**Especificação dos Requisitos**

**da**

**OPLA-Tool**

**Versão 4.0**

|  |  |  |
| --- | --- | --- |
| **Diego Fernandes da Silva** | **R.A.: PG 400800** |  |
| **Fernando S. Godói** | **R.A.: PG 400590** |  |
| **Mamoru Massago** | **R.A.: PG 400802** |  |
| **Tiago Madrigar** | **R.A.: PG 400594** |  |

|  |  |
| --- | --- |
| **Professor(a):** | **Edson Alves Oliveira Junior** |
| **Disciplina:** | **Engenharia de Software** |
|  |  |
|  |  |

**Sri**

Sumário

[1. Introdução 7](#_Toc517027856)

[1.1 Objetivos do Documento 7](#_Toc517027857)

[1.2 Escopo do Produto 7](#_Toc517027858)

[1.3 Público-Alvo 7](#_Toc517027859)

[1.4 Definições, Acrônimos e Abreviações 7](#_Toc517027860)

[1.5 Convenções 7](#_Toc517027861)

[1.6 Glossário 8](#_Toc517027862)

[1.7 Referências 8](#_Toc517027863)

[2. Visão Geral 10](#_Toc517027864)

[2.1 Perspectiva do Produto 10](#_Toc517027865)

[2.2 Funcionalidade do Produto 10](#_Toc517027866)

[2.3 Usuários 10](#_Toc517027867)

[2.4 Ambiente Operacional 10](#_Toc517027868)

[2.5 Restrições de Projeto e Implementação 10](#_Toc517027869)

[2.6 Documentação do Usuário 11](#_Toc517027870)

[2.7 Suposições e Dependências 11](#_Toc517027871)

[3. Especificação das Interfaces Externas 12](#_Toc517027872)

[3.1 Requisitos de Interface Externa 12](#_Toc517027873)

[3.1.1 Interfaces do Usuário 12](#_Toc517027874)

[3.1.2 Interfaces de Hardware 17](#_Toc517027875)

[3.1.3 Interfaces de Software 17](#_Toc517027876)

[3.1.4 Interfaces de Comunicação 18](#_Toc517027877)

[4. Requisitos Funcionais 19](#_Toc517027878)

[4.1 FUNC001 Representação da PLA 19](#_Toc517027879)

[4.1.1 RF001 Formato do arquivo de entrada 19](#_Toc517027880)

[4.1.2 RF002 Codificação da Entrada 20](#_Toc517027881)

[4.2 FUNC002 Otimização Multiobjetivo 21](#_Toc517027882)

[4.2.1 RF003 – Configurações Gerais (interface do item 3.1.1.1) 22](#_Toc517027883)

[4.2.2 RF004 – Configuração da Execução (interface do item 3.1.1.2) 22](#_Toc517027884)

[4.2.3 RF005 – Otimização utilizando Padrões de Projeto (interface do item 3.1.1.3) 23](#_Toc517027885)

[4.2.4 RF006 - Resultados (interface do item 3.1.1.4) 23](#_Toc517027886)

[4.2.5 RF007 – Comparação de Execuções (interface do item 3.1.1.5) 24](#_Toc517027887)

[4.2.6 RF008 - Logs (interface do item 3.1.1.6) 24](#_Toc517027888)

[4.3 FUNC003 Decodificação das alternativas de PLA 24](#_Toc517027889)

[4.3.1 RF009 Decodificação da Saída 24](#_Toc517027890)

[5. Requisitos Não-Funcionais 26](#_Toc517027891)

[5.1 Requisitos de Desempenho 26](#_Toc517027892)

[5.1.1 RNF001 Tempo para Feedback da Execução 26](#_Toc517027893)

[5.2 Requisitos de Segurança 26](#_Toc517027894)

[5.2.1 RNF002 Consistência de Dados 26](#_Toc517027895)

[5.3 Atributos de Qualidade do Software 26](#_Toc517027896)

[5.3.1 RNF003 Qualidade de Software 26](#_Toc517027897)

[5.4 Usabilidade 26](#_Toc517027898)

[5.4.1 RNF004 Usabilidade 26](#_Toc517027899)

[6. Outros Requisitos 27](#_Toc517027900)

[7. Workflow de Requisitos 28](#_Toc517027901)

[7.1 Visão de Negócio 28](#_Toc517027902)

[7.2 Modelo de Objetos de Negócio 29](#_Toc517027903)

[7.3 Modelo de Caso de Uso 30](#_Toc517027904)

[7.4 Tabela de Conceitos 36](#_Toc517027905)

[7.5 Arquitetura Inicial do Sistema 36](#_Toc517027906)

[8. Workflow de Análise 37](#_Toc517027907)

[8.1 Realização e Comunicação dos casos de uso 37](#_Toc517027908)

[8.1.1 UC001 – Gerir Configuração 37](#_Toc517027909)

[8.1.2 UC002 – Iniciar Otimização 39](#_Toc517027910)

[8.1.3 UC003 – Acompanhar Execução 40](#_Toc517027911)

[8.1.4 UC004 – Gerir Experimentos 41](#_Toc517027912)

[8.2 Análise Arquitetural do sistema 43](#_Toc517027913)

[8.3 Visão Global 44](#_Toc517027914)

[9. Workflow de Projeto 45](#_Toc517027915)

[9.1 Diagrama de Classes 45](#_Toc517027916)

[9.1.1 UC001 – Gerir Configuração 45](#_Toc517027917)

[9.1.2 UC002 – Iniciar Otimização 46](#_Toc517027918)

[9.1.3 UC003 – Acompanhar Execução 46](#_Toc517027919)

[9.1.4 UC004 – Gerir Experimentos 47](#_Toc517027920)

[9.2 Diagrama de Sequência 47](#_Toc517027921)

[9.2.1 UC001 – Gerir Configuração 47](#_Toc517027922)

[9.2.2 UC002 – Iniciar Otimização 49](#_Toc517027923)

[9.2.3 UC003 – Acompanhar Execução 49](#_Toc517027924)

[9.2.4 UC004 – Gerir Experimentos 50](#_Toc517027925)

