2 REFERENCIAL TEÓRICO

2.1 Energia e Transferência de Calor

Na natureza, reconhece-se muitas formas de energia, como energia térmica, mecânica, potencial, elétrica, dentre outros tipos. A energia em forma de calor que, segundo ÇENGEL (2009), pode ser entendida como a energia que é causada pela diferença de temperatura entre dois corpos, ocasionando uma transferência de calor. Empiricamente, notou-se que o calor em um sistema sempre flui do meio mais quente para o meio mais frio até que ocorra o equilíbrio térmico.

A Energia Total (E), contida em um sistema, é o somatório de todas as energias presentes. A energia interna do sistema (U) é entendida como energia microscópica, uma vez que está relacionada às partículas contidas no corpo, que é a soma da energia cinética e potencial das moléculas. Dessa forma, quando a transferência de calor e a energia interna não são capazes de vencer a força intermolecular, entende-se como calor sensível. Por outro lado, um sistema que recebe energia suficiente para mudar o estado físico de um corpo, vencendo a força intermolecular, denomina-se calor latente (ÇENGEL, 2009).

Outra definição importante se diz respeito ao calor específico, sendo definido como a energia necessária para elevar em um grau uma unidade de massa de uma substância. Ou seja, existe uma relação direta entre massa e energia que pode ser expressa pela Equação 01, sendo (U), a energia do sistema em kJ. m, a massa em kg. (T), a temperatura em Celsius e o calor específico, (c), em kJ/kg.C (COELHO, 2014).

$$\Delta U = m \cdot c \cdot \Delta T \qquad \text{Eq. 01}$$

A energia pode ser transferida por meio de trabalho (W) e calor expresso por (Q). Contudo, são mecanismos diferentes, uma vez que para haver transferência de energia em forma de calor, essencialmente, deve haver mudança de temperatura, caso contrário, é trabalho. O calor Q é calculado como o somatório da variação do calor Q em relação ao tempo (t). Além disso, faz-se comum nas literaturas a definição de Fluxo de Calor, sendo calor por unidade de área.

Para descrever o balanço energético dentro de um processo existe a primeira lei da termodinâmica. Essa aponta que a energia não pode ser criada nem destruída durante um processo, dessa forma, há apenas mudança de forma. Sendo assim, pode-se estabelecer que a taxa de energia que entra em um sistema é igual a taxa de energia que sai do sistema (ÇENGEL, 2009).

$$E_{ent} - E_{sai} = \Delta E \qquad \text{Eq. 02}$$

Sendo um pouco mais restritivo, em um sistema fechado, típico das análises termodinâmicas, pode ser considerado como um sistema estacionário, isto é, com energias cinéticas e potenciais constantes. Dessa forma, pode-se afirmar que toda a variação da energia interna do sistema se resume a variação da energia em forma de calor fazendo ser válida a relação entre a energia interna U com a primeira lei da termodinâmica (ÇENGEL, 2009).

$$E_{ent} - E_{sai} = \Delta U = m \cdot c \cdot \Delta T \qquad \text{Eq. 03}$$

2.2 Métodos para Modelos Numéricos

Com a praticidade de aplicação em algoritmos computacionais, os métodos numéricos passaram a ser mais utilizados para análises de problemas que fogem do convencional e se tornam inviáveis às resoluções de forma analítica. Embora sejam aplicados em grande escala para análises estruturais, os métodos numéricos possuem aplicabilidades em quase todas as áreas da engenharia. Esses podem ser vistos em simulações mecânicas, estáticas, dinâmicas, térmicas, fluidodinâmicas, eletromagnéticas, dentre outras áreas da engenharia e até em simulações computacionais voltadas para a medicina (TEIXEIRA-DIAS, 2010).

Análises numéricas tratam do desenvolvimento de um modelo para descrever a realidade. O processo inicia com a visualização do real, seguido da simplificação e filtragem de tudo que interfere significativamente em um determinado fenômeno que ocorre sobre um objeto.

Tendo como exemplo uma viga que recebe um carregamento, estando a viga em um ambiente arejado e com pouca variação térmica, notadamente, o efeito da

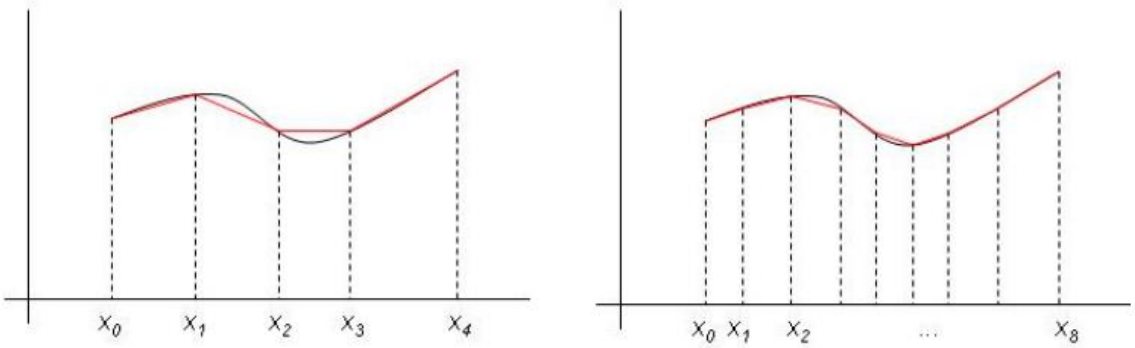
temperatura sobre esse objeto pode ser desconsiderado, o que não poderia ser feito caso o corpo estivesse próximo a uma máquina de aquecimento. As análises procuram um fluxograma que se inicie observando o real, descreva o modelo físico que contenha os fatores significativos, e com base em um modelo matemático, obtenham resultados passíveis de análise que descrevem o fenômeno estudado.

Dentre os métodos numéricos mais utilizados pelos engenheiros e analistas, pode-se citar os métodos dos volumes finitos (MVF) e o Método dos Elementos Finitos (MEF). Os métodos são semelhantes, tratam de processos que subdividem o sistema global em elementos menores, em um processo denominado discretização. Esses elementos são definidos em quantidades suficientes que garantam uma análise com o menor erro possível (GONCALVES, 2007).

O Método dos Elementos Finitos (MEF) consiste em subdividir um domínio de um sistema físico em elementos de quantidade finita, com o objetivo de analisar sistemas complexos que fogem do cálculo analítico. Os elementos discretizados são formados com um nó em cada vértice formando os elementos de diversas formas geométricas, nos casos bidimensionais sendo quadrado ou triângulos, ou tridimensionais com elementos tetraédricos e hexaédricos (HILDEBRANDT, 2021).

O problema unidimensional desenvolvido por Giacchini (2012), exemplifica precisamente como a definição dos elementos se ajusta com a geometria, uma vez que, matematicamente, os elementos e sua vizinhança descrevem a geometria do domínio (Figura 01).

Figura 01: Representação da influência da geometria na concepção matemática do domínio.



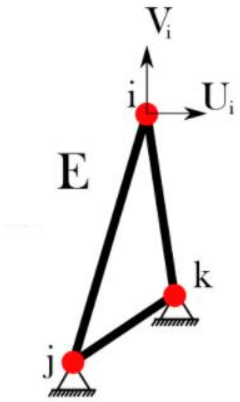
(a) (b)

Fonte: Giacchini, 2012

A modelação por Elementos Finitos consiste em três etapas, pré-processamento, processamento e pós-processamento. No Pré-processamento é realizada a seleção da geometria com suas respectivas características físicas, definição do tipo de malha, de carregamentos e condições de contorno. Normalmente, os processamentos são feitos com a utilização de plataformas de simulações numéricas, sendo *softwares* que unem as etapas e também os sistemas físicos, contando com um resultado contendo análises de tensões e deformações através de gráficos e tabelas, o que determina a etapa de Pós-processamento (TEIXEIRA-DIAS, 2010).

Realizando uma análise estática linear genérica, os nós da malha são vistos como molas que possuem respectivas constantes elásticas. Assim sendo, o elemento da malha passa a representar um conjunto de movimentos lineares com comportamento influenciado por todos os nós. Para uma abordagem simplificada da aplicação do MEF, a Figura 02 expressa a face de um elemento genérico (E) incluído em toda a malha global.

Figura 02: Representação elementar da malha para cálculo por Método dos Elementos Finitos.



Fonte: Desenvolvida pelo Autor

Destacando nó i, nota-se as componentes de força V_i e U_i (Eq. 04). Semelhante ao vetor de força, pode-se definir as componentes de deslocamentos expresso por u_i e v_i (Eq. 05). Dessa forma, o nó i, com suas propriedades mecânicas (geralmente considerada contínua em todo o elemento) e condições de contorno, é solicitado e também solicita os nós vizinhos. A junção das componentes de força (F) e deslocamento (A) de todos os nós conectados ao mesmo elemento de malha definem os vetores de Força e Deslocamento do elemento E (Eq. 06 e 07).

$$F_i = (U_i, V_i) \quad \text{Eq. 04}$$

$$A_i = (u_i, v_i) \quad \text{Eq. 05}$$

$$F_E = (U_i, V_i, U_j, V_j, U_k, V_k \cdots) \quad \text{Eq. 06}$$

$$A_E = (u_i, v_i, u_j, v_j, u_k, v_k \cdots) \quad \text{Eq. 07}$$

Na condição de equilíbrio, através da formulação da Lei de *Hooke*, existe a relação entre F_E e A_E que se dá através da Equação 08. O fator k_E na Equação 08 corresponde à matriz de rigidez elementar, a qual passa pelo processo de assemblagem, dessa vez, para a formação da matriz de rigidez da geometria completa, isto é, a junção de todas as matrizes dos elementos individuais.

$$F_E = k_E \cdot A_E \quad \text{Eq. 08}$$

As soluções da equação acima possibilitam as análises de forças, tensões, deslocamentos dentre outras solicitações da estrutura para determinada condição de contorno e carregamento (TEIXEIRA-DIAS, 2010).

