

Uma Ferramenta Computacional para Simulação de Escoamento Pulsátil em Modelos de Árvores Arteriais 1D

Igor Pires dos Santos

igor.pires@ice.ufjf.br

Orientadores: Rafael Alves Bonfim de Queiroz
& Ruy Freitas Reis



Programa de Pós-Graduação em Modelagem Computacional
Universidade Federal de Juiz de Fora

20 de setembro de 2021

Introdução

Modelo Matemático

Modelo Computacional

Ferramenta Computacional

InGU

IGU

Janelas

Resultados

Dissertação

Cronograma

- ▶ A construção de modelos de árvores arteriais é importante para a realização de estudos hemodinâmicos. Neste trabalho, apresentam-se:
 - ▶ (i) um esquema analítico para o cálculo das características locais das ondas de fluxo e pressão em modelos de árvores arteriais 1D
 - ▶ (ii) um ambiente computacional desenvolvido para a simulação e visualização dos resultados no tocante à construção de modelos e estudos hemodinâmicos. Os resultados obtidos neste trabalho estão condizentes com dados numéricos relatados na literatura.

- ▶ A propagação de ondas em um tubo é governada pela equação de onda para a pressão $p(x, t)$ e volume de fluxo $q(x, t)$, ambas sendo funções no tempo t e coordenada axial x ao longo do tubo.

$$\frac{\partial q}{\partial t} = -cY \frac{\partial p}{\partial x} \quad (1)$$

$$\frac{\partial p}{\partial t} = -\frac{c}{Y} \frac{\partial p}{\partial x} \quad (2)$$

- ▶ Para uma onda harmônica simples Eq.1 e Eq.2 ficam na forma:

$$p = \bar{p}_0 \exp\{i\omega(t - x/c)\} + R\bar{p}_0 \exp\{i\omega(t - 2L/c + x/c)\} \quad (3)$$

$$q = Y(\bar{p}_0 \exp\{i\omega(t - x/c)\} - R\bar{p}_0 \exp\{i\omega(t - 2L/c + x/c)\}) \quad (4)$$

- ▶ Para definir os valores do coeficiente de reflexão R e pressão média \bar{p}_0 de cada segmento foi utilizado o modelo matemático descrito por Duan & Zamir (1995).
- ▶ Este modelo matemático possui uma complexidade pequena e é capaz de ser iterado muito rapidamente para modelos geométricos numerosos.

- ▶ Inicialmente se definem as propriedades características de cada segmento :
- ▶ Espessura da parede (h):
- ▶

$$h = 0.1 \times r \quad (5)$$

- ▶ Velocidade de Onda (c):
- ▶

$$c = \sqrt{\frac{Eh}{\rho 2r}} \quad (6)$$

Modelo Matemático (cont.)

- ▶ Velocidade angular (ω):



$$\omega = 2\pi f \quad (7)$$

- ▶ Beta (β):



$$\beta = \omega \frac{L}{c} \quad (8)$$

- ▶ Admitância Característica (Y):



$$Y = \frac{\pi r^2}{\rho c} \quad (9)$$

► **Caso 2:** Ângulo de fase

► Módulo de Young (E_c):

►

$$E_c = \|E_c\| \exp\{i\phi\} \quad (10)$$

►

► Ângulo de Fase (ϕ):

►

$$\phi = \phi_0(1 - \exp\{-w\}) \quad (11)$$

- ▶ **Caso 3:** Viscoso

- ▶ Velocidade de Onda Viscosa (c_v):

- ▶

$$c_v = c\sqrt{\epsilon} \quad (12)$$

- ▶

- ▶ Admitância (Y_v):

- ▶

$$Y_v = Y\sqrt{\epsilon} \quad (13)$$

- ▶ Alpha (α):

- ▶

$$\alpha = R \sqrt{\frac{\omega \rho}{\mu}} \quad (14)$$

- ▶ Fator Viscoso (ϵ):

- ▶

$$\epsilon = 1 - F_{10}(\alpha) \quad (15)$$

- ▶ Função auxiliar do Fator Viscoso (F_{10}):

- ▶

$$F_{10}(\alpha) = \frac{2.0}{\alpha \sqrt{i}} \left(1.0 + \frac{1.0}{2.0\alpha} \right), \quad (16)$$

- ▶ **Reflection Coefficient «complex» e Admittance «complex»**

- ▶ Se folha $R = 0$, senão $R = \frac{Y - (Ye_r + Ye_l)}{Y + (Ye_r + Ye_l)}$.
- ▶ Se folha $Ye = Y$, senão $Ye = Y * \frac{(1 - R \exp\{-2i\beta\})}{(1 + R \exp\{-2i\beta\})}$.

- ▶ **Medium Pressure «complex»**

- ▶ Se raiz $\bar{P} = \bar{P}_0$, senão $\bar{P} = \bar{P}_f * \frac{((1 + R_f) \exp\{-i\beta_f\})}{(1 + R \exp\{-2i\beta\})}$.

► **Pressure «complex»**

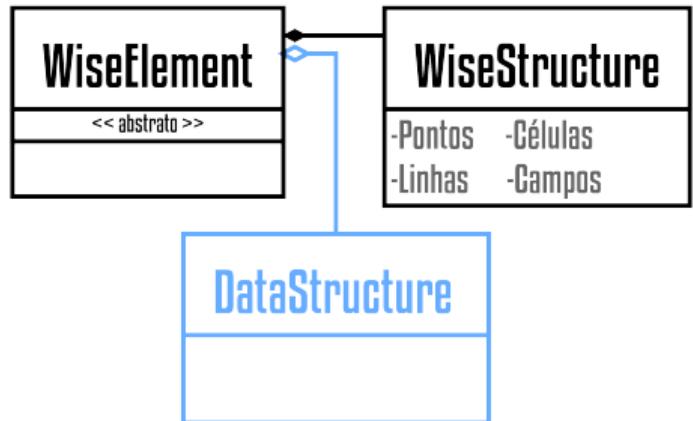
$$\text{► } P = \bar{P} * (\exp\{-i\beta X\} + R \exp\{-i2\beta\} \exp\{i\beta X\}).$$

► **Flow «complex»**

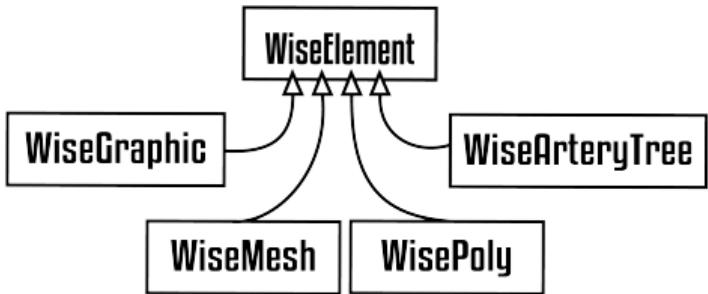
$$\text{► } Q = M \bar{P} * (\exp\{-i\beta X\} - R \exp\{-i2\beta\} \exp\{i\beta X\}).$$

$$\text{► } M = \frac{Y}{Y_r}$$

- ▶ Um elemento inteligente (*WiseElement*) possui duas estruturas básicas. A *WiseStructure* que representa os dados dispostos no padrão *VTK*(*Visualization Toolkit*) e *DataStructure* que representa os dados abstratos disponíveis na estrutura.

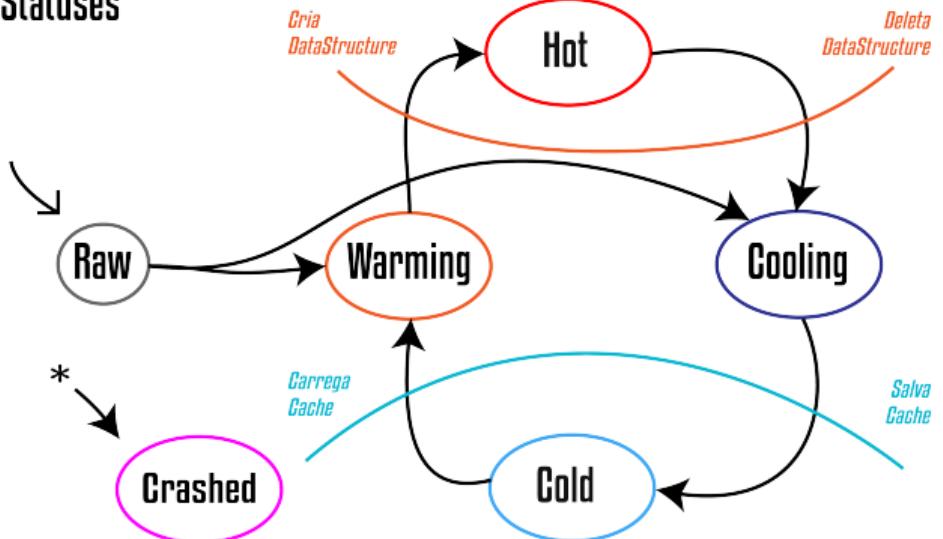


- ▶ Todos os elementos inteligentes recebem a estrutura básica *WiseStructure* por herança e requer por meio de funções virtuais a definição de métodos que possibilitem a manipulação dos dados abstratos.