[9.3 Diagrama de Estado 51](#_Toc517027926)

[9.4 Projeto Arquitetural 52](#_Toc517027927)

[9.5 Modelo de Instalação 53](#_Toc517027928)

[9.6 Padrões e Frameworks 54](#_Toc517027929)

[9.7 Diagrama de Classe – Visão Global 55](#_Toc517027930)

[10. Protótipo 56](#_Toc517027931)

[10.1 Configurações Gerais 56](#_Toc517027932)

[10.2 Configurações de Execução 57](#_Toc517027933)

[10.3 Padrão de Projeto (*Design Pattern*) 58](#_Toc517027934)

[10.4 Resultados 59](#_Toc517027935)

[10.5 Experimentos 60](#_Toc517027936)

[10.6 Logs 61](#_Toc517027937)

**Revisões**

|  |  |  |  |
| --- | --- | --- | --- |
| **Versão** | **Autores** | **Descrição da Versão** | **Data** |
| 0.1 | Diego  Mamoru  Tiago | Início do preenchimento da introdução e visão geral | 02/04/2018 |
| 0.2 | Diego  Mamoru  Tiago | Início especificação dos requisitos funcionais e não-funcionais | 03/04/2018 |
| 0.3 | Diego  Mamoru  Tiago | Correções de alguns requisitos | 05/04/2018 |
| 0.4 | Diego  Mamoru  Tiago | Segunda correção dos requisitos | 11/04/2018 |
| 0.5 | Tiago | Especificação das interfaces | 13/04/2018 |
| 0.6 | Diego  Mamoru  Tiago | Terceira correção dos requisitos | 14/04/2018 |
| 1.0 | Diego  Mamoru  Tiago | Finalização do documento de requisitos | 18/04/2018 |
| 1.1 | Diego  Mamoru  Tiago | Inserção dos casos de Uso, Modelo e Visão de Negócios | 01/05/2018 |
| 1.2 | Diego  Mamoru  Tiago | Inserção da Arquitetura inicial do sistema | 04/05/2018 |
| 1.3 | Diego  Mamoru  Tiago | Primeira Revisão do Workflow de Requisitos | 09/05/2018 |
| 1.4 | Diego  Mamoru  Tiago | Segunda Revisão do Workflow de Requisitos | 10/05/2018 |
| 2.0 | Diego  Mamoru  Tiago | Finalização do documento para a segunda entrega | 10/05/2018 |
| 2.1 | Diego  Mamoru  Tiago | Correção da versão 1.0 | 14/05/2018 |
| 2.2 | Fernando | Revisão e ajuste de diagramas | 30/05/2018 |
| 3.0 | Fernando | Finalização Workflow Análise | 04/06/2018 |
| 3.1 | Fernando | Correção Worklow Análise | 15/06/2018 |
| 3.2 | Diego  Fernando  Mamoru  Tiago | Correção Worklow Análise | 16/06/2018 |
| 3.2 | Diego  Fernando  Mamoru  Tiago | Workflow Projeto | 16/06/2018 |
| 4.0 | Diego  Fernando  Mamoru  Tiago | Finalização Workflow Projeto | 17/06/2018 |

1. **Introdução**

## Objetivos do Documento

Este documento tem como objetivo, especificar os requisitos relacionados ao sistema para otimização de arquitetura de linha de produto de software intitulado OPLA-Tool. Além disso, servirá como guia, fornecendo para arquitetos de software, acadêmicos e pesquisadores as informações necessárias para entendimento e uso da ferramenta.

## Escopo do Produto

A ferramenta OPLA-Tool proposta pela professora Drª Thelma Colanzi Elita Lopes, é um sistema que tem como objetivo, instanciar uma abordagem de otimização multiobjetivo automatizada para mensurar e aperfeiçoar um projeto de PLA (*Product Line Architecture*).

## Público-Alvo

Arquitetos de software, acadêmicos e pesquisadores.

## Definições, Acrônimos e Abreviações

* PLA – *Product Line Architecture*
* RF – Requisito funcional
* RNF – Requisito não funcional
* FUNC – Funcionalidade macro que agrupa requesitos funcionais de um mesmo contexto
* MOEA – *Multi-objective Evolutionary Algorithms*
* Arquivo XMI - conjunto de 3 arquivos (dos tipos .notation, .di e .uml) editados no Papyrus para servir como entrada para a ferramenta opla-tool.

O arquivo XMI será o alvo da otimização realizada pelo algoritmo de busca.

* MOA4PLA - *Multi-Objective Approach for Product-Line Architecture Design*.
* Soluções não dominadas - Soluções geradas pelo processo otimização multiobjetivo e consideradas melhores que outras soluções em relação a certas características.

## Convenções

Por convenção, a referência aos requisitos é feita por meio do identificador seguido do nome do requisito, [identificador do requisito - nome do requisito].

Os requisitos devem ser identificados com um identificador único. A numeração inicia com o identificador [RF001] ou [RNF001] e prossegue sendo incrementada à medida que forem surgindo novos requisitos.

## Glossário

Para o melhor entendimento por parte do leitor quanto aos termos utilizados no decorrer do documento, foi especificado um glossário contendo os significados desses termos dentro do contexto em que o sistema OPLA-Tool se insere.

O glossário é apresentado a seguir:

* **Arquiteto de Software**: Profissional que irá utilizar a ferramenta para gerar arquiteturas de linha de produto de software otimizadas para seu uso.
* **Pesquisador**: Profissional da área acadêmica e/ou científica que irá utilizar a ferramenta com fins experimentais para comparações de resultados, agregando conhecimento acadêmico e/ou científico.
* **PLA**: Arquitetura de linha de produto de software construída na ferramenta Papyrus seguindo as especificações descritas no decorrer deste documento e quer serve como entrada do sistema OPLA-Tool.
* **População**: Conjunto de arquiteturas de linha de produto de software (soluções) otimizadas, seguindo os conceitos de teoria da evolução (algoritmo genético).
* **Solução**: Arquitetura de linha de produto de software gerada após o processo de otimização de uma PLA (produto da execução da ferramenta).
* **Função Objetivo**: Medida utilizada para mensurar a qualidade das soluções geradas.
* **Métricas**: Formas para avaliação de características específicas de uma solução.
* **Algoritmo Evolutivo Multiobjetivo**: Algoritmo baseado na teoria da teoria da evolução por seleção natural, onde, indivíduos com melhores características são mantidos na população.