O Método dos Volumes Finitos foi desenvolvido como forma de integrar as equações de conservação (massa, movimento e energia) através dos elementos de volume, chamados volumes de controle (VC). As equações 09, 10 e 11 definem a conservação de massa, movimento e energia em coordenados espaciais e a equação 12 atribui para todo o VC representando-o através da variável Φ que representa o seu centroide. Partindo da discretização do sistema, as equações individuais dos volumes de controle são generalizadas para todo o sistema (GONCALVES, 2007).

$$\frac{\partial}{\partial t}(\rho) + \frac{\partial}{\partial xj}(\rho uj) = 0 \qquad \text{Eq. 09}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_j}\left(\rho u_j u_i\right) = \frac{\partial}{\partial x_i}(P) + \frac{\partial}{\partial x_j}\left(\mu \frac{\partial}{\partial x_j}(u_i)\right) + S^{u_i} \qquad \text{Eq. 10}$$

$$\frac{\partial}{\partial t}(\rho T) + \frac{\partial}{\partial x_j}\left(\rho u_j T\right) = \frac{\partial}{\partial x_j}\left(\frac{k}{C_p}\frac{\partial}{\partial x_j}(T)\right) + S^T \qquad \text{Eq. 11}$$

$$\frac{\partial}{\partial t}(\rho \emptyset) + \frac{\partial}{\partial x_k}(\rho u_k \emptyset) = \frac{\partial}{\partial x_k}\left(T \frac{\partial}{\partial x_k}(\emptyset)\right) + S_\emptyset \qquad \text{Eq. 12}$$

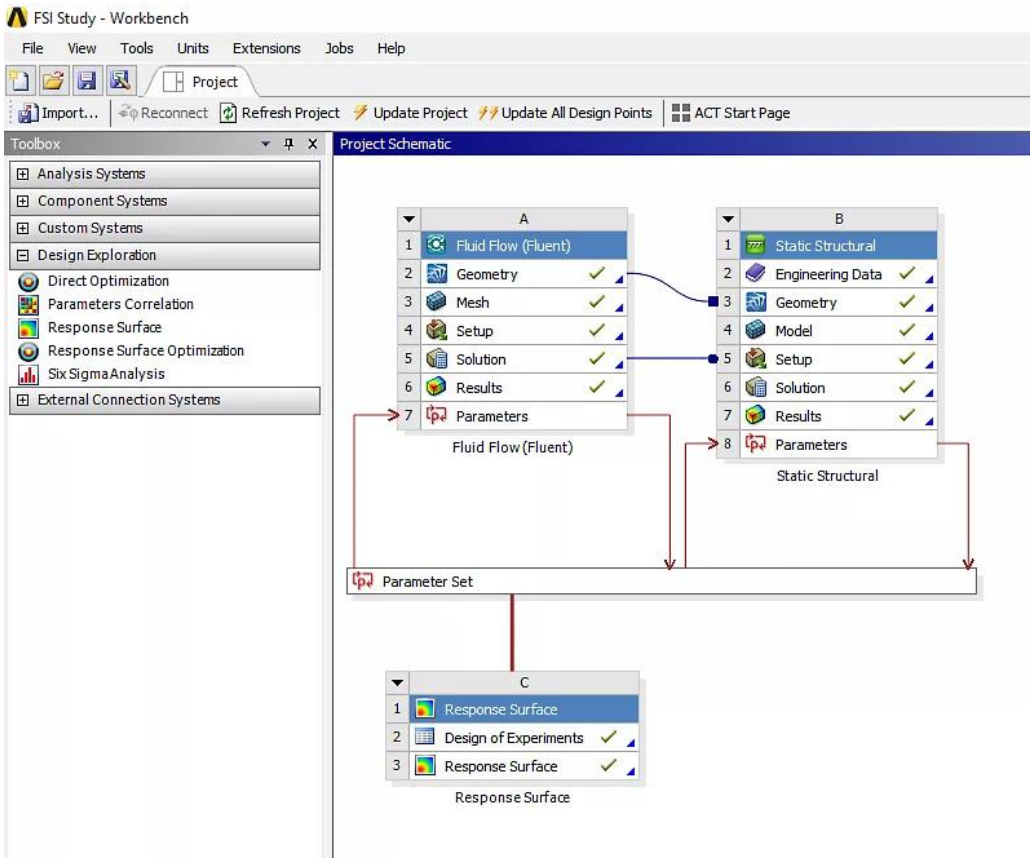
2.3 Softwares de Simulação Numérica

No mercado de simulação numérica os *softwares* comerciais costumam ser desenvolvidos como um ambiente de simulação, isto é, um único programa une diversas outras aplicações que descrevem o processo de modelagem numérica. Normalmente, os *softwares* são organizados dentro do fluxograma de uma análise física como ANSYS *Workbench* e o COMSOL *Multiphysics*. Também pode ser visto *softwares* CAD que contém menus dedicados a análise numérica com funções reduzidas como o Solid Edge.

Outra observação a respeito das plataformas de modelagem numérica é o caso das macro etapas das análises que são o pré-processamento, cálculo numérico e pós- processamento. Assim, é possível notar que os *softwares* trabalham para simplificar todo o processo e deixam claro todas etapas em um processo contínuo e progressivo, seguindo da geometria até a visualização gráfica dos resultados.

A plataforma Ansys *Workbench* é um ambiente de simulação multifísica integrada que está a mais de 50 anos no mercado. Os cálculos são realizados conforme o Método dos Elementos Finitos. O software possui inúmeros sistemas físicos de análise, como estrutural estático e dinâmico, análises fluidodinâmicas, análises térmicas, eletromagnética dentre outros. Na Figura 03, no título da análise *Static Structural* nota-se o fluxograma de modelagem numérica o qual se inicia com as propriedades dos materiais, seguido da geometria, malha de elementos finitos (*mesh*), condições de contorno (*setup*) e as ferramentas de pós-processamento (*Solution / Results*).

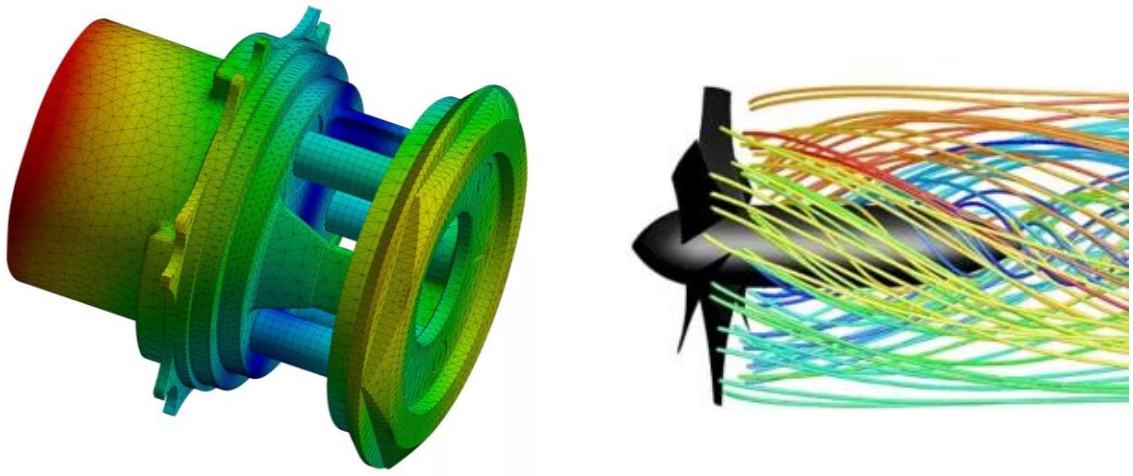
Figura 03: Organização do ambiente de simulação do *software* Ansys Workbench



Fonte: Blog Ansys

O programa possui um banco de dados com propriedades dos materiais para todos os sistemas de análise. Além disso, é possível criar novos materiais, apagar e editar os existentes. Quanto à geometria, o *software* conta com alguns *subsoftwares* internos da plataforma, sendo possível desenvolver geometrias complexas como exemplifica a Figura 04, a) e b) abaixo.

Figura 04: Peças mecânicas detalhando modelagens numéricas de geometrias complexas desenvolvido com Ansys.



a) b)

Fonte: Blog Ansys

2.4 Modelagem Open Source

Nas últimas décadas, observou-se os sistemas operacionais de código aberto em ascensão, principalmente visualizando os avanços da plataforma Linux, a qual é base do sistema *Android* dos *smartphones*. O termo *Open Source* vai muito além do que simplesmente um programa de computador gratuito. Os *softwares*, além de serem gratuitos, possuem seu código fonte compartilhado para modificações conforme desejo dos usuários. Junto com essas empresas, são criados grupos de desenvolvedores, fóruns de pesquisa, salas de *chats* para compartilhamento de arquivos, dentre outras formas de difundir os produtos advindos desse método de desenvolvimento (FUGGETTA, 2002).

No contexto de simulações físicas de engenharia, paralelo ao desenvolvimento de *softwares* que facilitam a aplicação dos métodos numéricos em problemas físicos,

existem comunidades de desenvolvedores que ajudam na criação de *softwares* de código livre para simulações multi físicas.

Na engenharia civil, o *software* de análise de estruturas Ftool foi desenvolvido de forma *Open Source* por acadêmicos brasileiros e se tornou algo comum utilizá-lo no contexto acadêmico, assim como também no desenvolvimento de projetos de engenharia. Ademais, cita-se o FreeCad que se caracteriza como uma ferramenta para desenvolvimento CAD de peças 3D além de conter alguns módulos de modelagem numérica.