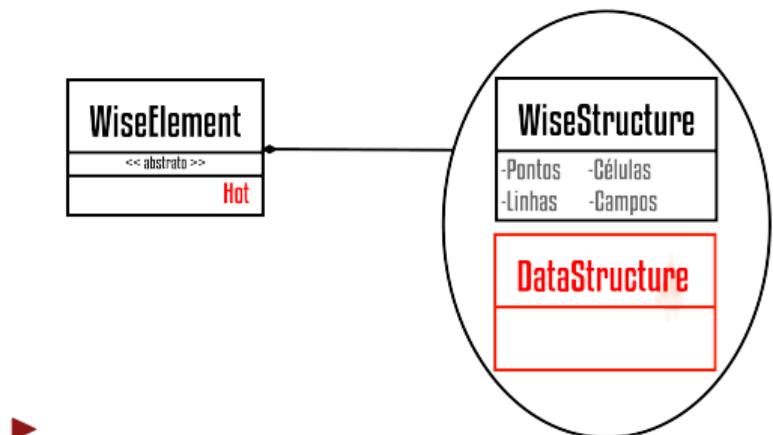


- ▶ Os elementos inteligentes são regidos por uma máquina de estados, aonde os estados representam condições esperadas do elemento inteligente e as transições indicam ações tomadas com o elemento inteligente.

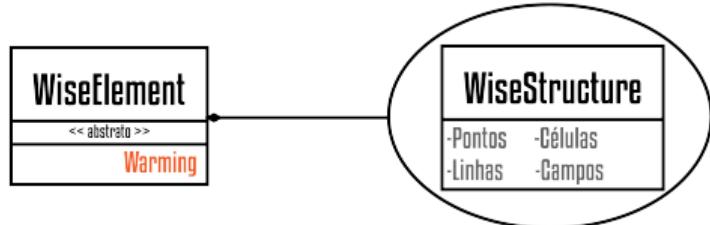
WiseElement Statuses



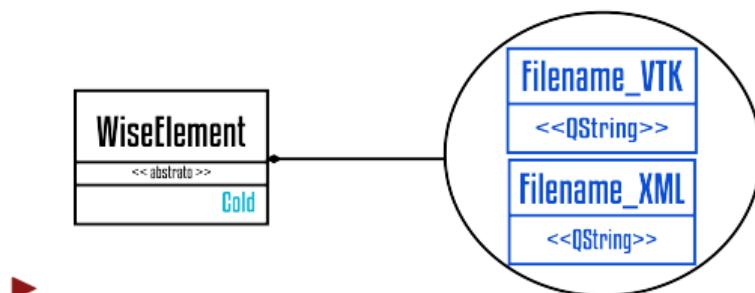
- ▶ Neste estado espera-se que um elemento possua consigo ambas as estruturas presentes no elemento inteligente, seus dados abstratos e a estrutura inteligente *WiseStructure*.



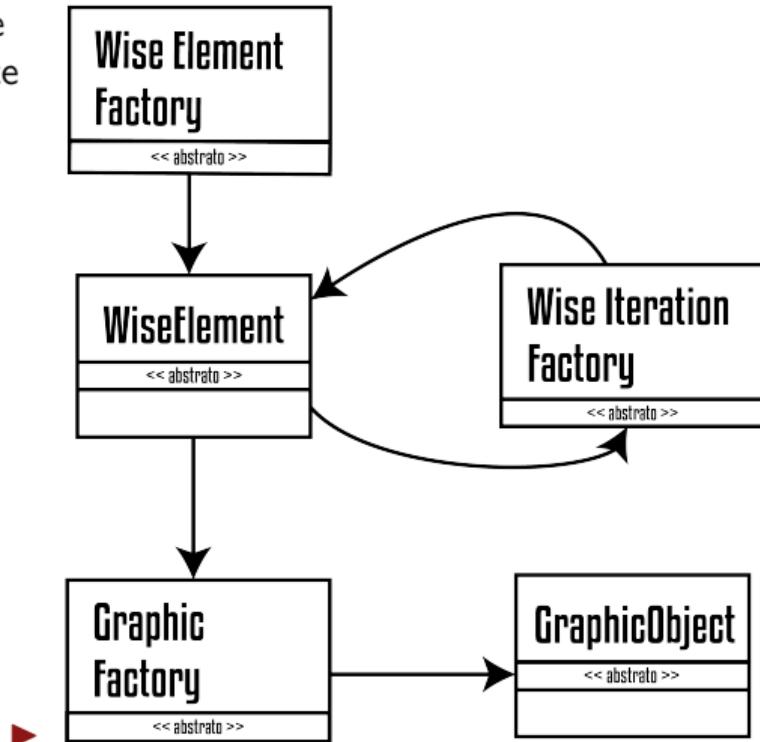
- ▶ O estado *Warming* é equivalente ao estado *Cooling* e *Raw*.
- ▶ Estes estados indicam que somente a estrutura inteligente deste elemento está presente. No caso de um elemento no estado *Raw*, não é esperado que a estrutura completa esteja presente nesta estrutura.
- ▶ Os outros estados indicam que o elemento está completamente carregado na estrutura inteligente e aguarda esfriamento ou aquecimento, processo de armazenar e recuperar arquivos em HD.



- ▶ Espera-se que os elementos neste estado estejam salvos em HD.

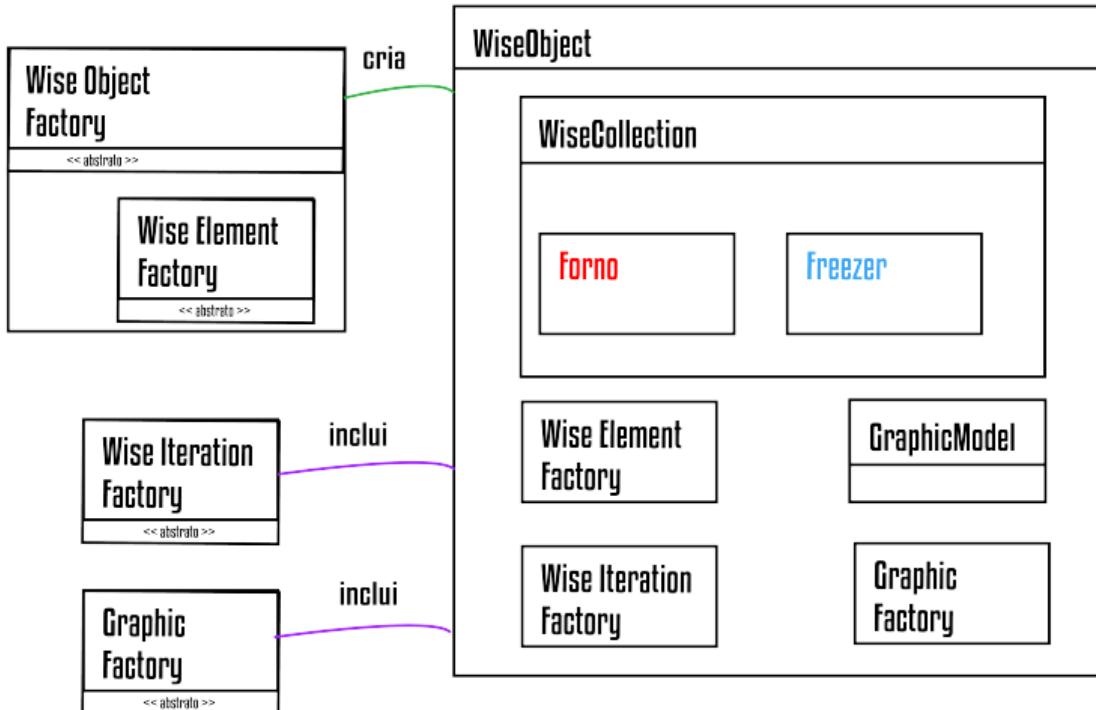


- ▶ Utilizando o padrão de fábricas para criar e manipular elementos inteligentes, o seguinte fluxo de trabalho foi idealizado:

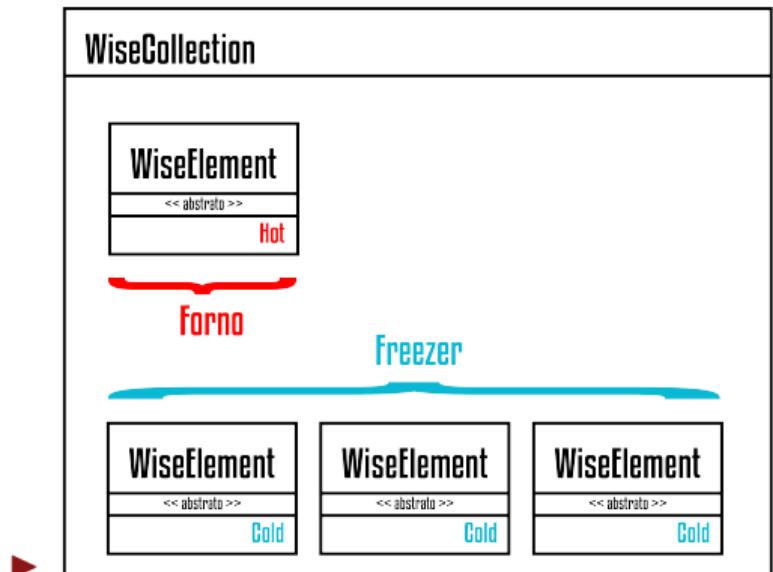


- ▶ A primeira fábrica *WiseElementFactory* é responsável por criar corretamente cada tipo de elemento inteligente.
- ▶ A fábrica *WiseIterationFactory* tem a função de utilizar os dados abstratos de um elemento inteligente com a finalidade de executar algum algoritmo.
- ▶ Finalmente, a fábrica *GraphicFactory* gera os objetos capazes de se desenhar com diretrivas OpenGL *GraphicObject*.