Leva-se em conta a otimização de várias características da PLA (modularização de características, reúso, coesão, acoplamento, etc), sendo de escolha do arquiteto de software/pesquisador.

## Referências

COLANZI, Thelma Elita. **Uma abordagem de otimização multiobjetivo para projeto arquitetural de linha de produto de software.** 2014. 215 f. Tese (Doutorado em Ciência da Computação) – Programa de Pós, Universidade Federal do Paraná, Curitiba, 2014.

FÉDERLE, Édipo Luis. **Uma Ferramenta de apoio ao Projeto Arquitetural de Linha de Produto de Software Baseado em Busca.** 2014. 142 f. Dissertação (Mestrado) - Curso de Informática, Universidade Federal do Paraná, Curitiba, 2014.

GUIZZO, Giovanni. **Uso de padrões em projeto arquitetural baseado em busca de linha de produto de software.** Curitiba, 2014.

OLIVEIRA JUNIOR, Edson Alves Oliveira de. **SystEM-PLA: um método sistemático para avaliação de arquitetura de linha de produto de software baseada em UML**. 2010. 281 f. Tese (Doutorado) – Cursos de Ciência da Computação e Matemática Computacional, Universidade de São Paulo, São Paulo, 2010.

MARIANI, Thaina. **Preservando o estilo arquitetural no projeto baseado em busca de linha de produto de software.** Universidade Federal do Paraná (UFPR), Curitiba, 2015.

1. **Visão Geral**

## Perspectiva do Produto

A ferramenta OPLA-Tool foi proposta para viabilizar a automatização da abordagem de otimização para projeto de PLA chamada MOA4PLA.

Tal abordagem visa dar suporte ao arquiteto de software na obtenção de PLAs de melhor qualidade em termos de modularização de características, reuso e outros princípios básicos de projeto. Para isso, faz uso de algoritmos evolutivos multiobjetivos.

## Funcionalidade do Produto

* FUNC001 Representação da PLA – é subdividida em: Formato de arquivo de entrada e Codificação da entrada;
* FUNC002 Otimização Multiobjetivo – compõe os módulos OPLA-Core e OPLA-Patterns e tem as seguintes subdivisões: Configurações Gerais, Configurações da Execução, Otimização utilizando Padrões de Projeto, Resultados, Comparações de Execuções e Logs;
* FUNC003 Decodificação das alternativas de PLA – é composto pela decodificação da Saída.

## Usuários

* Arquiteto de Software
* Acadêmico
* Pesquisador

## Ambiente Operacional

Sistema multiplataforma executando sem a necessidade de acesso à internet, sendo exigido ao usuário apenas um computador para ambiente experimental.

## Restrições de Projeto e Implementação

Computador que possua configuração mínima com:

* Processador de 4 cores;
* 4GB de memória RAM;
* 15GB de memória de disco disponível (sem contar o espaço para o sistema operacional);
* Ubuntu com ambiente gráfico ou Windows;
* Java 8 ou superior;
* Ferramenta Papyrus;
* Netbeans ou Eclipse;

## Documentação do Usuário

Documentação da utilização prática da ferramenta existente de forma básica, com ênfase em artigos e publicações de caráter acadêmico e científico, não sendo de fácil entendimento para o público em geral.

## Suposições e Dependências

A seguir serão apresentados alguns fatores que afetam os recursos que este documento inclui:

* A configuração do computador inferior aos requisitos mínimos para execução apropriada do Netbeans e Eclipse, poderá impossibilitar o desenvolvimento da ferramenta;
* Versão incompatível do Netbeans, Eclipse, Java ou a inexistência de quaisquer dependências como bibliotecas ou arquivos de configuração, poderá gerar inconsistências no sistema que podem interferir diretamente no desenvolvimento e/ou execução da ferramenta;
* Arquivo de entrada em formato incorreto (o formato deve ser XMI editado no Papyrus).

1. **Especificação das Interfaces Externas**

## Requisitos de Interface Externa

* + 1. **Interfaces do Usuário**
       1. **Configurações Gerais**

A tela apresentada na Figura 1 fornecerá dados relativos às configurações gerais do sistema, bem como os caminhos referentes à utilização e locais de perfis, e locais dos arquivos XMI, assim como a exibição do atual conteúdo do arquivo de configuração.

* Na seção *Profiles Configuration* deve-se selecionar quais arquivos de perfis serão utilizados e seus respectivos caminhos absolutos;
* Na seção Template Configuration deve-se especificar o diretório dos arquivos XMI;
* Na seção *Manipulation Directory* deve-se especificar o diretório para manipulação de arquivos temporários (gerados durante a execução de experimentos).