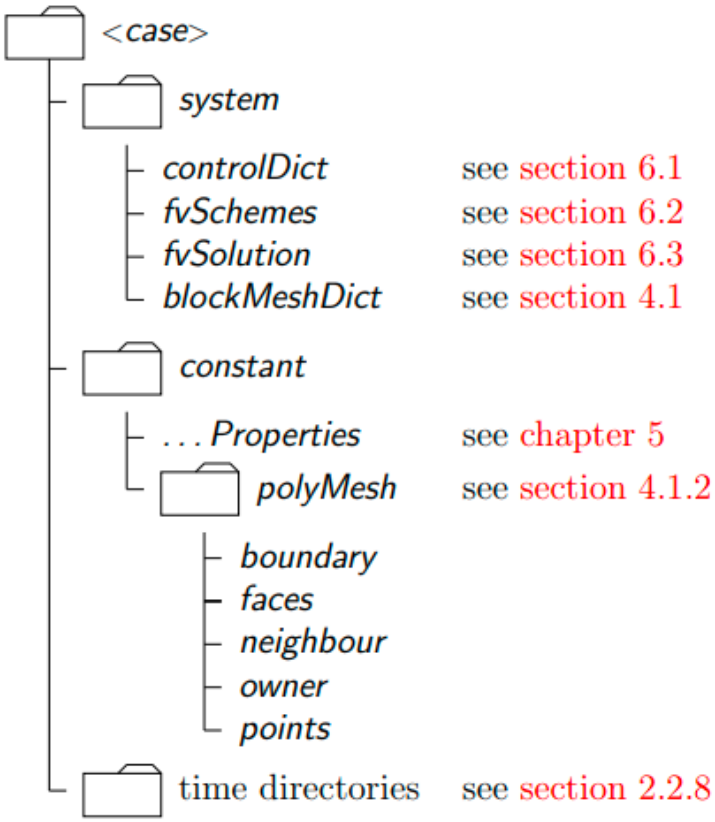
Neste documento, utilizar-se-á o software *Open Source Field Operation and Manipulation* (OpenFoam) que é uma plataforma de simulação multifísica de código aberto e que permite desenvolvimentos de modelos numéricos comprovados. Em paralelo com os *softwares* comerciais, o OpenFoam permite o pré-processamento, a configuração do *solver*, conforme o sistema físico escolhido para aplicação do MVF e ainda o pós-processamento com visualizações gráficas e ferramentas de manipulação de resultados (OPENFOAM, 2021).

A modelagem numérica em código livre também possui passos que descrevem todo o processo de uma análise física. Cada *software* possui uma modelagem de dados diferente, estando no mesmo método, as etapas na busca do resultado tendem a ser semelhantes.

Com a utilização do OpenFoam, o passo-a-passo da simulação se inicia com a escolha do *solver*, sendo que no programa existem inúmeros *solver* com sistemas de análises diferentes, podendo ser citados os solver eletromagnético, análise de *stress*, condução de calor dentre outros. Cada solver possui modelos exemplares e um dos passos iniciais se faz com a escolha do sistema físico conforme o problema a ser analisado.

Posteriormente, faz-se a cópia de um modelo existente, uma vez que todas as configurações são arquivos de textos com *scripts* padronizados, a criação de todos esses diretórios pode ser evitada com a realização da cópia e a alteração dos parâmetros, poupando tempo do analista. A Figura 05 mostra a estrutura geral dos casos presentes no *software*.

Figura 05: Estrutura dos diretórios dos casos exemplares.



Fonte: *User Guide*, OPENFOAM, 2021.

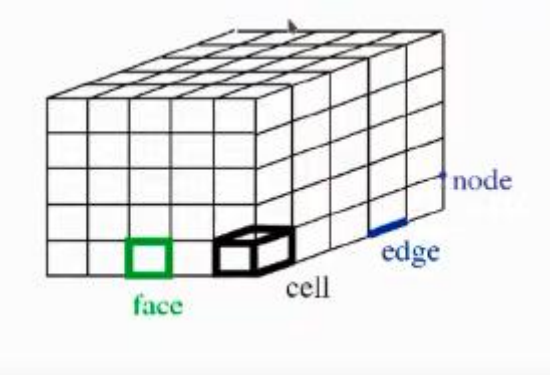
Quanto à configuração de malha, o OpenFoam conta com sub-ferramentas de pré-processamento. Basicamente, algumas ferramentas constroem as malhas e outras apenas convertem arquivos importados de outros *softwares* de pré-processamento.

Neste trabalho, apresenta-se comparativamente a configuração através do *blockMesh*, contudo, pode-se citar também a utilização da ferramenta *PolyMesh*, que permite a geração de malhas complexas e recebe as configurações dos arquivos de importação. Por fim, cita-se os scripts de conversão advindo de *softwares* terceiros que também realizam pré-processamento (OpenFoam, 2021).

O *blockMesh* é uma ferramenta interna de alguns *solvers* e trata-se da construção de malha com células de formatos hexaédricos que são desenvolvidas através de blocos (Figura 06). Para o uso do *blockMesh*, segue com a instalação do *OpenFoam*, no caso, para os sistemas operacionais da Microsoft necessita-se de um *software* para a tradução, uma vez que o programa essencialmente foi desenvolvido

para utilização em Linux. Sendo assim, o *blockMesh* e, consequentemente, o *OpenFoam* foi conseguido através da instalação do projeto blue CFD®-Core da FSD *blueCAPE Lda* que possibilita o desenvolvimento de projetos de simulação numérica através do *OpenFoam* em plataforma do Windows.

Figura 06: Esquematização dos elementos presentes na discretização da malha.



Fonte: Google Imagens

Abaixo, apresenta-se a tela de comando do blue CFD®-Core, na qual foi executado um exemplo tutorial do solver de transferência de calor entre multi regiões. Foi utilizado o comando `$ nano blockMeshDict` para executar o arquivo de texto no qual consta o *script* escrito em linguagem de programação C++ que define a malha da geometria desenvolvida nessa simulação. O comando de definição de vértices e blocos de geometria hexaédrica são todos passados manualmente.

Quadro 01 – Exemplos de simulação nativos do OpenFoam

```
PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/OpenFOAM-4.x/tutorials

$ cd $FOAM_TUTORIALS/heatTransfer/chtMultiRegionFoam/multiRegionHeater/system

PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/OpenFOAM-4.x/tutorials/
heatTransfer/chtMultiRegionFoam/multiRegionHeater/system

$ nano blockMeshDict
```

Fonte: Desenvolvido pelo autor

Quadro 02 – Código fonte de configuração blockMesh

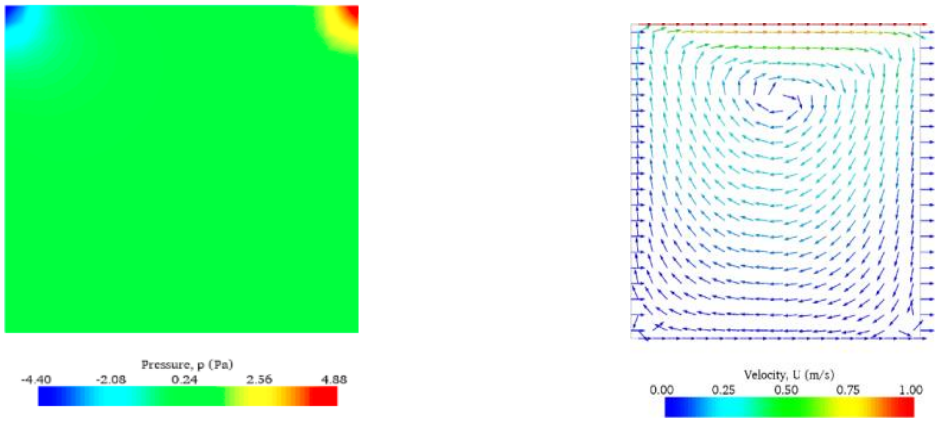
```
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    object      blockMeshDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

convertToMeters 1;

vertices
(
    (-0.1 -0.04  -0.05)
    ( 0.1 -0.04  -0.05)
    ( 0.1  0.04  -0.05)
    (-0.1  0.04  -0.05)
    (-0.1 -0.04   0.05)
    ( 0.1 -0.04   0.05)
    ( 0.1  0.04   0.05)
    (-0.1  0.04   0.05)
);

blocks
(
    hex (0 1 2 3 4 5 6 7) (30 10 10) simpleGrading (1 1 1)
);
```

Fonte: Desenvolvido pelo autor

As próximas etapas das simulações via OpenFoam correspondem à configuração de *scripts* em arquivos de texto. Assim, as especificações das condições de contorno, parâmetros físicos, definição das funções de cálculo e configurações dos *solver* são configurações com diretamente na linguagem de programação.

O pós-processamento do programa se faz com a utilização da ferramenta de visualização gráfica e pós-processamento de dados ParaView, lançada no ano de 2002. Trata-se de uma ferramenta simples e de eficiente aplicabilidade, além disso, permite a leitura de arquivos *scripts* na linguagem de programação *Python*, visualização de escalas gráficas como visto nas Figuras 07 a) e b) (ParaView, 2021).

Figura 07: Exemplos de pós-processamento de dados com o *software* ParaView. Caso Cavity, *solver* icoFoam.



a) b)

Fonte: User Guide, OpenFoam.

Os demais métodos de visualização inclusas no *OpenFoam*, são ferramentas capazes de converter os dados resultantes das análises e os transformar em dados gráficos. Nesse sentido, há comandos que permitem a conversão de dados processados para a extensão VTK através do comando foamToVTK. A ideia por trás da extensão VTK e outros produtos de pós-processamento exportados com o *OpenFoam* podem ser encontrados no guia do usuário do programa. (OPENFOAM, 2021).