- O objeto inteligente *WiseObject* é o objeto capaz de executar todo esse fluxo de trabalho.

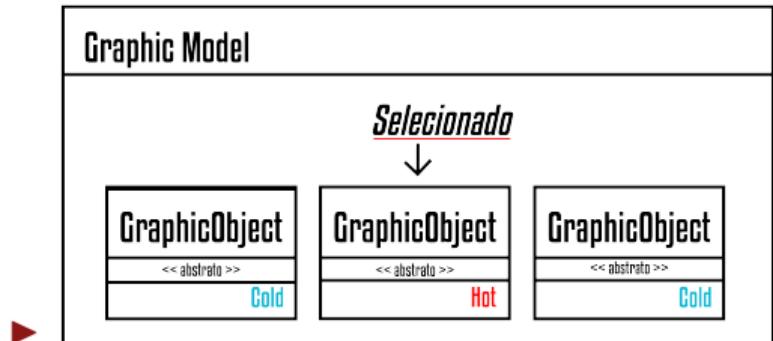


- ▶ Cada objeto inteligente contém uma coleção de elementos inteligentes:

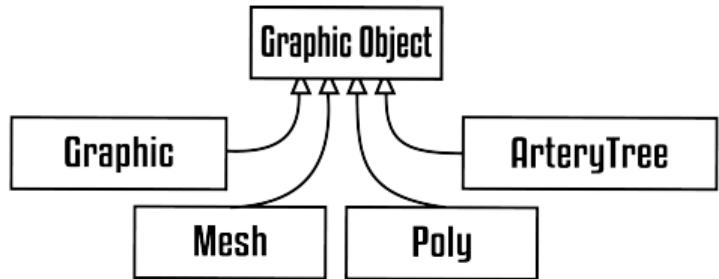


- ▶ Pode-se criar um objeto inteligente com um elemento inteligente, portanto é possível construir um objeto inteligente à partir de qualquer instância de elemento inteligente e sua fábrica.
- ▶ A estrutura *Forno* é um ponteiro para um elemento que estará sempre no estado *Hot*, ele representa o último elemento gerado pela iteração e é utilizado a cada nova iteração.
- ▶ Finalmente, a estrutura *Freezer* é responsável por armazenar os elementos em cache e manter o seu registro.

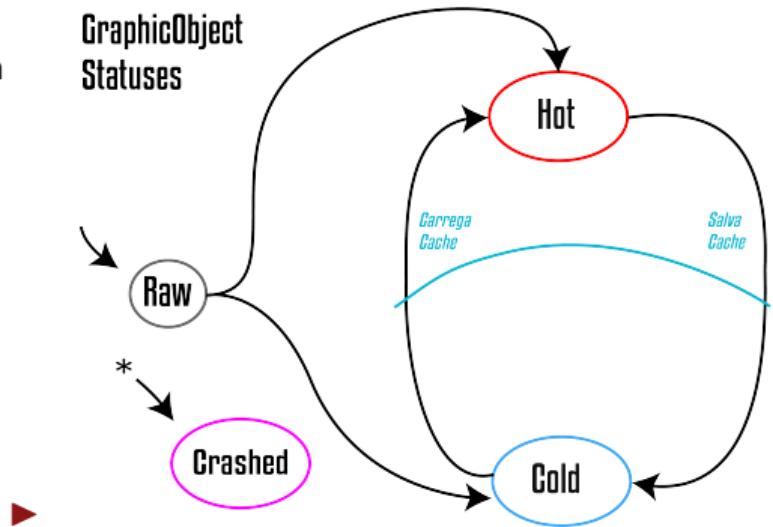
- É possível também que seja incluída a estrutura de visualização do objeto inteligente:



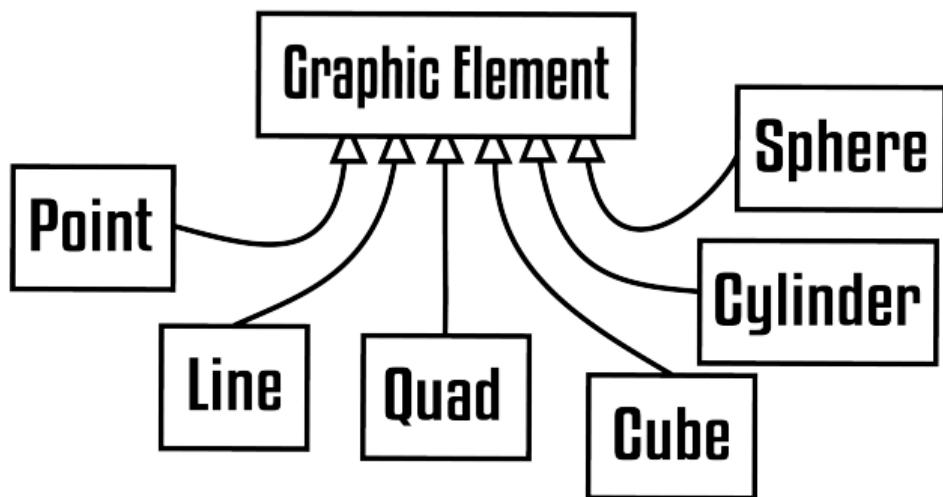
- ▶ Os objetos gráficos são objetos que possuem uma lista de elementos gráficos com valores e é capas de desenhá-los em um quadro OpenGL.
- ▶ O modelo gráfico *GraphicModel* irá manter a coleção de objetos gráficos em memória conforme a necessidade para visualização, mantendo uma quantidade mínima de objetos em memória a todo momento.



- Objetos gráficos também possuem uma máquina de estados que diz se a estrutura está presente em memória ou armazenada em cache:

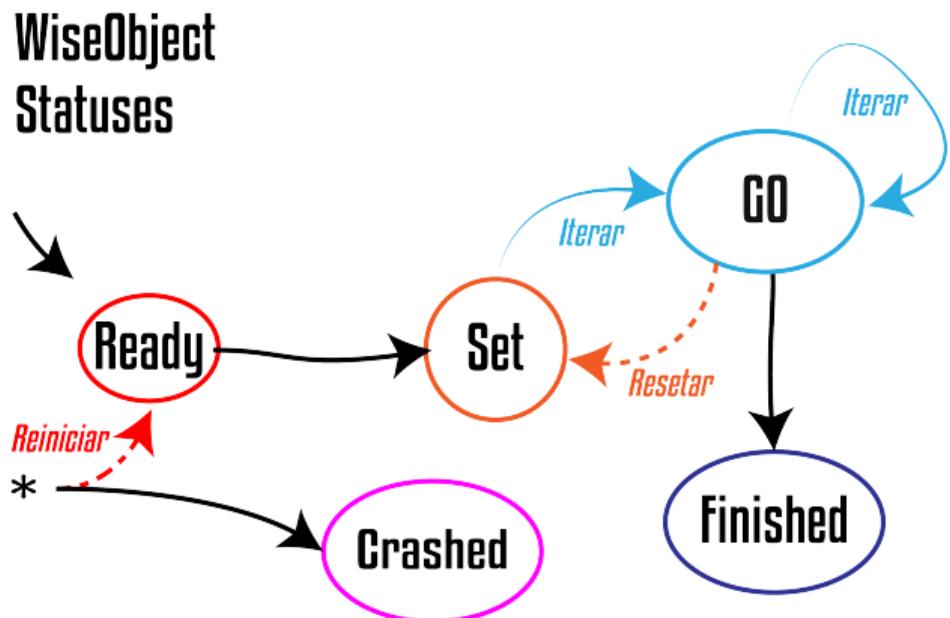


- ▶ Os elementos gráficos são todas as instâncias que implementam a classe abstrata *GraphicElement*:



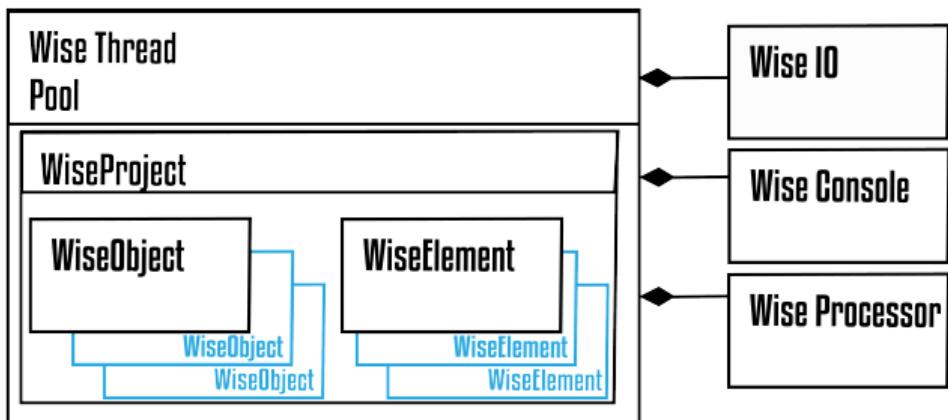
- ▶ Os elementos gráficos armazenam os pontos necessários para desenhá-lo e um valor associado. Este valor associado corresponde à algum valor armazenado em um ponto, uma linha, uma célula ou um campo da *WiseStructure*.

- ▶ Os objetos inteligentes *WiseObject* possuem também uma máquina de estados:



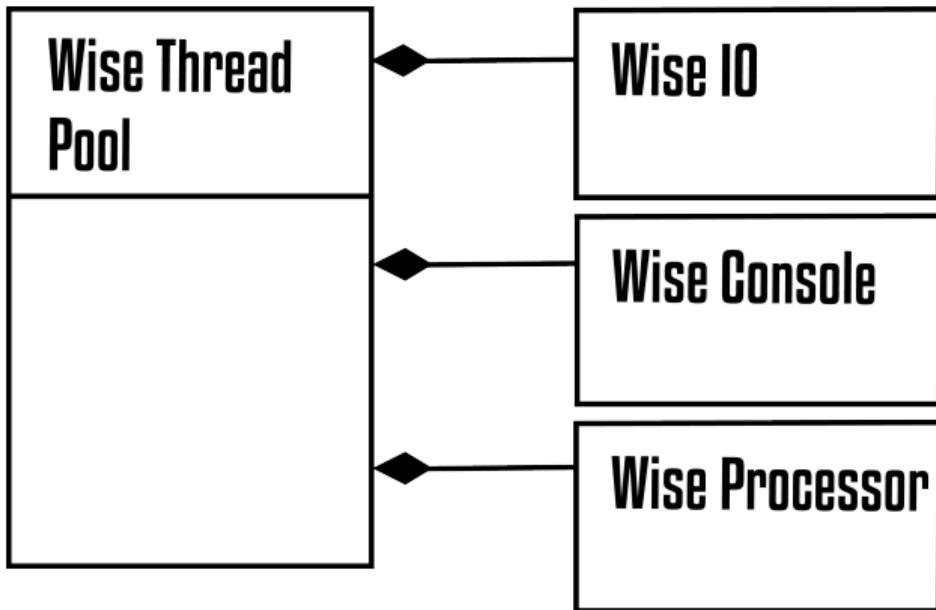
- ▶ Todas as transições de estados e as ações que podem ser executadas sobre um objeto inteligente são interações do usuário. Ao se criar o objeto somente o elemento inicial e sua cópia estarão presentes no *Forno* e no *Freezer*, respectivamente.
- ▶ No estado inicial *Ready* o objeto é capaz de incluir suas fábricas de iteração e gráficas.
- ▶ Após a inclusão de uma fábrica de iteração o objeto pode avançar para o estado *Set*. Neste estado os parâmetros de iteração são adicionados à estrutura *WiseStructure* e podem ser editados pelo usuário, parâmetros como frequência, viscosidade e ângulo de fase.
- ▶ Pode-se arbitrariamente definir o fim das iterações e enviar o objeto para o estado *Finished*, que impossibilitará o objeto de continuar iterando.
- ▶ Finalmente, todos os estados podem levar ao estado *Crashed* que indica o mau funcionamento do objeto.

- Com todas as estruturas básicas definidas, o pacote de classes que organiza os objetos e elementos inteligentes e suas necessidades intitula-se *WiseThreadPool*.

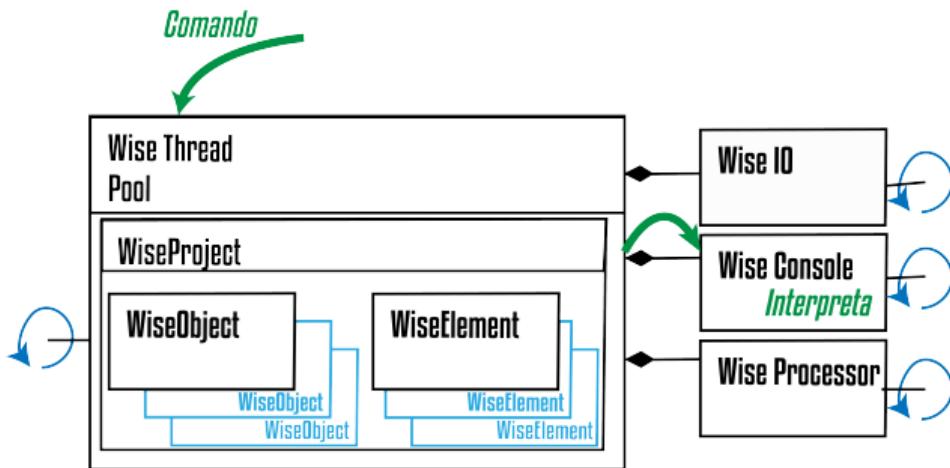


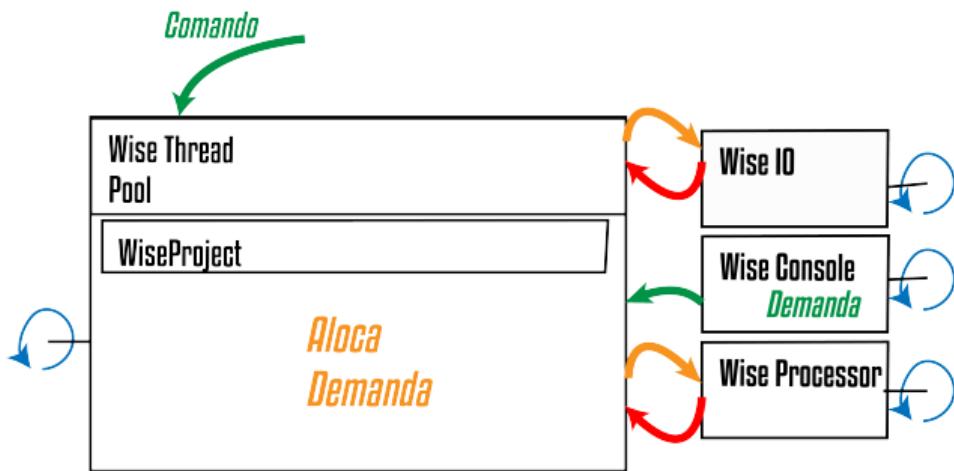
- ▶ Os objetos e elementos inteligentes organizam-se em projetos inteligentes, *WiseProject*.
- ▶ A classe *WiseThreadPool* é responsável por alocar estes projetos e receber suas demandas, resfriar ou aquecer objetos. É responsável também pelas demandas feitas pelo usuário através de linhas de comando.

- ▶ Cada uma das classes à seguir está contida em uma *thread* própria e tem o próprio *loop* de iteração.

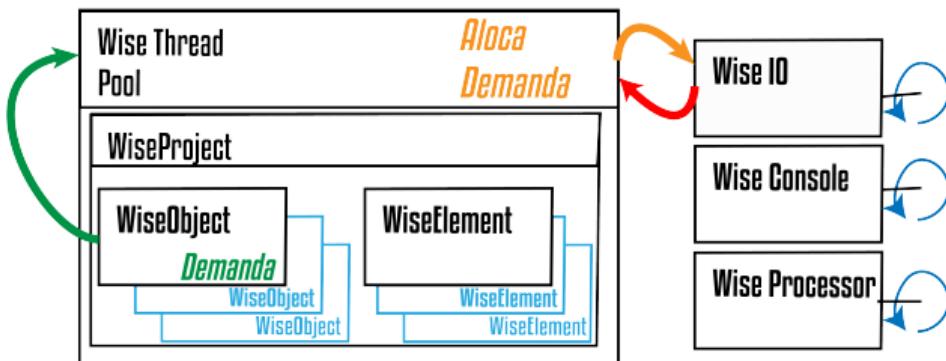


- ▶ Ao receber uma linha de comando o objeto *WiseThreadPool* enviar a mensagem e o projeto atual à uma instância de *WiseConsole*, que é responsável por interpretar o comando.
- ▶ Caso seja um processo que utilize o disco rígido, uma instância *WiseIO* será necessária para executar o comando.
- ▶ Caso seja um processo de iteração, uma instância *WiseProcessor* será utilizada.
- ▶ Para os restantes dos casos a própria classe *WiseConsole* será utilizada para finalizar o comando. A alteração de parâmetros ou sua escala cai neste último caso.