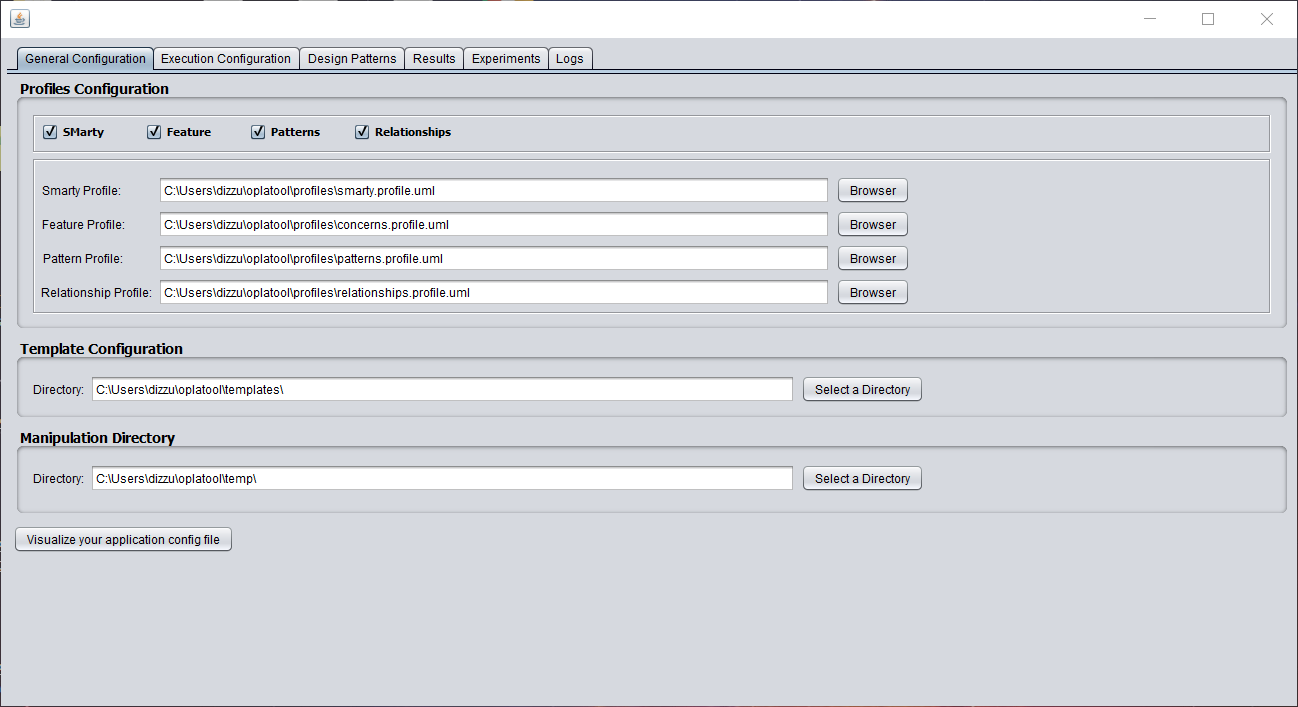


Figura 1 - Tela de Configurações Gerais

* + - 1. **Configurações de Execução**

A tela apresentada na Figura 2 ilustra a aba *Execution Configuration* que concentrará configurações relativas à execução de experimentos.

Será possível escolher uma MOEA *(Multiobjective Evolutionary Algorithm)*, assim como, funções objetivo, operadores de mutação, etc..

* Na seção *Settings*, o usuário deverá selecionar o algoritmo multiobjetivo e seus respectivos parâmetros, como: (número de rodadas, número de iterações, o tamanho da população e o tamanho do arquivo);
* Na seção *Objective Functions*, o usuário deverá escolher quais funções objetivo serão utilizadas;
* Na seção *Operators*, o usuário deverá selecionar se deseja utilizar o operador de mutação e/ou o operador de cruzamento, assim como, suas respectivas probabilidades de ocorrência;
* A seção *Mutation Operators,* deverá ser habilitada somente a escolha do operador de mutação no checkbox. Esses operadores definem como ocorrerá a mutação de acordo com propriedades da PLA;
* Na seção *Input Architecture*,deverá ser especificado o caminho absoluto do arquivo **.uml** referente à PLA escolhida pelo usuário;
* Na seção *Output Directory*, deverá ser especificado o caminho absoluto para armazenamento dos resultados obtidos pelas execuções.

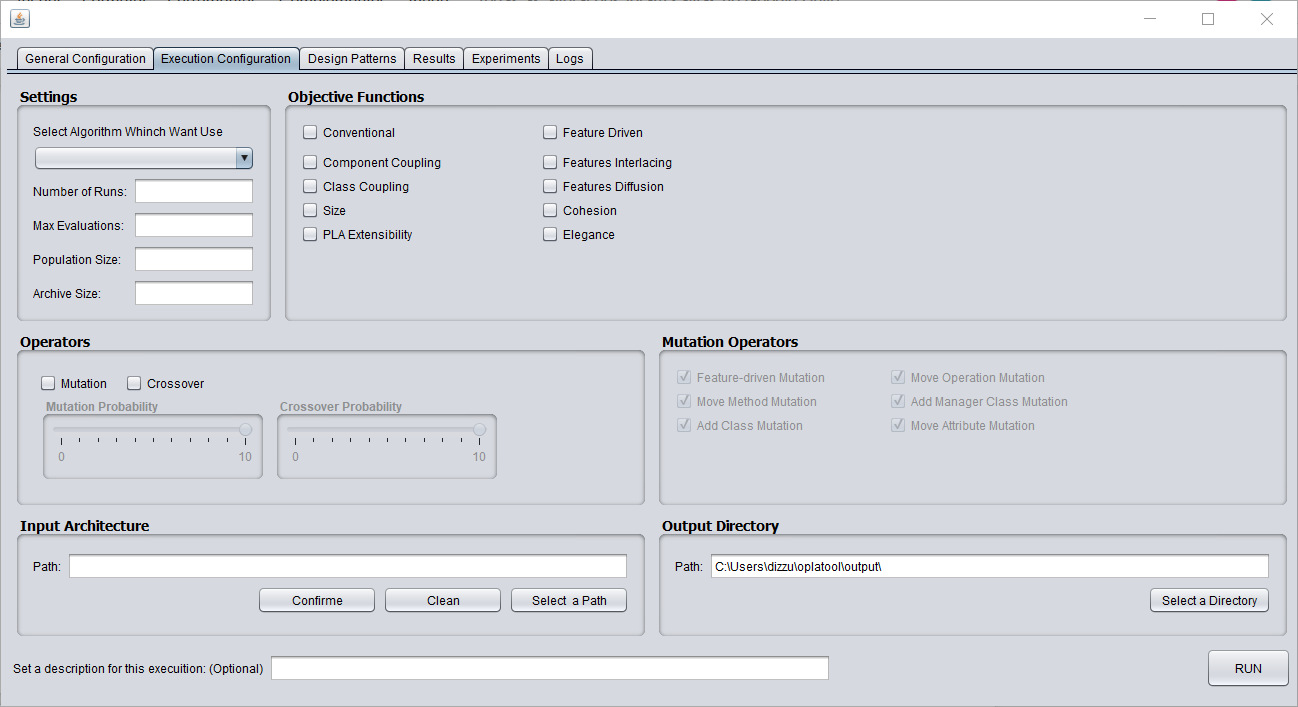


Figura 2 - Tela de Configurações da Execução

* + - 1. **Padrão de Projeto (*Design Pattern*)**

Na tela apresentada na Figura 3 será possível selecionar padrões de projetos deseja-se utilizar no processo de otimização. Três opções estão disponíveis, sendo elas: *Mediator*, *Strategy* e *Bridge*.