3 METODOLOGIA

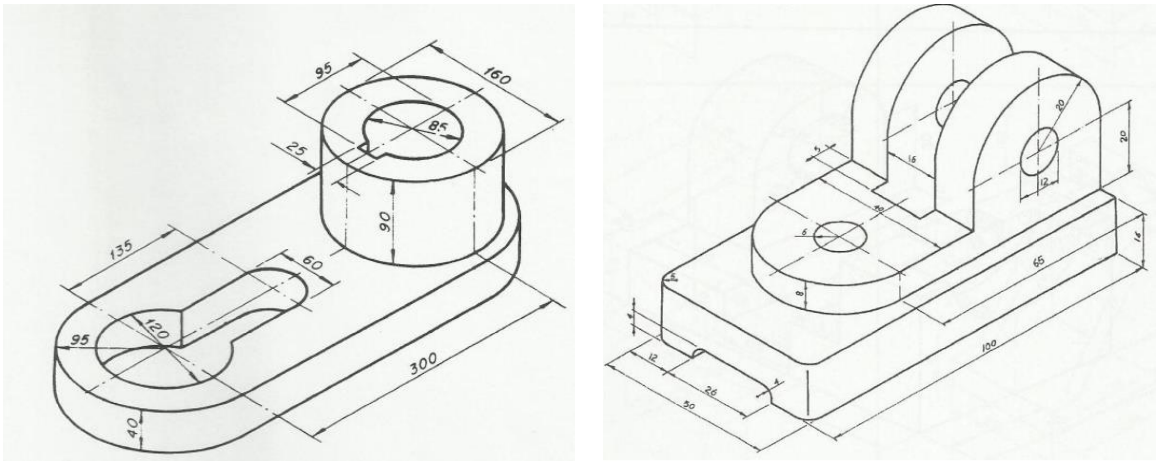
3.1 Pré-processamento

Para a obtenção do comparativo entre as análises numéricas com *softwares* do mercado e modelagens aplicadas em programas de código aberto, foi necessário abordar todas as etapas da modelagem numérica por elementos finitos. Dessa forma, foi verificada a eficiência dessa aplicabilidade desde a definição do modelo físico através de uma geometria, passando pela definição do modelo matemático com a construção da malha, até a análise de resultados com *softwares* livres de pós-processamento.

A definição da geometria se fez através do software de código livre, o *Salome*, que é uma plataforma de *Open Source* de integração para simulação numérica. Esse *software*, além de possibilitar a concepção geométrica, também permitiu o pré-processamento dos modelos numéricos desenvolvido neste trabalho, ou seja, a análise e otimização geométrica, geração e refinamento de malha.

Na análise dos *softwares* de concepção de geometrias buscou-se diversas peças mecânicas, comumente encontradas para exercitar as diversas ferramentas de desenho CAD 3D e 2D, que foram retiradas de apostilas de desenho técnico como podem ser vistas na Figura 08, a) e b). Essas peças permitem verificar a possibilidade de serem criadas nos *softwares*, assim com a eficiência e praticidade da utilização das funções de desenvolvimentos de *sketch*, extrusão, *slice*, contando com a criação de linhas, arcos e formas irregulares.

Figura 08: Figuras retiradas de apostila de desenho técnico para teste de aplicabilidade de *softwares* CAD de código aberto.



a) b)

Fonte: Senger (2013)

O próximo fator a ser analisado nos *softwares Open Source* é a etapa de discretização geométrica em elementos finitos. No caso deste *software* livre, a discretização em volumes finitos. Para isso, neste trabalho foram analisadas duas ferramentas, *blockMesh*, que é ferramenta interna do OpenFoam e também o próprio *software* de pré-processamento Salome que permite exportação de extensões que podem ser importadas no OpenFoam.

A verificação da etapa de discretização no Salome foi aplicada nas configurações para as geometrias desenvolvidas na etapa anterior. As geometrias são pré-configuradas, exportadas para a extensão .unv, copiada para o diretório da análise numérica e convertida para simulação no OpenFoam. O *script* abaixo exemplifica o processo de conversão, é possível notar no diretório da simulação um arquivo .unv a com o comando `ideasUnvToFoam` seguido nome do arquivo, assim, a malha é importada para o problema em questão.

Quadro 03 – Diretórios do caso telhado

```
PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/ofuser-of4/run/telhado

$ ls

0      1200   13800  16200  18000  20400  22800  24600  27000  29400  31200
33600  3600   5400   7200   9600                  system
10200  12000  14400  16800  18600  21000  23400  25200  27600  3000   31800
34200  36000  600    7800   CASO-TELHADO-UNV.unv  VTK
10800  12600  15000  17400  19200  21600  2400   25800  28200  30000  32400
34800  4200   6000   8400   constant
11400  13200  15600  1800   19800  22200  24000  26400  28800  30600  33000
35400  4800   6600   9000   EnSight

PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/ofuser-of4/run/telhado

$ ideasToUnvFoam CASO-TELHADO-UNV.unv
```

Fonte: Desenvolvida pelo autor

No caso do *blockMesh*, utilizou-se os modelos bases inclusos no *software* para exemplificar a aplicação geométrica por *script*. Executando o comando `$FOAM_TUTORIALS`, existem todos os sistemas de análise presentes no *software* além da possibilidade de navegar, editar, e visualizar os arquivos das simulações exemplares.

Quadro 04 –Sistemas físicos de análise do OpenFoam

```
$ cd $FOAM_TUTORIALS

PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/OpenFOAM-4.x/tutorials

$ ls

Allclean  Alltest   Allwmake  combustion    discreteMethods  electromagnetics
heatTransfer    lagrangian  multiphase  stressAnalysis
Allrun    Allwclean  basic     compressible  DNS              financial
incompressible  mesh        resources
```

Fonte: Desenvolvido pelo autor

3.2 Simulação de Transferência de Calor

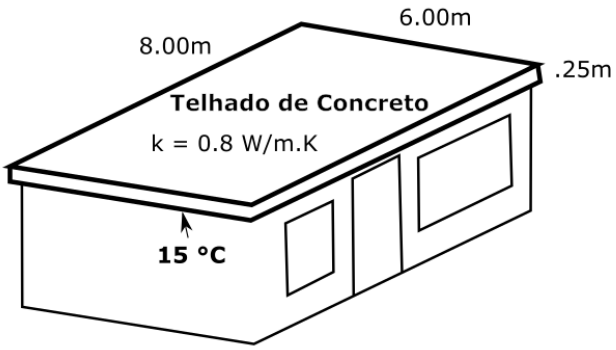
As etapas de especificação dos parâmetros físicos e condições de contorno e ainda todo o pós-processamento foram analisadas através da construção de uma simulação numérica de transferência de calor. Assim, foi possível verificar o processo de configuração do *solver* de cálculo através dos *scripts* de texto.

Recorreu-se a literatura para a busca de um problema aplicável de engenharia, neste caso, escolheu-se o *solver* `laplacianfoam` o qual possui a aplicação da Equação de Laplace para a transferência de calor em um corpo que possui uma variação de temperatura entre as faces extremas. A Equação 13 demonstra a Equação de Laplace para a Transferência de Calor, na qual é possível notar a temperatura como variável independente sendo aplicada no operador laplaciano que retornara o gradiente de temperatura ao das dimensões da função T(x,y,z).

$$\frac{\partial T}{\partial t} - \nabla^2 (D_T \cdot T) = 0 \qquad \text{Eq. 13}$$

A Figura 09 representa uma adaptação do problema de condução de calor que pode ser resolvido analiticamente com a Equação de Fourier para a Condução de Calor, assim como também com a Equação de Laplace utilizada pelo *solver* `laplacianfoam` do *software* OpenFoam. O problema esquematizado corresponde a uma residência com aquecimento elétrico coberta por uma laje maciça de concreto a qual possui 15 °C na face interna, formando um gradiente térmico com a face externa. A condutividade térmica do material é k = 0,8W/mK e a transferência de calor pelo telhado equivale a dQ/dt = 1690 W. A situação em questão trata-se de uma aplicação da Lei de Fourier da Condução Térmica que foi usada aqui como base para a calibragem do modelo.

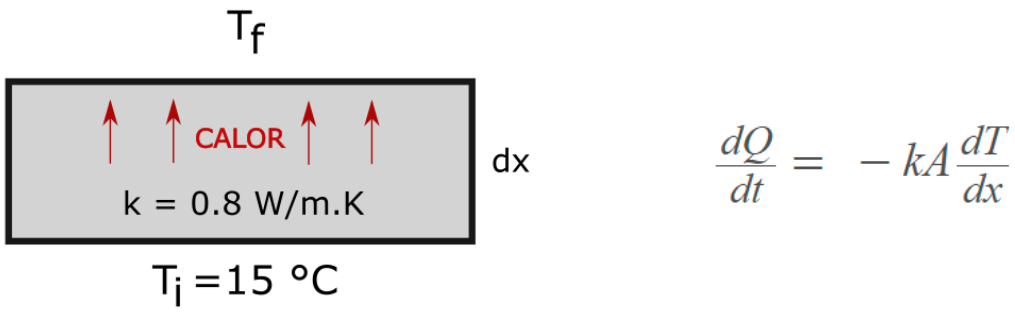
Figura 09: Modelo físico do problema de condução de calor para aplicação em laplacianFoam.



Fonte: Desenvolvido pelo autor

A calibragem do modelo consistiu em calcular o problema analiticamente de acordo com a Lei de Fourier da Condução Térmica, como pode-se observar na equação abaixo presente na Figura 10.