- O resfriamento ou aquecimento de elementos envolve sempre uma instância de *WiseIO*.



- ▶ Um *WiseThreadPool* irá conter todas as instâncias de *WiseIO*, *WiseConsole* e *WiseProcessor* e está preparado para utilizar mais de uma instância destes objetos por vez.
- ▶ Essencialmente isto significa que os objetos e elementos armazenados pelo projeto inteligente, *WiseProject*, podem ser acessados pelos diferentes processadores independentemente.
- ▶ Isto dá ao usuário uma imensa liberdade na hora de decidir executar algum algoritmo concorrentemente, entretanto requer cuidados especiais na hora de utilizar os objetos.
- ▶ Como os objetos podem ser acessados por mais de um processador, eles são bloqueados por uso.

- ▶ Com a estrutura de dados definida a ferramenta foi dividida em dois ambientes:
- ▶ A interface gráfica ***IGU*** (Interface Gráfica Universal), que representa o ambiente gráfico da ferramente.
- ▶ E, o console ***InGU***(Interface não-Gráfica Universal), que é um ambiente executado sem interface gráfica e recebe comandos via texto.

- O ambiente **IngU** ao ser executado imprime no console o cabeçalho do programa e carrega os elementos previamente carregados na ferramenta.

```
=====
/=====/
. /====/      IGU (Iterador Gráfico Universal)      /=====/
. /====/      (or, Universal Graphic Iterator)      /=====/
. /=====/
. /=1.1=====
. /=====/
. segunda-feira 09/08/2021 19:19:08 313
. /=====
. [WISE CONSOLE] LOAD invoked
. <p1:WISE_PROJECT> 'WISE_ID [0] (NAME: p1) WISE PROJECT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [0] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_OBJECT> 'WISE_ID [0] (NAME: obj1) WISE OBJECT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [1] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [2] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [3] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [4] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [5] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [6] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [7] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [8] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [9] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [10] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [11] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [12] (NAME: el1) WISE ELEMENT CREATED'
. <el1:WISE_ELEMENT> 'WISE_ID [13] (NAME: el1) WISE ELEMENT CREATED'
```

- ▶ O comando de ajuda é o primeiro comando da interface e foi feito para listar todas as entradas possíveis do programa.
- ▶ Ao receber este comando a thread *WiseConsole* envia o texto pré-definido com todos os comandos.

Linha de Comando	help
Escopo	nenhum
Thread Responsável	WiseConsole
Entrada	Nenhuma

Tabela 1: Descrição do comando ajuda.

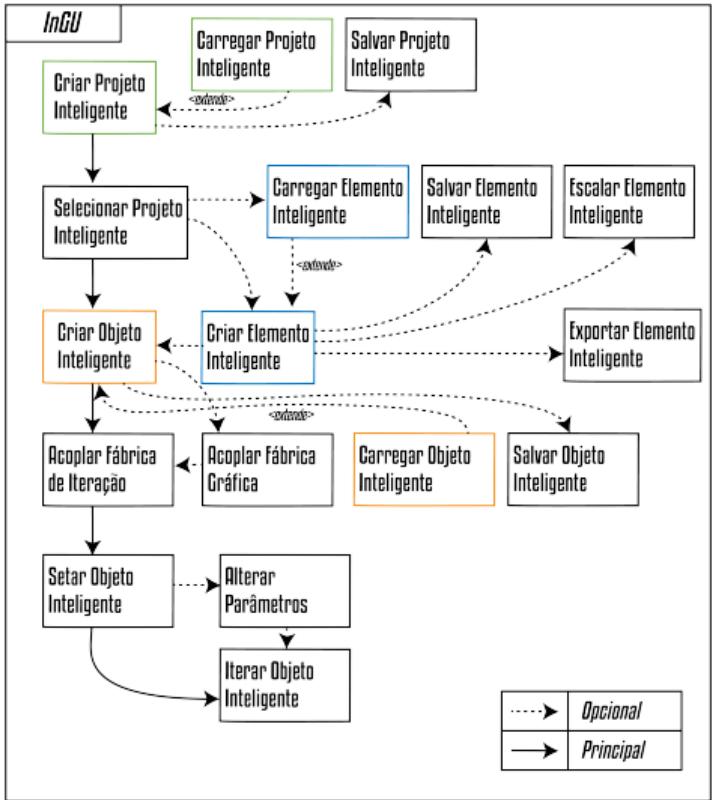
Assim como o comando de criação de elementos inteligentes, o comando de criação de objetos irá acessar a fábrica de elementos inteligentes *WiseElementFactory*. É possível criar objetos inteligentes de duas formas: A primeira, utilizando um elemento inteligente; A segunda utilizando os exemplos disponibilizados pela fábrica de elementos inteligentes.

Assim como descrito anteriormente, ao criar um objeto inteligente, um elemento inteligente é adicionado à estrutura do *Forno*, enquanto um *Clone* é acoplado ao *Freezer*;

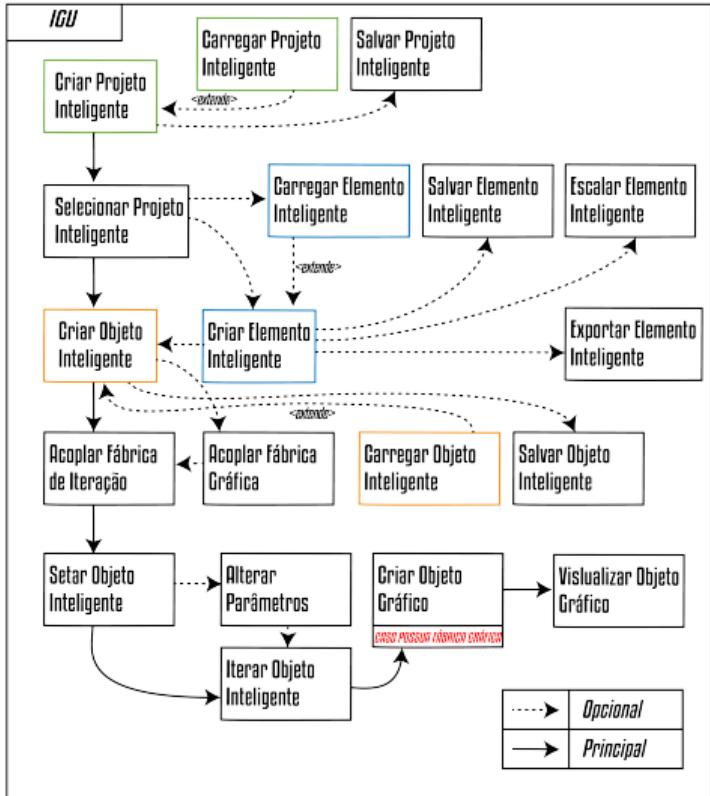
Linha de Comando	object create <object_name> <element_name> object create <type> <example> <name> <element_name> [ARGS]	
Escopo	OBJECT	
Thread Responsável	WiseConsole	
Entrada	<object_name> <element_name> <type> <name> [ARGS]	Nome do objeto à ser criado. Nome do elemento à ser utilizado na criação ou do elemento à ser criado a partir do exemplo. Tipo de elemento inteligente à ser criado. Nome do exemplo de elemento inteligente à ser criado. Individualmente, as fábricas podem receber pa- râmetros para a criação de elementos.

Tabela 2: Descrição do comando para criar objetos inteligentes.

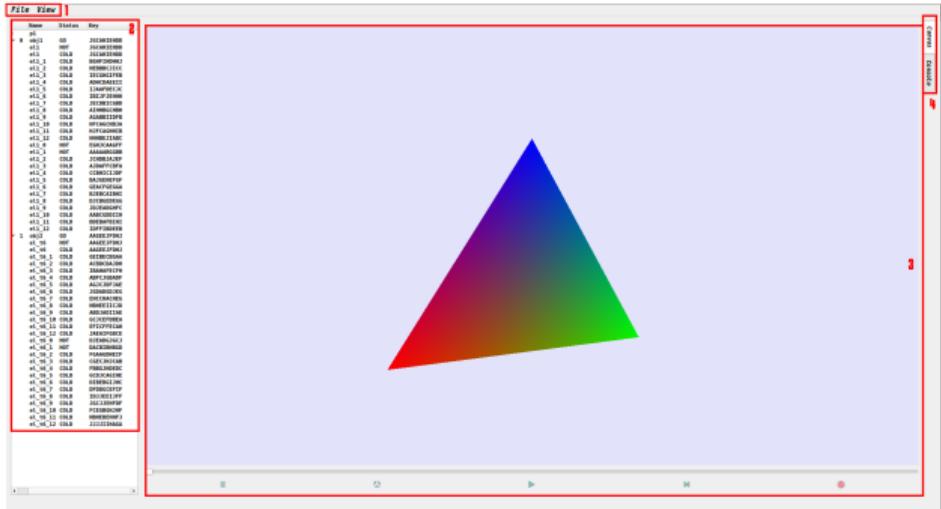
- O ambiente **InGU** tem como principal objetivo a iteração modelos utilizando alguma lógica pré-definida por algum algoritmo disponível. Através dos comandos disponibilizados é possível que estes modelos sejam criados, alterados, iterados e, opcionalmente, visualizados. Com isto o principal fluxo de uso foi disponibilizado.