Uma vez selecionado um ou mais padrões de projetos, será exibida uma opção onde o usuário poderá selecionar opções para customização (utilização randômica ou somente em elementos com mesmos padrões de projeto ou nenhum) de padrões de projeto;

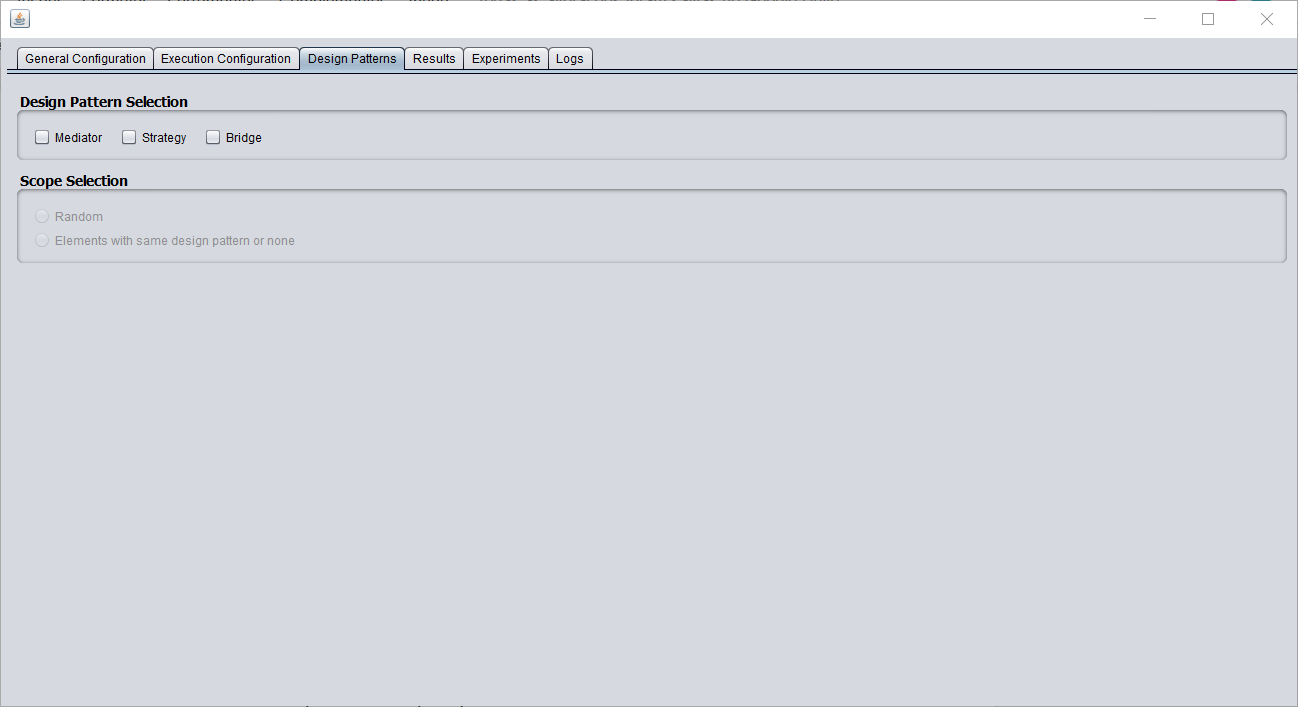


Figura 3 - Tela de Padrões de Projeto

* + - 1. **Resultados**

Na tela da Figura 4 (tela de resultados) será possível ter acesso aos resultados da execução. Serão apresentadas todas as execuções realizadas pela ferramenta OPLA-Tool até o momento.

* Na seção *Executions*, o usuário deverá selecionar uma execução;
* Na seção *Runs*, deverão ser carregadas automaticamente as rodadas da execução selecionada;
* No *combobox* *Solution*, deverão ser carregadas soluções referentes à execução selecionada e o usuário deve escolher uma delas (filtro);
* No *combobox Objective Solution*, deverão ser carregadas funções objetivo utilizadas pela execução selecionada. Abaixo do *combobox* serão carregados valores referente as funções objetivo relativos à solução selecionada no *combobox Objective Solution*;
* Na seção entre o *combobox Objective Solution* e o botão *Non-Dominated Solutions* o serão carregados valores de métricas que compõem a função objetivo selecionada no *combobox*;
* Ao clicar em *Non-Dominated Solutions* deverá ser carregada uma janela contendo informações sobre soluções não dominadas.