Figura 10: Modelo físico de transferência de calor



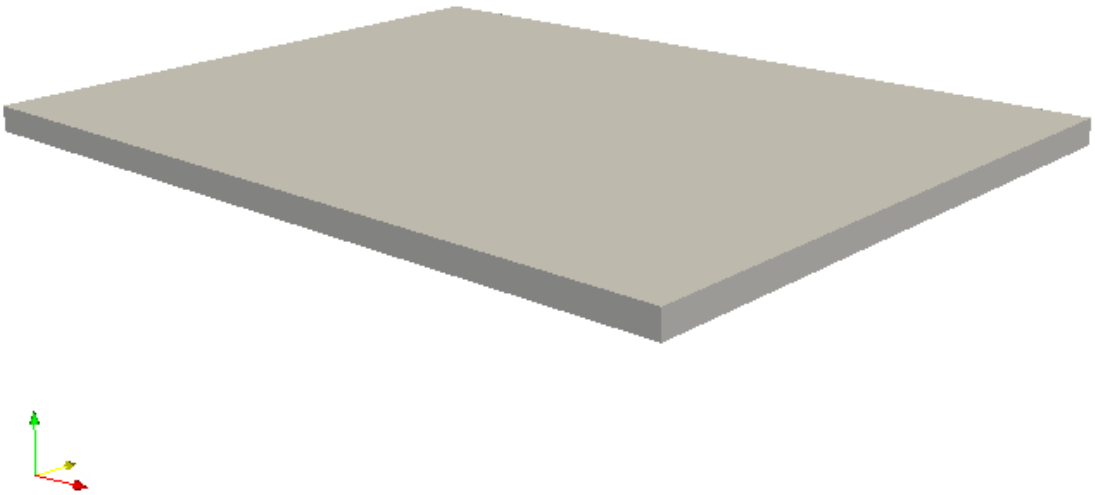
Fonte: Desenvolvido pelo autor

Onde, dQ/dt, é a taxa de variação do calor no tempo em W. A condutividade térmica k, (W/m.K). dT, o gradiente de temperatura do corpo em Celsius(°C) e dx em metros (m) medindo a espessura da peça.

Considerando que o fluxo de calor se dá de dentro para fora do ambiente, o objetivo é descobrir a distribuição na laje da edificação. Sendo assim, quando aplicada a equação da condução e calor de Fourier encontra-se que a temperatura externa equivale a 4,00 °C. A simulação desenvolvida nesse trabalho é um problema simples, contudo, permite passar por todas as etapas da modelagem numérica desenvolvida no *solver* do OpenFoam.

O processo de cálculo inicia-se definindo a geometria no OpenFoam. Para isso, foi desenvolvido o corpo geométrico no Salome, que foi salvo na extensão *.unv*, copiado para o diretório da simulação, para finalmente ser convertido e importado através da linha de comando `ideasUnvToFoam`. A Figura 11 representa a geometria que foi simplificada a uma análise de placa com as dimensões expressas na Figura 09. Em seguida, está presente o processo de conversão e importação para o OpenFoam.

Figura 11: Visualização da geometria do telhado no *software* de pós-processamento ParaView.



Fonte: Desenvolvida pelo autor, *software* ParaView

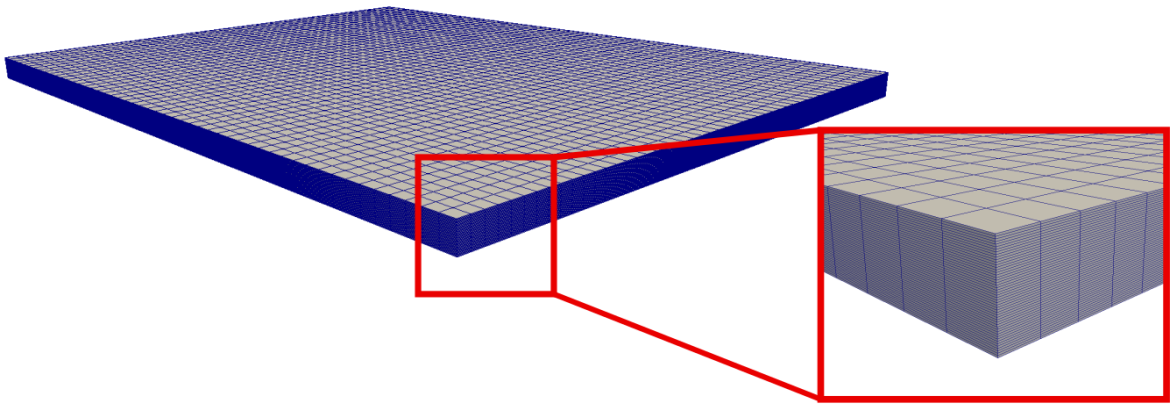
Quadro 05 – Algoritmo de importação da geometria pré-processada

```
PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/ofuser-of4/run/telhado

$ ideasUnvToFoam CASO-TELHADO-UNV.unv
```

Fonte: Desenvolvida pelo autor

O comando `ideasUnvToFoam` corresponde à importação de todo o pré-processamento incluindo a geometria e a malha. Na configuração de malha, utilizou-se múltiplos elementos, em grande maioria, elementos tetraédricos como pode ser visto na Figura 12. Além disso, foi limitado o tamanho máximo do elemento a 15 cm e o mínimo de 10 cm, isso garante uma melhor distribuição na malha visto que a dimensão máxima do telhado corresponde a 25 cm.

Figura 12: Discretização do elemento em volumes finitos visualização em programa de pós-processamento ParaView.



Fonte: Desenvolvido pelo autor

Após a importação da malha discretizada, as configurações são adicionadas a estrutura do OpenFoam, todos os arquivos gerados na importação podem ser acessados o que permite visualizar e editar o *script* base. No diretório no qual está salva a simulação, é possível visualizar a pasta `constant >>> polyMesh` que define os pontos, faces e limites dos elementos, vale ressaltar que todos os arquivos correspondem a arquivos de texto.

Quadro 06 – Diretórios da pasta *constant*

```
PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/ofuser-of4/run/telhado/constant

$ tree
.
├── polyMesh
│   ├── boundary
│   ├── faces
│   ├── neighbour
│   ├── owner
│   └── points
└── transportProperties

1 directory, 6 files
```

Fonte: Desenvolvida pelo autor

Em sequência, foi feita a conexão entre as condições de contorno e a malha definida. É necessário também verificar como foi realizado a divisão da geometria, isto é, as faces que receberam gradientes termais devem ser nomeadas e definidas.

Assim, o *script* é configurado conforme as identificações das faces ou limites presentes na geometria.

No algoritmo abaixo, visualiza-se o arquivo T, presente na pasta de condições de contorno iniciais `$FOAM_RUN/telhado/0/T.txt`. É possível identificar os elementos definidos dentro de `boundaryField`, sendo face_int e face_ext, que descrevem as condições de contorno das faces internas e externas como mostrado na figura seguinte.

O termo `dimensions`, abaixo, corresponde às dimensões físicas presentes na simulação, o quarto termo dentro do parêntese significa que a dimensão de tempo tem um expoente 1, dessa forma, as outra dimensões recebem o expoente 0 simbolizando que essas dimensões não serão consideradas nesta análise. Dentre as outras dimensões, estão presentes dimensões de deslocamento em metros (m), tempo em segundos (s), corrente elétrica em Ampere (A), dentre outras. A tabela 01 apresenta a ordem das unidades dos expoentes expressos no vetor `dimensions [0 0 0 1 0 0 0]`.

Tabela 01: Padronização das dimensões físicas. Unidade no SI.

No.	***Property***	***SI Unity***
1	*Mass*	*kilogram (kg)*
2	*Length*	*metre (m)*
3	*Time*	*second (s)*
4	*Temperature*	*Kelvin (K)*
5	*Quantity*	*kilogram-mole (kgmol)*
6	*Current*	*ampere (A)*
7	*Luminous intensity*	*candela (cd)*

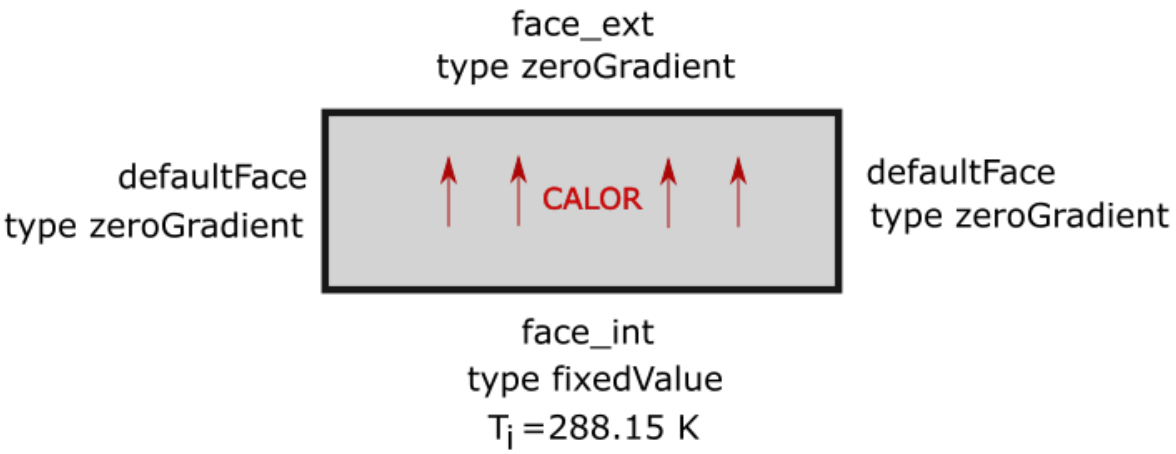
Fonte: User Guide, Open Foam.

