

# Resumo Executivo: Código FEniCSx Genérico

---

## Principais Características

### ✓ Descoberta Automática de Physical Groups

```
# 🛠 FEniCSx: Extração automática de tags
self.discovered_cell_tags = np.unique(self.cell_tags.values)
self.discovered_facet_tags = np.unique(self.facet_tags.values)
```

### ✓ Mapeamento Dinâmico YAML ↔ Malha

```
# Em vez de hardcoded
if domain_id == 5: # ✗
    material = "concreto_face"

# Mapeamento dinâmico
if domain_id in self.material_mapping: # ✓
    material = self.material_mapping[domain_id]
```

### ✓ Etapas Construtivas com Birth/Death

```
# Verificação dinâmica de ativação
if domain_id not in self.active_layers or not
self.active_layers[domain_id]:
    continue # Pula domínios inativos
```

### ✓ Formulação Genérica Adaptativa

```
# 🛠 FEniCSx: Formulação que adapta aos domínios encontrados
for domain_id in self.discovered_cell_tags:
    if domain_id in self.active_layers and self.active_layers[domain_id]:
        F += termo_variacional * dx_tags(domain_id)
```

---

## Principais Comandos FEniCSx

### 1. Carregamento de Malha

```
# 🛠️ FEniCSx: I/O de malha
with io.XDMFFile(self.comm, self.xdmf_file, "r") as xdmf:
    self.mesh = xdmf.read_mesh(name="malha")
    self.cell_tags = xdmf.read_meshtags(self.mesh, name="malha_cells")
    self.facet_tags = xdmf.read_meshtags(self.mesh, name="malha_facets")
```

## 2. Espaços de Função

```
# 🛠️ FEniCSx: Definição de espaços
self.V = fem.functionspace(self.mesh, ("Lagrange", 1))
self.T = Function(self.V)
self.v = TestFunction(self.V)
```

## 3. Formulação Variacional

```
# 🛠️ FEniCSx: Formulação UFL
F += rho * cp * (self.T - self.Tn) / dt * self.v * dx_tags(domain_id)
F += k * dot(grad(T_theta), grad(self.v)) * dx_tags(domain_id)
```

## 4. Resolução do Sistema

```
# 🛠️ FEniCSx: Solver não-linear
problem = NonlinearProblem(F, self.T, bcs)
solver = NewtonSolver(self.comm, problem)
n_iterations, converged = solver.solve(self.T)
```

## 5. Salvamento de Resultados

```
# 🛠️ FEniCSx: Exportação para Paraview
with io.XDMFFile(self.comm, output_file, "w") as xdmf:
    xdmf.write_mesh(self.mesh)
    xdmf.write_function(self.T, current_time)
```



## Vantagens da Abordagem

### 📈 Reutilização

- Funciona com **qualquer malha Gmsh**
- Aceita **qualquer configuração YAML**
- Suporta **qualquer sequência construtiva**

## Manutenibilidade

- **Zero hardcoding** - valores extraídos dinamicamente
- **Configuração externa** - parâmetros no YAML
- **Modular** - funções independentes e testáveis

## Performance

- **Otimização automática** - só processa domínios ativos
- **Paralelização MPI** - suporte nativo
- **Memória eficiente** - estruturas otimizadas

---

## Comandos FEniCSx por Categoria

### Malha e I/O

- `io.XDMFFile()` - Leitura/escrita XDMF
- `xdmf.read_mesh()` - Carregamento de malha
- `xdmf.read_meshtags()` - Leitura de Physical Groups
- `mesh.topology.create_connectivity()` - Conectividade

### Espaços de Função

- `fem.functionspace()` - Criação de espaço FEM
- `Function()` - Funções no espaço
- `TestFunction()` - Funções de teste
- `Constant()` - Constantes

### Formulação UFL

- `dx(subdomain_data=...)` - Integração por subdomínio
- `ds(subdomain_data=...)` - Integração de superfície
- `grad()`, `dot()`, `inner()` - Operadores diferenciais

### Resolução

- `NonlinearProblem()` - Problema não-linear
- `NewtonSolver()` - Solver Newton
- `solver.solve()` - Resolução do sistema





### Paralelização

- `MPI.COMM_WORLD` - Comunicador MPI
- `self.comm.Get_rank()` - Rank do processo
- `PETSc.ScalarType` - Tipos de dados PETSc

---

## Conclusão

Este código representa um **framework avançado** que demonstra como o **FEniCSx** pode ser usado para criar simuladores:

-  **Genéricos** - não particularizados
-  **Robustos** - com descoberta automática
-  **Eficientes** - com otimizações automáticas
-  **Reutilizáveis** - para problemas similares

A arquitetura desenvolvida serve como **base sólida** para simulações de elementos finitos complexas, combinando a flexibilidade do **FEniCSx** com práticas avançadas de engenharia de software.