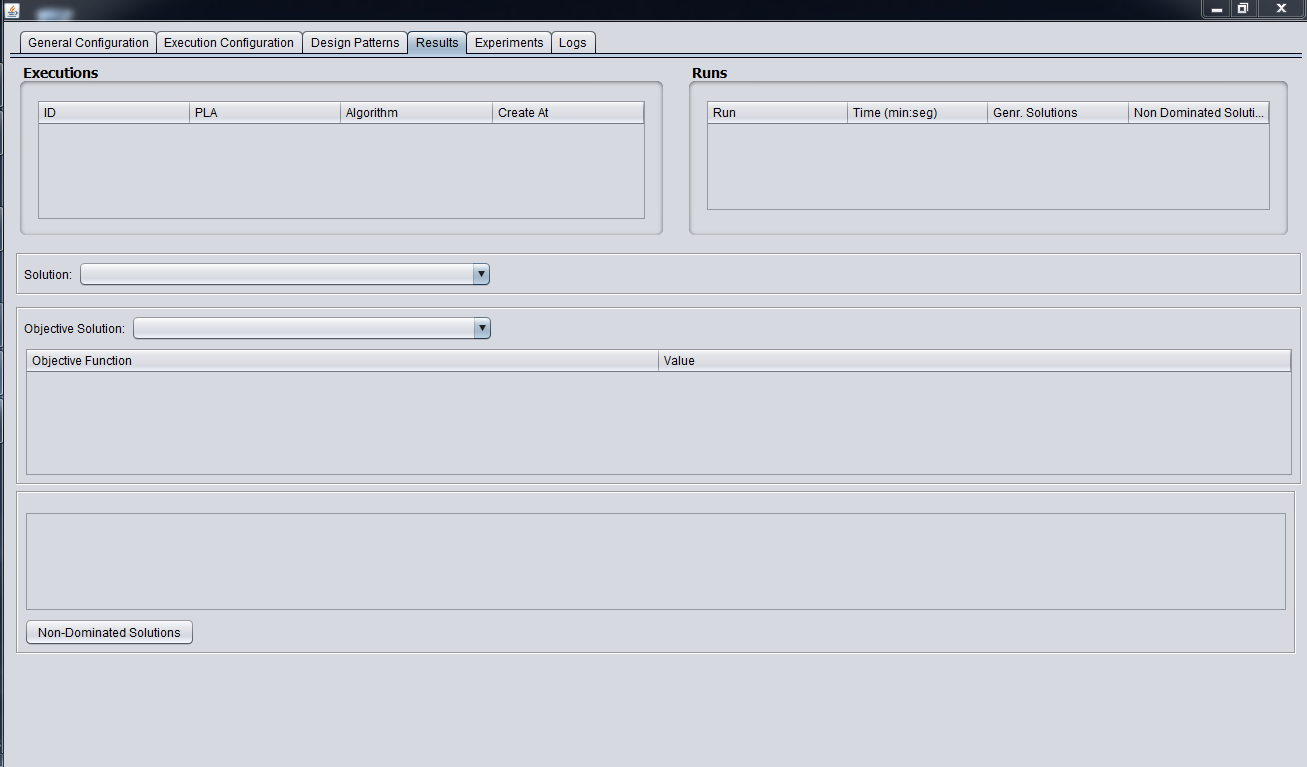


Figura 4 - Tela de Resultados

* + - 1. **Experimentos**

A tela apresentada na Figura 5 (tela de experimentos) terá como objetivo fornecer uma forma rápida e fácil de realizar comparações entre as execuções.

* Na parte superior da tela o sistema mostra o histórico das execuções em uma tabela e o usuário deverá selecionar execuções a serem comparadas;
* Na seção *Soluctions in the Search Space* o usuário deverá marcar funções objetivo recuperadas, em seguida clicar no botão *Generate Chart*. Com isso o sistema montará um gráfico comparando as soluções no espaço de busca;
* Na seção *Euclidian Distance* o sistema montará um gráfico sobre números de soluções por distância euclidiana;
* Na seção *Hypervolume* será disponibilizada a opção para calcular o *hypervolume* com dados normalizados ou não.

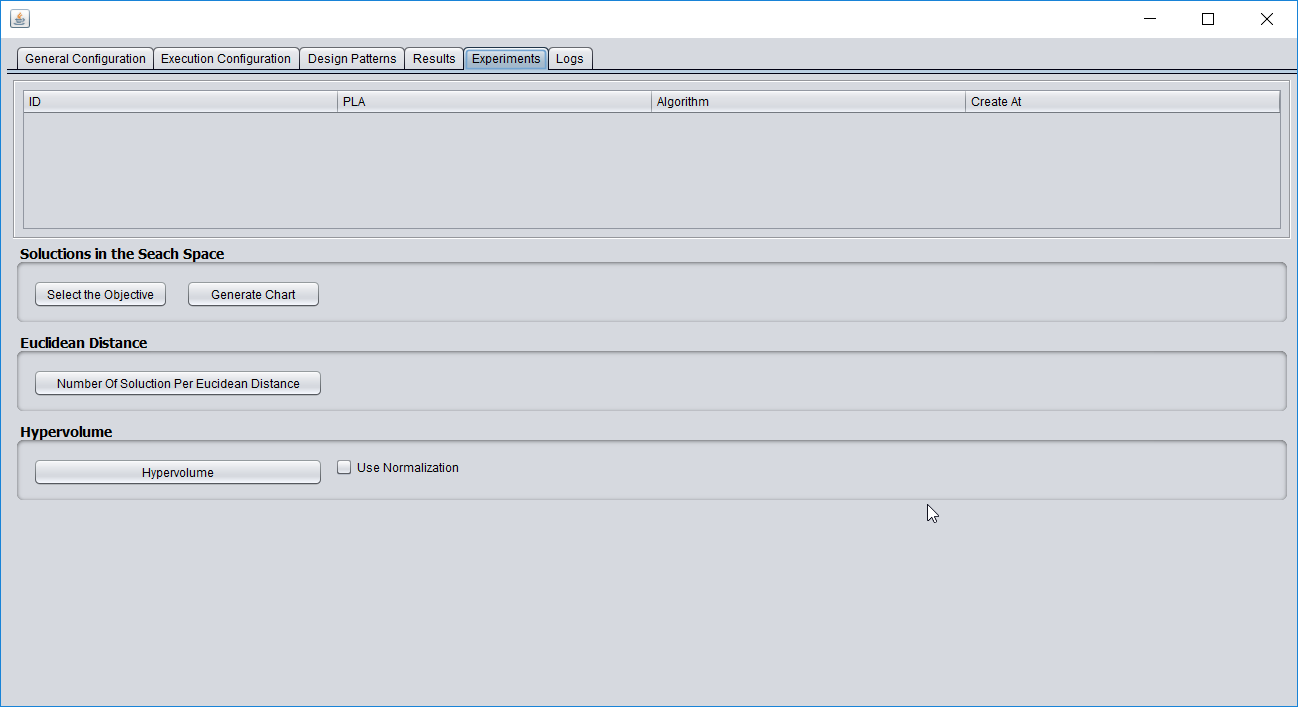


Figura 5 - Tela de Experimentos

* + - 1. **Logs**

O sistema conterá uma aba chamada Logs, que mostrará informações pertinentes aos procedimentos realizados pela ferramenta, criando um histórico de datas e horários de execuções.