Quadro 07 – Arquivo de configurações iniciais $FOAM_RUN/telhado/0/T.txt

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       volScalarField;
    object      T;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

dimensions      [0 0 0 1 0 0 0];

internalField   uniform 278.65;

boundaryField
{
    face_int
    {
        type            fixedValue;
        value           uniform 288.15; // 15.0 C°
    }

    face_ext
    {
        type            fixedValue;
        value           uniform 277.15; //4.0 C°

    }

    defaultFaces
    {
        type            zeroGradient;
    }
}
```

Fonte: Desenvolvida pelo autor

Figura 13: Mapeamento de grupos geométricos para configuração de condições de contorno.



Fonte: Desenvolvida pelo autor

Conforme a aplicação na Equação de Laplace para o cálculo da distribuição de temperatura, verifica-se que as variáveis de entrada correspondem às temperaturas iniciais das faces externas, internas e a difusividade térmica como a que é configurada no arquivo presente no diretório `constant >> transportProperties`.

Quadro 08 – Visualização dos arquivos de configuração polyMesh e transportProperties

```
PC@NOTE MINGW64 OpenFOAM-4.x ~/blueCFD/ofuser-of4/run/telhado/constant

$ ls

polyMesh  transportProperties
```

Fonte: Desenvolvido pelo autor

Segue abaixo o *script* presente no arquivo `transportProperties`.

Quadro 09 – Código fonte do arquivo transportProperties

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "constant";
    object      transportProperties;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

DT              DT [0 2 -1 0 0 0 0] 1.58e-07;

// ************************************************************************* //
```

Fonte: Desenvolvido pelo autor

A variável DT, cujo valor recebido é igual $1.58 \cdot 10^{-7}$, foi calculada conforme a definição da difusividade térmica e essa é uma das constantes presentes na Equação de Laplace (Equação 13). É possível notar as dimensões de comprimento do quadro correspondente ao expoente 2, assim como também a dimensão de tempo com o expoente -1 o qual forma a unidade de medida da difusividade térmica m²/s.

Por fim, no fluxograma da modelagem numérica via OpenFoam as condições de contorno que regem todo o sistema físico são configuradas no arquivo controlDict dentro do diretório `$FOAM_RUN/telhado/system/controlDict.txt`. Neste *script* estão presentes as condições de tempo assim como o número de interações e saídas de resultados desejados. Na simulação de transferência de calor na laje de concreto ver foi utilizado um delta T de 10 horas assim sendo as unidades de tempo em segundos

a variável endTime recebeu um valor de 36000 segundos com a variável `deltaT` sendo passada a cada 600 segundos. No Quadro 10 segue o *script* presente no arquivo `controlDict`.

Os demais arquivos presentes na pasta `system` correspondem à configuração da equação utilizada no modelo, a Equação de Laplace em função da difusividade térmica e temperatura ($f(DT, T)$), método de integração e os métodos de correção estão presentes no arquivo `fvScheme`. Por fim, o arquivo `fvSolution` traz os parâmetros do *solver* e as configurações da tolerância. Esse não foram configurados dado que o arquivo base foi pré editado para o sistema físico atual.

Quadro 10 – Código fonte do arquivo de configuração controlDict

```
FoamFile
{
    version     2.0;
    format      ascii;
    class       dictionary;
    location    "system";
    object      controlDict;
}
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //

application     laplacianFoam;

startFrom       latestTime;

startTime       0;

stopAt          endTime;

endTime         36000;

deltaT          600;

writeControl    runTime;

writeInterval   0.1;

purgeWrite      0;

writeFormat     ascii;

writePrecision  6;

writeCompression off;

timeFormat      general;

timePrecision   6;

runTimeModifiable true;
```

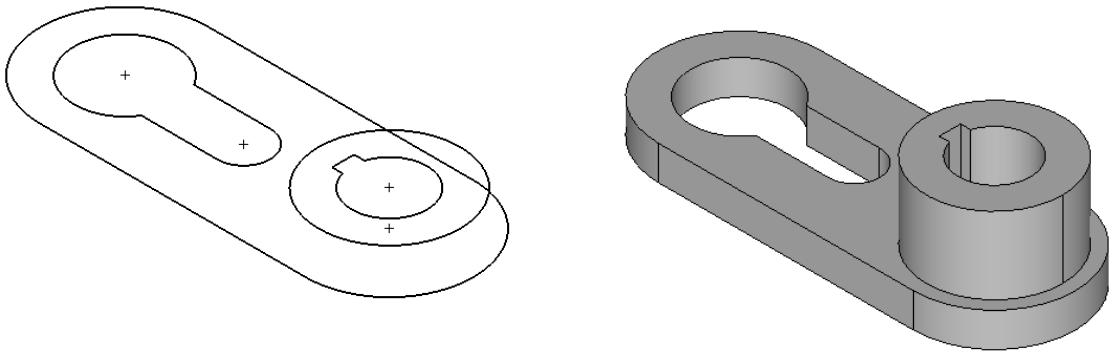
Fonte: Desenvolvido pelo autor

4 RESULTADOS E DISCUSSÕES

Foi verificado o processo de criação da geometria, dessa forma com a utilização do *software Open Source* Salome desenvolveu-se diversas peças mecânicas com ferramentas típicas encontradas nos *softwares* comerciais. Com as peças encontradas em literaturas de desenhos técnicos, a Figura 14, a e b mostram geometrias que exigem o uso de acessórios CAD como *sketches*, ferramentas de construção como *extrude*, *slice*, dentre outros acessórios de desenhos 2D e 3D.

Confere-se que o software Salome atende perfeitamente todas as necessidades para criação de geometrias 2D e 3D e pré-processamento de modelos numéricos. O *software* conta com uma interface gráfica coerente com os *softwares* comerciais, embora sendo gratuito, possui um leque de módulos integrados como *shaper*, para a elaboração de corpos em duas e três dimensões, *Geometry*, conecta a geometria com a pré configuração para receber a discretização dos elementos, e até o módulo *Mesh*, que é em se um conjunto de ferramentas para a criação da malha. Ademais, o *software* possui interações com outras extensões universais como .STEP e .IGES.

Figura 14: Desenvolvimento de geometria no *software* CAD Open Source Salome.



Fonte: Desenvolvida pelo autor

Com relação à discretização da geometria em elementos, foram testados o blockMesh, interno do OpenFoam em alguns *solvers* e o software Salome, que permite criar e exportar a configuração da malha juntamente com a geometria.

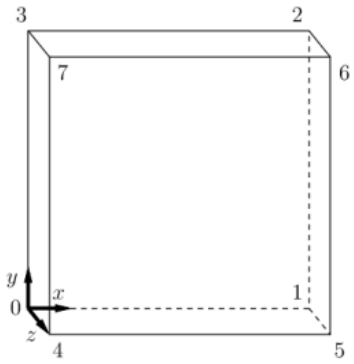
Analisando o blockMesh a partir de um modelo exemplo do programa, verifica-se a maneira manual de desenvolver um corpo geométrico. No exemplo, todos os

vértices do corpo são descritos em linhas de código formando as coordenadas (Figura 15). Os vértices de 0 a 7 correspondem ao formato do paralelepípedo, geometria da simulação. A função vértice define os pontos que contornam a geometria para que, em seguida, seja feita a construção dos blocos e definição dos elementos hexaédricos.

Esse método de construção de malha é pouco viável quando se imagina geometrias complexas, uma vez que as simulações, com a capacidade de processamento existente no mercado, tendem a ter um maior número de detalhes. Dessa forma, criar geometrias por linhas de código se faz inviável.

Outro fator a ser citado diz respeito à definição da discretização da malha. A ferramenta blockMesh se limita a construção de malha hexaédrica, e existindo essa limitação, filtra-se a infinitas possibilidades de geometrias possíveis fazendo com que uma análise numérica venha a ser prejudicada com o uso de elementos que deforme a representação geométrica original.

Figura 15: Construção de geometria com blockMesh, caso Cavity, icoFoam, Software OpenFoam.



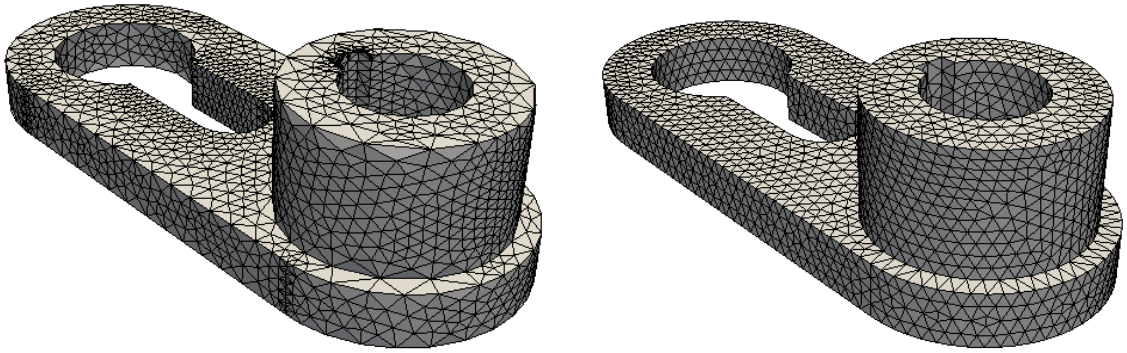

```
vertices
(
    (0 0 0)
    (1 0 0)
    (1 1 0)
    (0 1 0)
    (0 0 0.1)
    (1 0 0.1)
    (1 1 0.1)
    (0 1 0.1)
);
```

Fonte: User Guide OpenFoam.

Por outro lado, quando se utiliza o programa de pré-processamento Salome observa-se os passos para a construção da malha bem definidos através de interfaces gráficas. Abaixo, vê-se a configuração da malha para as geometrias, ou seja, é possível visualizar a malha antes do processamento, editá-las e configurá-las. Observa-se na Figura 16 a), que a malha foi configurada, e logo após, reconfigurada na Figura 16 b) com métodos de refinamento. Ademais, o *software* possui diversas ferramentas de configuração de malha para elementos distintos. Dessa forma, vê-se que a utilização do Salome para pré-processamento, incluindo a discretização da

malha, é uma alternativa na construção de modelos numéricos complexos, e além de tudo, é notório a eficiência e aplicabilidade da ferramenta.