- ▶ Já o ambiente **IGU** tem como principais objetivos possibilitar a realização do fluxo disponibilizado no ambiente **InGU** e incorporar um interface de usuário gráfica. Através da interface de usuário é possível que estes modelos sejam criados, alterados, iterados e, opcionalmente, visualizados.

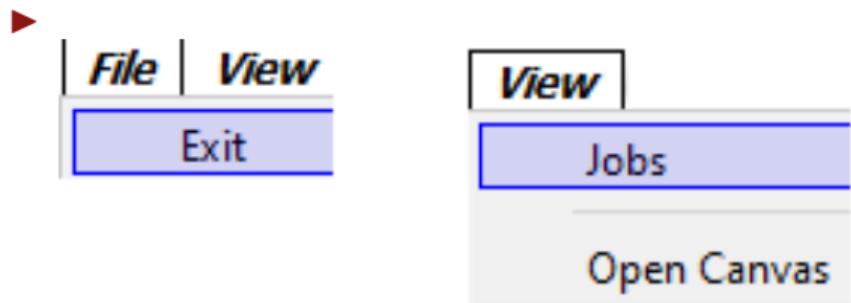


- A janela principal do ambiente computacional **IGU** é composta por: 1. Menu principal do programa; 2. Árvore de projetos e seus elementos; 3. Área de trabalho, no caso mostrando OpenGL *Canvas*; 4. Seleção de abas.



IGU (cont.)

- ▶ O primeiro grupo são os elementos que compõe o menu principal da aplicação. Cada opção do menu é representada por uma linha de texto, ao selecionar a linha uma ação é executada:
- ▶ **Exit:** Fecha o ambiente computacional.
- ▶ **Jobs:** Abre a Janela que exibe os trabalhos inteligentes criados.
- ▶ **Open Canvas:** Abre uma nova janela contendo um novo elemento gráfico OpenGL Canvas.

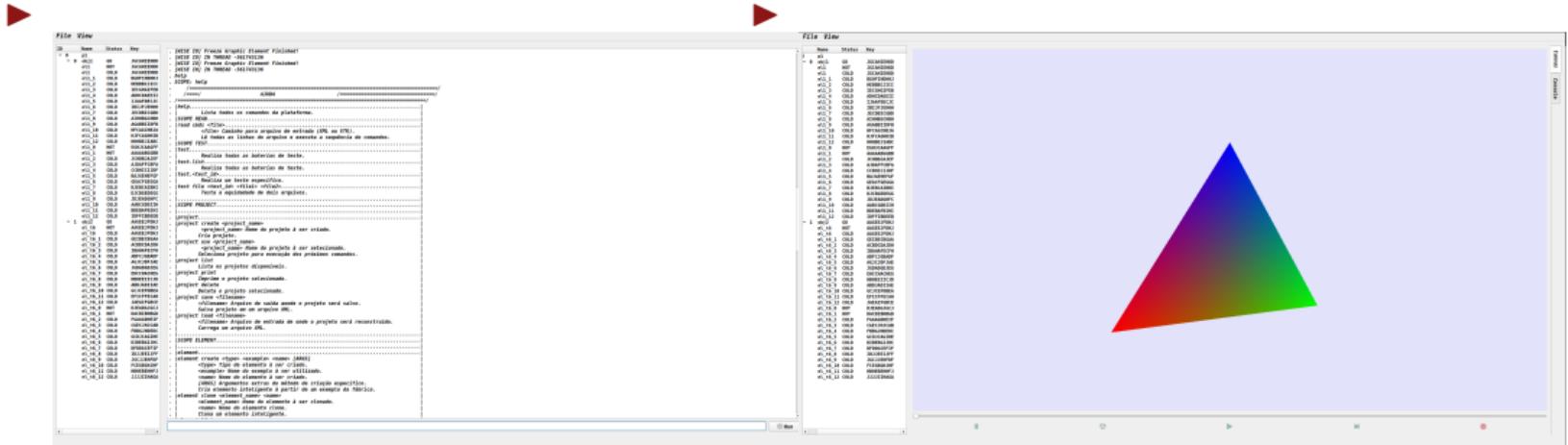


- ▶ Ao selecionar uma das abas, os elementos de interface de usuário da área de trabalho são trocados. Estes elementos são objetos do tipo *QWidget*, estes objetos que são divididos em dois ambientes, o *Canvas* e o *Console*.

- ▶ Ao selecionar a opção do *Console* um ambiente similar ao disponibilizado pelo ambiente *InGU* é disponibilizado.

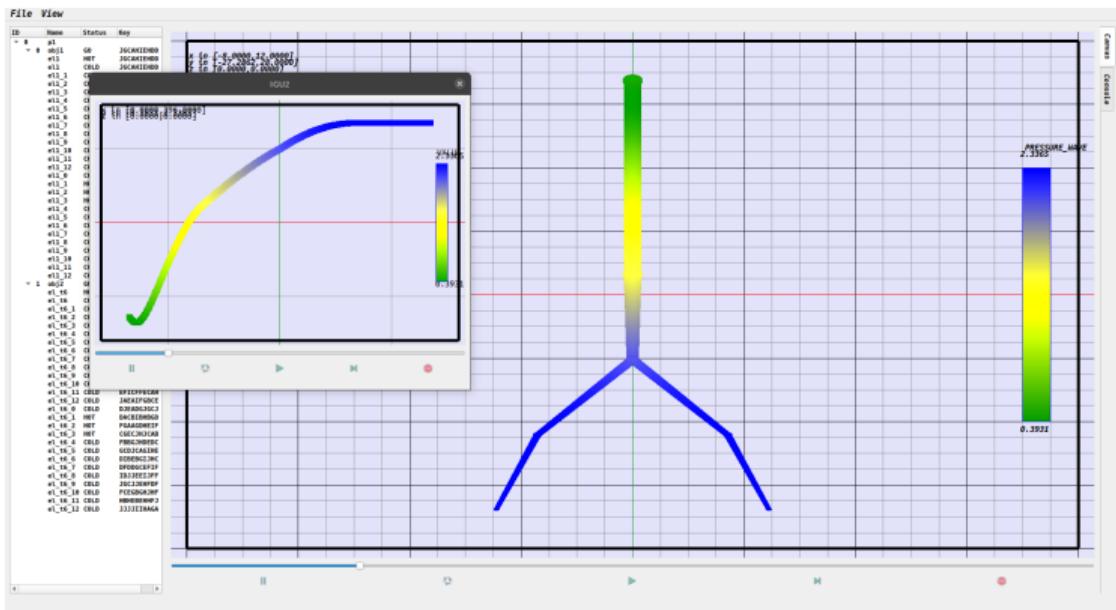


IGU (cont.)



- ▶ A janela e área de trabalho *Canvas* possibilita que o usuário selecione o quadro à ser exibido. Cada quadro representa um objeto do tipo *GraphicObject* que é um componente do modelo gráfico *GraphicModel*. O quadro é selecionado através de uma barra de progresso, além de botões que alteram a animação feita com o objeto gráfico.

Janelas (cont.)