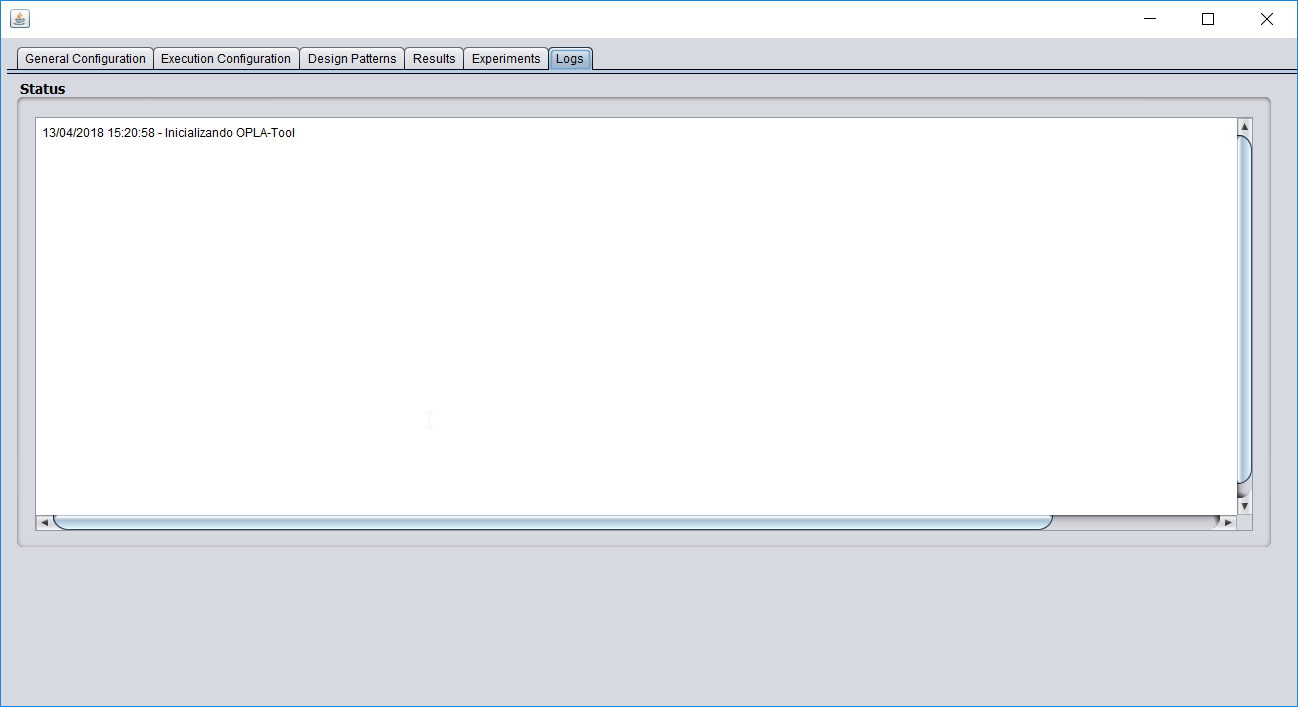


Figura 6 - Tela de Logs do Sistema OPLA-Tool

* + 1. **Interfaces de Hardware**

Dispositivos desktops.

* + 1. **Interfaces de Software**

Nessa seção, são apresentados softwares e arquivos externos envolvidos com a OPLA-Tool.

* Papyrus para manipulação da entrada e saída.
* Arquivo XMI gerado pelo Papyrus para permitir o intercâmbio de metadados entre ferramentas de modelagem e repositórios de metadados em ambientes heterogêneos e distribuídos. Com o XMI, é possível descrever objetos que podem conter atributos de diferentes tipos.
* UML2 para a manipulação de modelos UML.
* Banco de Dados para o armazenamento de dados.
  + 1. **Interfaces de Comunicação**

Não necessita de comunicação com a rede. A ferramenta é offline.

1. **Requisitos Funcionais**

Os requisitos funcionais deste documento estão organizados em termos de alguns módulos da OPLA-Tool, sendo eles: OPLA-Encoding, OPLA-Core, OPLA-Patterns e OPLA-Decoding, que serão representados por: Representação da PLA, Otimização Multiobjetivo, Otimização utilizando Padrões de Projeto e decodificação das alternativas de PLA, respectivamente.

A seguir serão apresentadas as principais funcionalidades da OPLA-Tool e suas subdivisões. Nessas subdivisões, ao se referir a algum componente relativo à interface, será utilizado termos em inglês, pois, a ferramenta foi desenvolvida na língua universal inglesa.

* 1. **FUNC001 Representação da PLA**
     1. **RF001 Formato do arquivo de entrada**

A ferramenta OPLA-Tool deverá receber um arquivo de entrada fornecido pelo usuário exclusivamente no formato XMI. O arquivo deverá ser modelado na ferramenta Papyrus e baseado na representação do metamodelo proposto por Colanzi (2014), ilustrado na Figura 7.

Uma PLA modelada com a ferramenta Papyrus consiste em um conjunto de três arquivos, sendo eles:

* **.uml**: Armazena elementos da arquitetura propriamente ditos (classes, métodos, relacionamentos, etc);
* **.di**: Armazena informações gerais sobre o diagrama;
* **.notation**: Armazena a representação gráfica da arquitetura.

Destes 3(três) arquivos, o OPLA-Encoding utilizará o **.uml** para gerar o metamodelo proposto por (Colanzi,2014). Os arquivos .di e .notation deverão ser incluídos no mesmo diretório.

O projeto de PLA modelado no Papyrus deve conter os seguintes perfis UML (*Unified Modeling Language*):

* *SMarty Profile*: O SMartyProfile é composto por estereótipos e meta-atributos destinados a representar uma variabilidade em modelos UML de LPS. Seu objetivo é fornecer uma extensão da UML para representação gráfica do conceito de variabilidade (Oliveira Junior, 2010);
* *Concern Profile*: Corresponde às características (*features*) da PLA alvo da otimização;
* *Relationships e Patterns Profiles*: É composto por estereótipos utilizados para caracterizar escopos onde é propícia a aplicação de padrões de projeto. A aplicação destes padrões é restrita ao módulo OPLA-Patterns, no entanto, a falta destes arquivis poderá impactar no resultado final não do execução (Guizzo, 2014).