Figura 16: Discretização da malha via Salome com importação no OpenFoam



a) b)

Fonte: Desenvolvida pelo autor

Referente à configuração dos parâmetros físicos e condições de contorno, não há muitas diferenças em relação aos *softwares* comerciais, basicamente, a maior dificuldade é compreender os locais nos quais eles são configurados. Nos *softwares* comerciais, existem interfaces gráficas que permitem configurar os parâmetros de propriedades de material, condições físicas, etc. No caso *Open Source*, as interfaces são substituídas por arquivos de textos programáveis, o processo se faz semelhante aos demais *softwares*.

Na tabela abaixo, há um comparativo entre a hierarquia de diversos *solvers* do OpenFoam. Na Tabela 02, nota-se que os *solvers* tendem a ter arquivos semelhantes padronizados, sendo o diretório `0`, para condições iniciais e as variáveis do sistema físico, a pasta `constant`, na qual existem as propriedades de materiais e configurações de malha, dentre outros exemplos que podem ser observados padrões.

O sistema `heatTransfer` tende a ter uma maior quantidade de diretório, o que o diferencia dos outros sistemas de análise, contudo, isso se deve ao fato de ser um *solver* que possui geometrias sólidas e fluidas, mesmo assim, nota-se que o processo de configuração continua padronizado.

Tabela 02: Comparação da árvore de diretórios de diferentes sistemas de físicos.

Basic: laplacianFoam	**stressAnalysis: solidDisplacementFoam**	**heatTransfer: chtMultiRegionFoam**

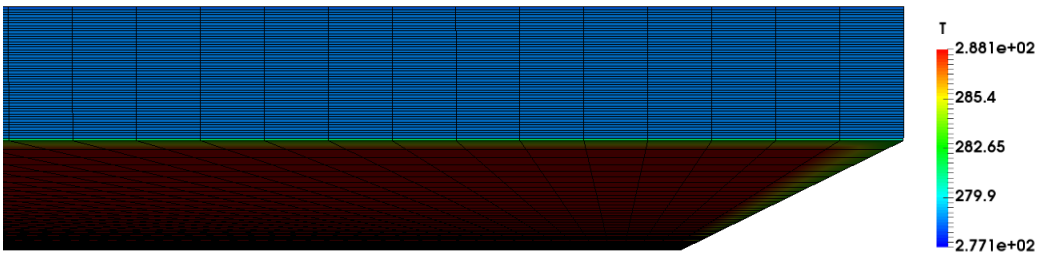
`$ tree`
`.`
`├── 0`
`│ └── T`
`├── Allclean`
`├── Allrun`
`├── constant`
`│ └── transportProp.`
`├── flange.ans`
`└── system`
`├── controlDict`
`├── fvSchemes`
`└── fvSolution`
 	
`3 directories, 8 files`
	`$ tree`
	`.`
	`├── 0`
	`│ ├── D`
	`│ └── T`
	`├── Allclean`
	`├── Allrun`
	`├── constant`
	`│ ├──mechanicalProp.`
	`│ └── thermalProp.`
	`└── system`
	`├── blockMeshDict`
	`├── controlDict`
	`├── fvSchemes`
	`├── fvSolution`
	`└── singleGraph`
	
	`3 directories, 11 files`
		`$ tree`
		`.`
		`├── Allclean`
		`├── Allrun`
		`├── cavity`
		`│ ├── 0`
		`│ │ ├── p`
		`│ │ └── U`
		`│ ├── constant`
		`│ │ └──`
		`transportProp.`
		`│ └── system`
		`│ ├──`
		`blockMeshDict`
		`│ ├── controlDict`
		`│ ├── fvSchemes`
		`│ └── fvSolution`
		`├── cavityClipped`
		`│ ├── 0`
		`│ │ ├── p`
		`│ │ └── U`
		`│ ├── constant`
		`│ │ └──`
		`transportProp.`
		`│ └── system`
		`│ ├──`
		`blockMeshDict`
		`│ ├── controlDict`
		`│ ├── fvSchemes`
		`│ ├── fvSolution`
		`│ └──`
		`mapFieldsDict`
		`└── cavityGrade`
		`├── 0`
		`│ ├── p`
		`│ └── U`
		`├── constant`
		`│ └──`
		`transportProp.`
		`└── system`
		`├──`
		`blockMeshDict`
		`├── controlDict`
		`├── fvSchemes`
		`├── fvSolution`
		`└──`
		`mapFieldsDict`
		
		`12 directories, 25 files`

Fonte: Desenvolvido pelo autor

No caso da condução de temperatura na laje de concreto, o pós-processamento comparativamente com os softwares comerciais. Esta etapa da modelagem numérica é mais um exemplo de *softwares Open Source* que possuem ferramentas eficazes e cumprem com as necessidades de visualização gráfica dos resultados advindos da etapa anterior, o processamento.

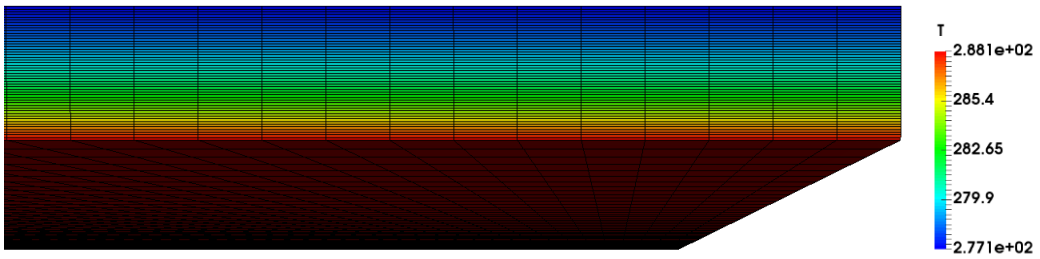
Abaixo, as Figuras 17 e 18 foram retiradas dos resultados do processamento da Equação de Laplace para a condução de calor através de uma laje de concreto com o *software* ParaView. Com a interface gráfica do *software ParaView* foi possível mostra os resultados de temperatura assim como a discretização da malha plotados na geometria 3D.

Figura 17: Distribuição de temperatura no tempo t = 0s.



Fonte: Desenvolvido pelo autor

Figura 18: Distribuição de temperatura no tempo t = 36000s (10hrs)



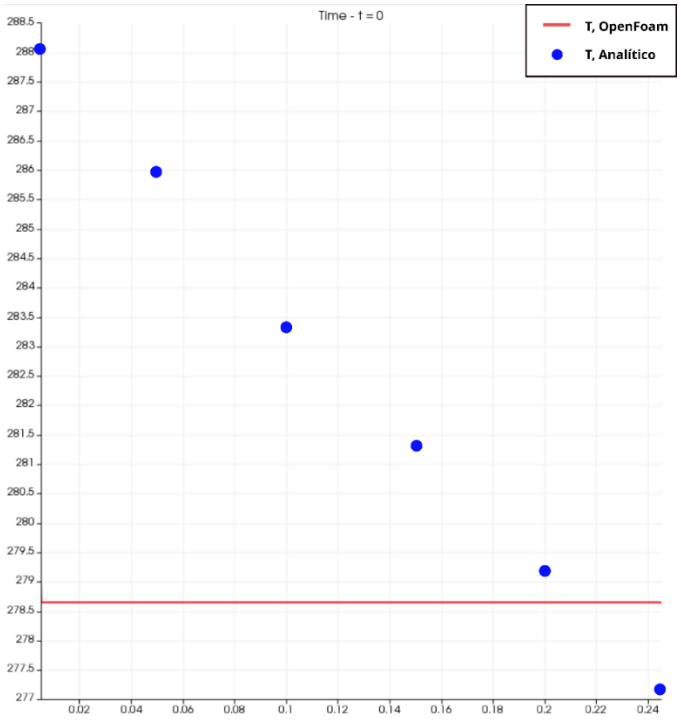
Fonte: Desenvolvida pelo autor

Na escala gráfica linear apresentada na Figura 19, nota-se um ponto de mudança da taxa de variação do gradiente de temperatura. Esse fato se deve a presença de uma temperatura ambiente que contorna os elementos nas suas condições iniciais, pois a equação de Laplace corresponde a uma distribuição de temperatura dado que exista diferença térmica entre pontos distintos do corpo.

Comparativamente ao resultado, o cálculo através da equação de Fourier foi adicionado aos gráficos. Dessa forma, vê-se que o cálculo através da equação de Laplace para condução de calor tende a descrever a distribuição de temperatura ao longo da estrutura de concreto.

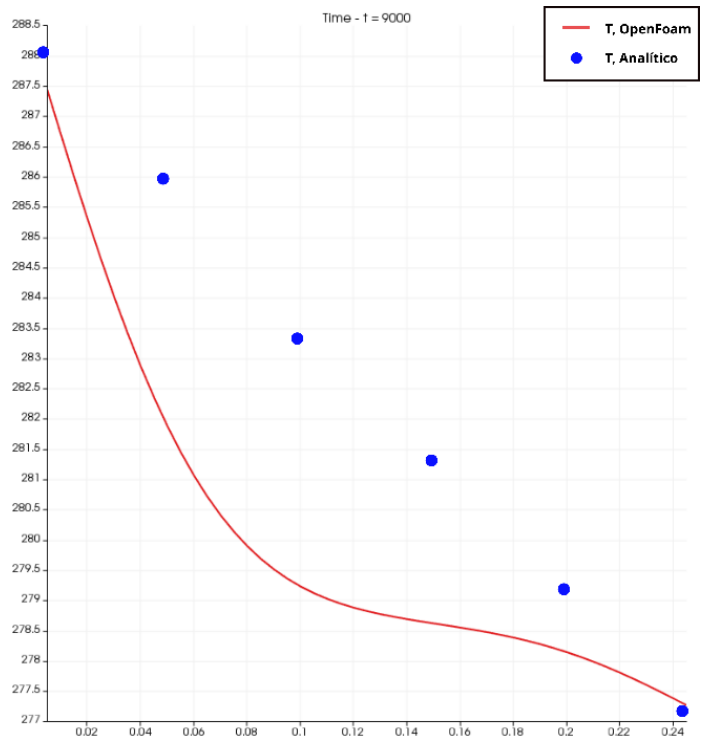
Nos Gráficos 01 a 04 os valores de temperatura tendem a convergir ao valor analítico. De 0 segundos a 36000 segundos (10 hrs) os valores convergem para uma distribuição linear. Indica-se buscar mudanças nas considerações de cálculo, principalmente nas configurações do solver apontando uma maior a diferença de tempo.