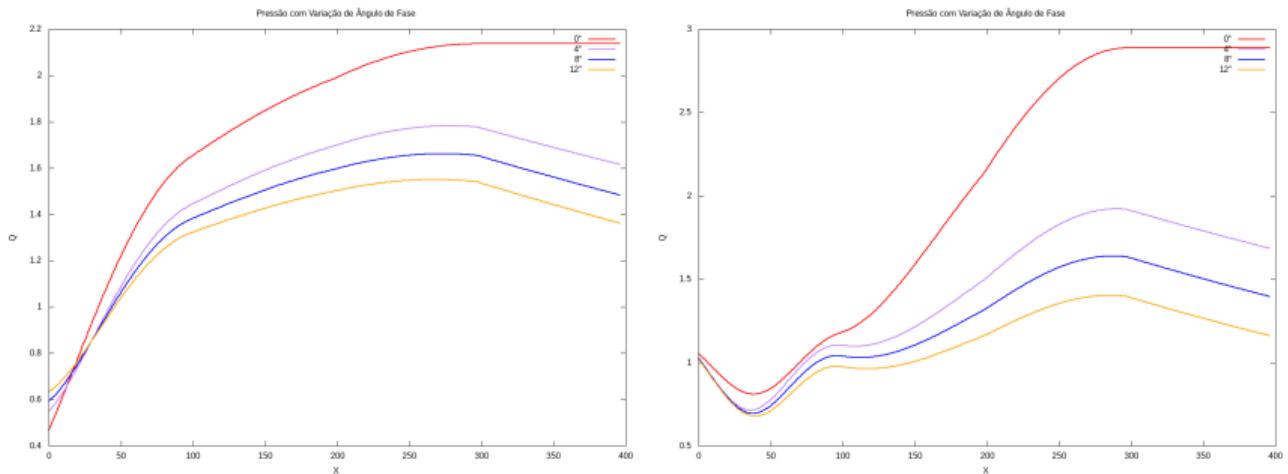


- A janela *Jobs* exibe uma listagem comprehensiva dos processos iniciados e enviados à thread inteligente *WiseThreadPool*, cada comando executado via *Console* ou elemento de interface gráfica irá gerar processos listados nesta janela.

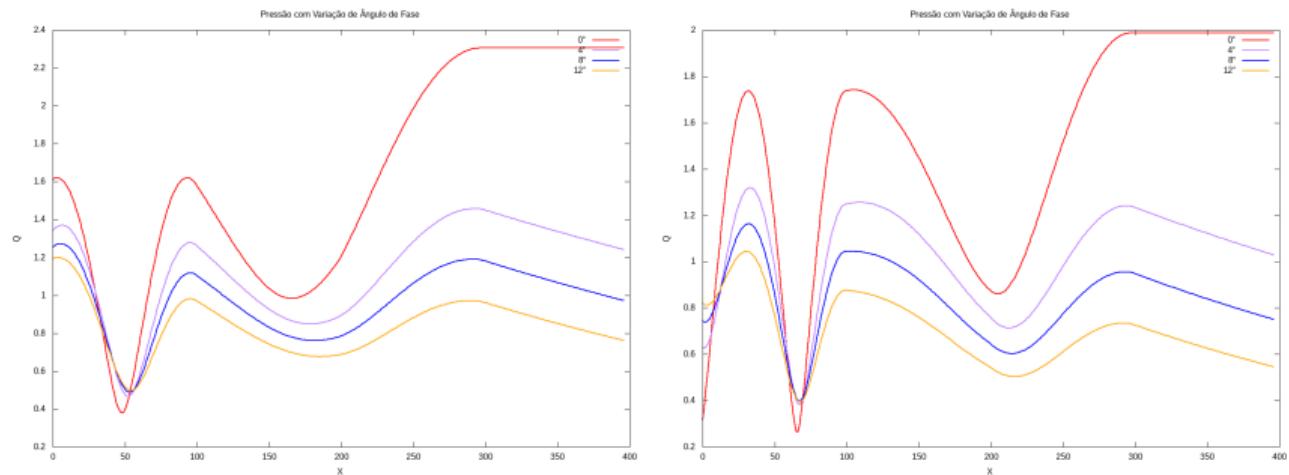
ID	Workload	Type	Status	Created at	Msec to Q	Msec to Run	Msec to Finish	Msec Running	Finished at
52	51	0	RUN_CMD	FINISHED	15/08 23:23:09.564	0	0	0	15/08 23:23:09.565
53	52	0	CANVAS_LINE	FINISHED	15/08 23:23:09.565	0	0	0	15/08 23:23:09.565
54	53	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:29.695	23	23	38	15/08 23:23:29.735
55	54	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:26.361	16	16	33	15/08 23:23:26.395
56	55	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:26.361	16	16	34	15/08 23:23:26.396
57	56	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:31.146	17	17	32	15/08 23:23:31.179
58	57	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:31.146	17	17	33	15/08 23:23:31.180
59	58	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:41.461	16	16	34	15/08 23:23:41.495
60	59	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:41.461	16	16	34	15/08 23:23:41.496
61	60	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:41.461	16	16	35	15/08 23:23:41.497
62	61	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:41.461	16	16	36	15/08 23:23:41.498
63	62	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:45.511	16	16	34	15/08 23:23:45.545
64	63	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:45.511	16	16	34	15/08 23:23:45.546
65	64	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:49.145	16	16	34	15/08 23:23:49.179
66	65	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:49.145	16	16	34	15/08 23:23:49.179
67	66	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:52.345	16	16	33	15/08 23:23:52.378
68	67	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:52.345	16	16	34	15/08 23:23:52.379
69	68	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:56.230	16	16	34	15/08 23:23:56.265
70	69	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:23:56.230	16	16	35	15/08 23:23:56.265
71	70	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:04.244	16	16	34	15/08 23:24:04.279
72	71	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:04.244	16	16	34	15/08 23:24:04.279
73	72	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:04.244	16	16	35	15/08 23:24:04.280
74	73	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:04.244	16	16	36	15/08 23:24:04.281
75	74	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:08.178	16	16	34	15/08 23:24:08.212
76	75	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:08.178	16	16	34	15/08 23:24:08.213
77	76	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:15.495	16	16	34	15/08 23:24:15.529
78	77	6	FREEZE_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:15.495	16	16	34	15/08 23:24:15.529
79	78	6	HEAT_GRAPHIC_ELEMENT	FINISHED	15/08 23:24:15.778	16	16	33	15/08 23:24:15.812

Janelas (cont.)

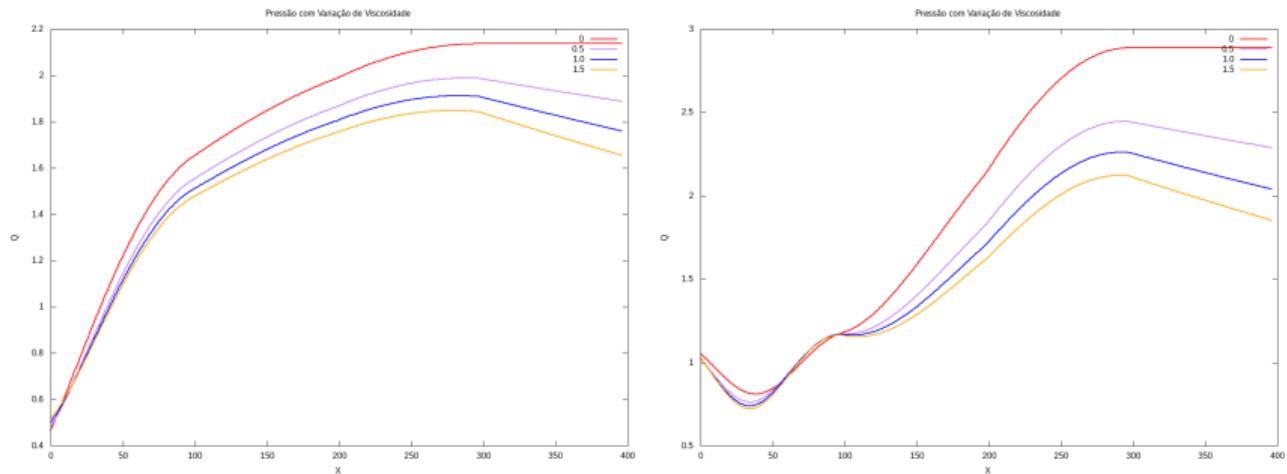
Resultados



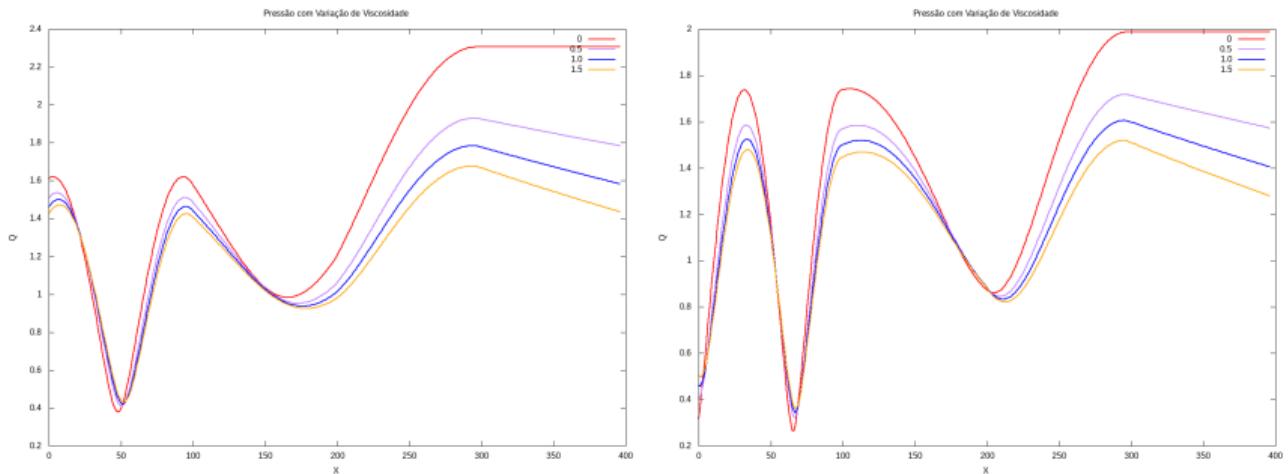
Resultados (cont.)