Gráfico 01: Temperatura T = 278.65 K em função da distância dz da laje de concreto no tempo t = 0s.



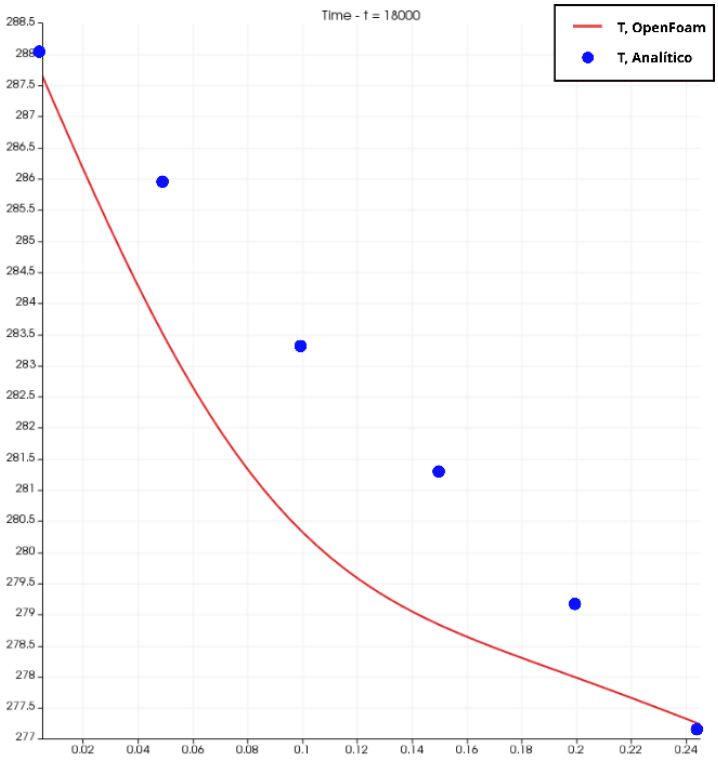
Fonte: Desenvolvido pelo Autor

Gráfico 02: Temperatura em função da distância dz da laje de concreto no tempo t = 9000s.



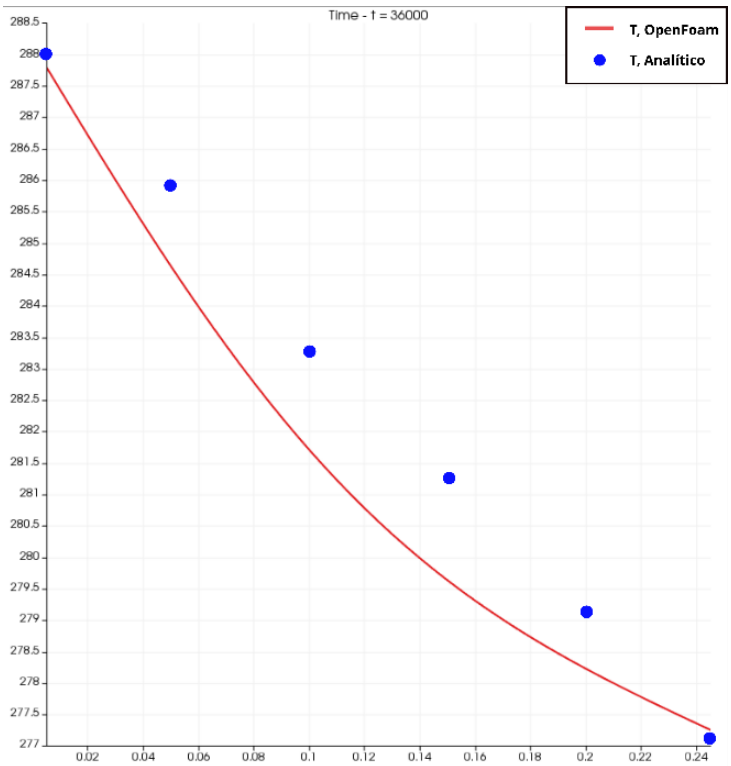
Fonte: Desenvolvido pelo Autor

Gráfico 03: Temperatura em função da distância dz da laje de concreto no tempo t = 18000s.



Fonte: Desenvolvido pelo Autor

Gráfico 04: Temperatura em função da distância dz da laje de concreto no tempo t = 36000s.



Fonte: Desenvolvido pelo Autor

5 CONCLUSÃO

Contudo, a modelagem numérica por meio de *software Open Source* é uma alternativa viável para a elaboração de simulações físicas com resultados precisos. Os programas analisados quanto à parte de pré-processamento possuem interfaces gráficas com uma usabilidade que pode ser comparada aos *softwares* comerciais de análise numérica.

Os geradores de malha, que são configurados através de linha de comando, tendem a dificultar a construção de geometrias complexas constando como um ponto negativo desta metodologia. As especificações de contorno e as propriedades do sistema, que também são desenvolvidas via linha de comando, não alteram a usabilidade, uma vez que são parâmetros descritos em poucas linhas. Quanto às equações do sistema, é pouco intuitivo a visualização de resultados derivados da temperatura como a deformação térmica. No que diz respeito ao pós-processamento, a ferramenta ParaView foi a única testada no modelo numérico e atende perfeitamente a visualização gráfica.

Portanto, aponta-se que o documento contribui para a apresentação de uma alternativa de desenvolvimento de modelos numéricos através de programa de código livre. Foi demonstrado, com teor comparativo, as etapas de uma simulação física através de métodos numéricos, e assim, aponta-se a existência uma metodologia eficiente quando apresentados paralelamente *softwares Open Source* e programas comercias.

Embora algumas ferramentas analisadas ainda se faz pouco aplicáveis, espera-se o desenvolvimento de novas ferramentas e novas versões com o apoio das comunidades de desenvolvedores que seguem contribuindo para um conhecimento mais democrático.

REFERÊNCIAS

AZEVEDO, Gabriela Moreira; PENNA, Samuel Silva; DA SILVA, Ramon Pereira. **PARALELIZAÇÃO DA MONTAGEM DE SISTEMAS DE EQUAÇÕES PARA O MÉTODO DOS ELEMENTOS FINITOS**. XII SIMEC, Simpósio de Mecânica Computacional. UFES, Espírito Santo, 2018.

CASTRO, Antônio Carlos Gardel de Freitas. **O cenário atual da tecnologia Bim em empresas de arquitetura, engenharia e construção: um estudo de caso na cidade de Pau dos Ferros-RN**. UFERSA, Mossoró, 2019.

ÇENGEL, Yunus A.; GHAJAR, Afshin J. Tranferência de Calor e Massa. **AMGH editora**, McGraw Hill, 2009.

COELHO, N. A. et al. **Influência das propriedades térmicas do concreto massa na análise da temperatura em estruturas de grandes dimensões.** 10th World Congress on Computational Mechanics. São Paulo, 2012.

FUGGETTA, Alfonso. Open source software—an evaluation**. Journal of Systems and software**, v. 66, n. 1, p. 77-90, 2003.

GIACCHINI, Breno Loureiro. **Uma breve introdução ao Método dos Elementos Finitos. Departamento de Matemática**: Instituto de Ciências Exatas, Universidade Federal de Minas Gerais, 2012.

GONÇALVES, Nelson Daniel Ferreira et al. **Método dos Volumes Finitos em Malhas não estruturadas**. 2007. Tese (Mestre em Engenharia Matemática) – Faculdade de Ciências da Universidade do Porto, Porto, Portugal, 2007.

HILDEBRANDT, Clara Costa. **Estudo da Influência de Parâmetros Geométricos na Pressão de Falha de Dutos Corroídos Utilizando o Método dos Elementos Finitos.** 2021. Trabalho de Conclusão (Graduação em Engenharia Mecânica). Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2021.

LOTTI, Raquel S. et al. Aplicabilidade científica do método dos elementos finitos. **Revista Dental Press de Ortodontia e Ortopedia Facial**, Maringá, v. 11, n. 2, p. 35-43, 2006.

MESQUITA, Esequiel; **ENGENHARIA DO PATRIMÔNIO**. Ithalia, Curitiba, 2020.

OPEN FOAM, The Open Source CFD Toolbox. **User Guide**. Version v2112. 2021.

PARAVIEW, Developers. **ParaView Users Guide Documentation.** Release 5.10.0. 2021.

SENGER, Valdir Airton. **CADERNO EXERCÍCIOS DE DESENHOR TECNICO MECÂNICO**. Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande de Sul, IFRS. Ibirubá, 2013.

SILVA, Rodrigo Amado Garcia et al. MODELAGEM NUMÉRICA DE EROSÃO DA PRAIA DE PIRATININGA, NITERÓI–RJ POR AÇÃO DE ONDAS DE TEMPESTADE. **ABR Hidro**, Campinas - SP, 2020.

TEIXEIRA-DIAS, Filipe et al. Método dos Elementos Finitos: Técnicas de Simulação Numérica em Engenharia**. ETEP**, 1ª ed., 2010.

VÄLIKANGAS, Turo; KARVINEN, Reijo. Conjugated heat transfer simulation of a fin-and-tube heat exchanger. **Heat Transfer Engineering**, Tampere University of Technology, Tampere, Finland. v. 39, n. 13-14, p. 1192-1200, 2018.