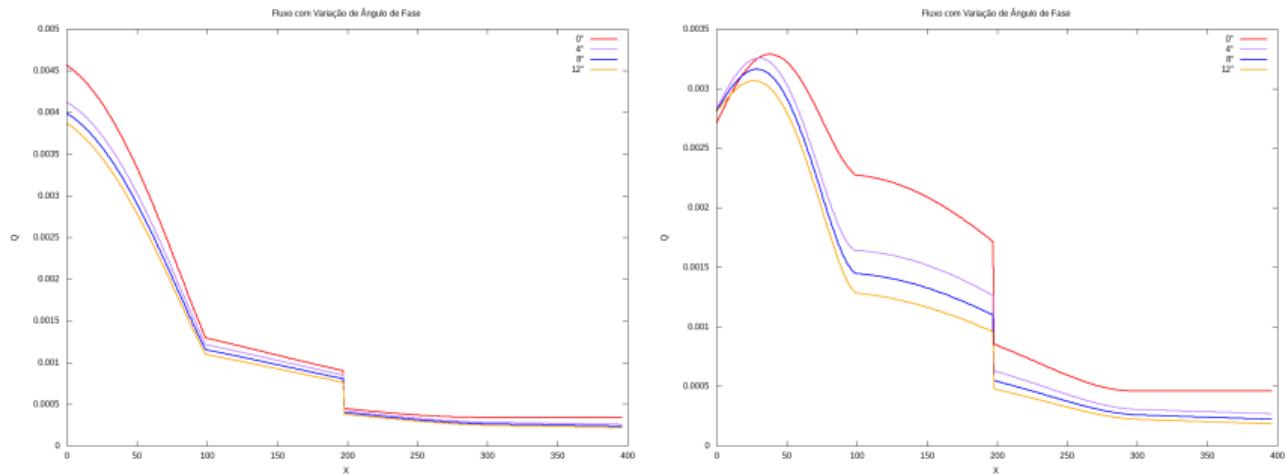
Resultados (cont.)



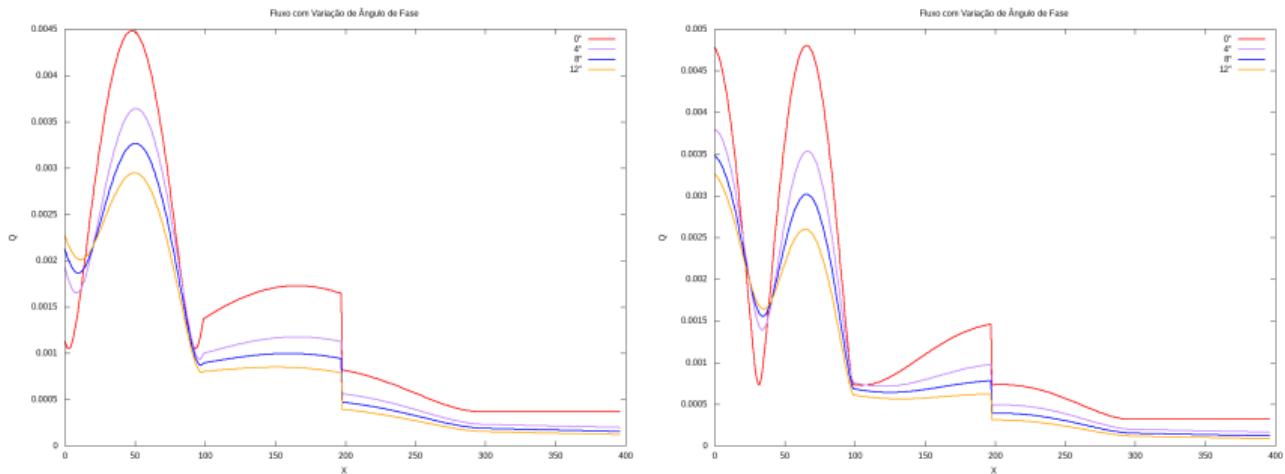
Resultados (cont.)



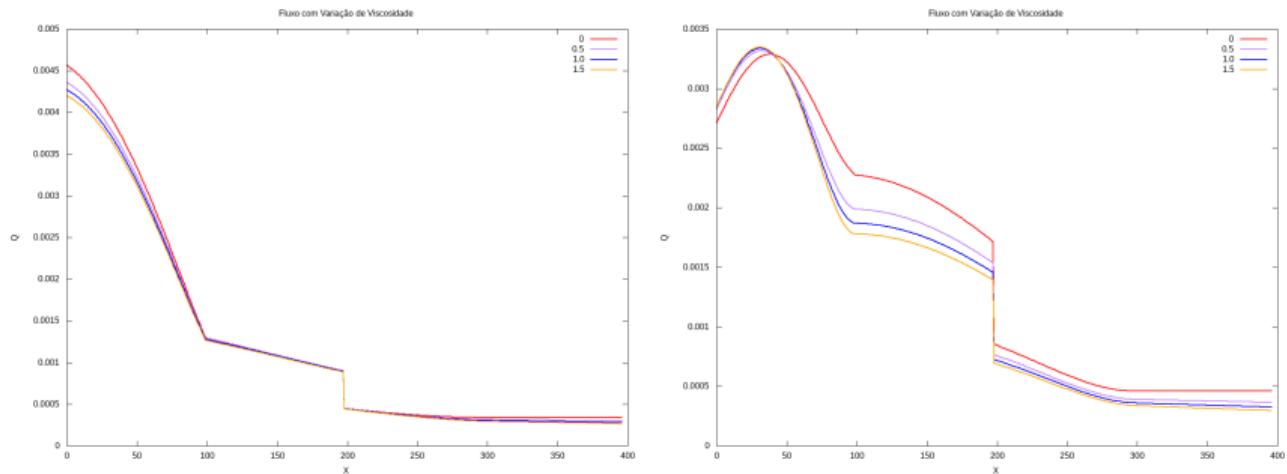
Resultados (cont.)



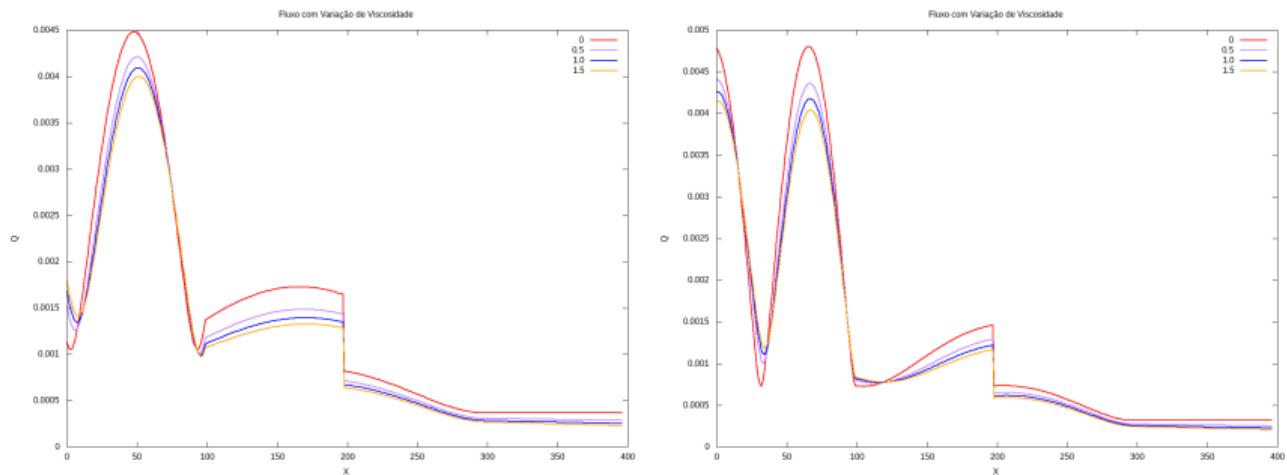
Resultados (cont.)



Resultados (cont.)



Resultados (cont.)



- ▶ Ferramenta Computacional
- ▶ Estrutura de dados
- ▶ **dividir em elemento inteligente, objeto inteligente e objeto gráfico?**
- ▶ (Sinais e slots, paralelização)
- ▶ Lista de comandos
- ▶ Interface gráfica
- ▶ Resultados
- ▶ Fluxo
- ▶ Pressão
- ▶ Conclusão

- ▶ Mar: Ajustes dissertação + Fim da dissertação
- ▶ Abr: Ajustes finais
- ▶ Mai: Ajustes finais