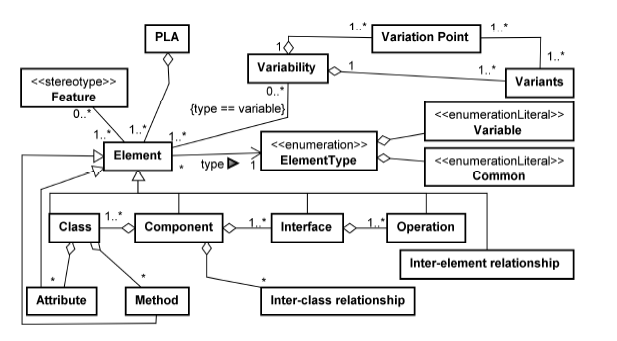


Figura 7 - Metamodelo da PLA (Colanzi, 2014)

* + 1. **RF002 Codificação da Entrada**

O módulo OPLA-Encoding deverá receber como entrada uma PLA modelada no Papyrus seguindo o formato descrito no RF001, com isso, deverá realizar o *parser* de cada um dos elementos encontrados (conforme ilustrado na Figura 8), na seguinte ordem:

(a) Ler e instanciar todos os pacotes;

(b) Para cada pacote devem ser lidas e instanciadas suas classes e interfaces, juntamente com seus respectivos atributos, métodos e operações;

(c) Classes e interfaces não pertencentes a pacotes devem ser lidas e instanciadas;

(d) Instanciar os relacionamentos existentes entre os elementos arquiteturais;

(e) Ler e instanciar as variabilidades e os pontos de variação da PLA. No final deste processo, um objeto do tipo *Architecture* deve ser gerado como saída.

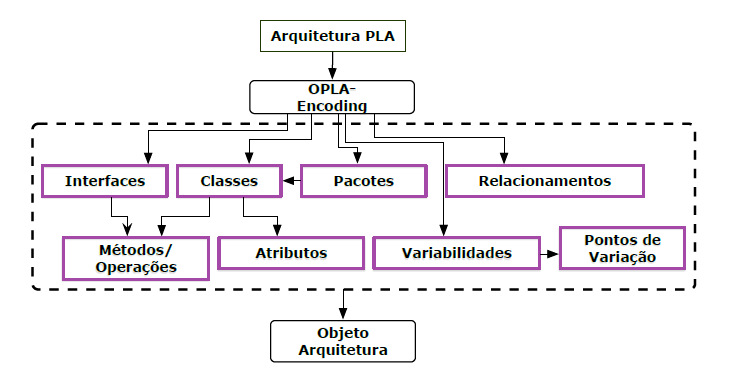


Figura 8 - Processo de leitura realizado pelo módulo OPLA-Encoding (Féderle, 2014)

* 1. **FUNC002 Otimização Multiobjetivo**

A funcionalidade Otimização Multiobjetivo, é responsável pela execução e está dividida em dois módulos, sendo:

* OPLA-Core: Módulo que será responsável por executar os experimentos utilizando algoritmos evolutivos multiobjetivos.
* OPLA-Patterns: Módulo que se integrará ao OPLA-Core para permitir a aplicação do operador de mutação responsável por aplicar padrões de projeto no projeto de PLA que está sendo otimizado.

Para a execução do OPLA-Core o usuário precisará vincular configurações relativas à entrada e ao algoritmo a ser executado. Além disso, também são necessárias informações adicionais como operadores a serem utilizados e os critérios de execução.

Tais configurações que são necessárias para o OPLA-Core e serão apresentadas nos tópicos Configurações Gerais e Configuração da Execução:

* + 1. **RF003 – Configurações Gerais (interface do item 3.1.1.1)**

A configuração geral permitirá que o usuário escolha os perfis de variabilidade a serem utilizados.

O usuário deverá informar o caminho absoluto desses arquivos de configuração, bem como, o caminho do arquivo XMI e o caminho absoluto do diretório de manipulação temporário que será utilizado internamente pela ferramenta.

Esse requisito é dividido na seguinte forma:

* *Profiles Configuration*: Selecionar perfis a serem aplicados e caminho absoluto do arquivo de configuração do perfil.
* *Template Configuration*: Selecionar o caminho absoluto de armazenamento das informações referente ao template (arquivos XMI) a ser utilizado;
* *Manipulation Directory*: Selecionar o caminho absoluto de armazenamento da pasta temporária usada para manipulação interna de arquivos da ferramenta.

* + 1. **RF004 – Configuração da Execução (interface do item 3.1.1.2)**

Este requisito tem como objetivo instanciar as configurações necessárias para realizar a execução das tarefas de otimização do OPLA-Core.

O usuário deverá informar o algoritmo evolutivo, operadores de mutação, probabilidade de ocorrência da mutação, métricas (*objective functions*) que serão levadas em conta no processo de avaliação da PLA, caminho para o arquivo de entrada a ser processado (arquivo XMI) e onde caminho de armazenamento para saídas geradas pela ferramenta.

Este requisito é dividido da seguinte forma:

* *Settings*: Deverá ser selecionado o algoritmo evolutivo e multiobjetivo a ser utilizado no processo de otimização e parâmetros da função selecionada, tais como, número de execuções, avaliações máximas, tamanho da população e tamanho do arquivo;
* *Objective Functions*: Deverá exibir funções objetivo que podem ser consideradas na verificação da qualidade das soluções. O usuário poderá selecionar uma ou mais dependendo do critério de avaliação escolhida pelo usuário;
* *Operators*: Deverá selecionar os operadores que devem ser utilizados (mutação, cruzamento ou ambos) e suas respectivas probabilidades de ocorrência;
* *Mutation Operators*: No caso da utilização do operador de mutação, deverá ser selecionado os operadores de mutação que serão aplicados sobre o processo de otimização da PLA (como deve ser realizada a mutação dos indivíduos);
* *Input Architecture*: Deverá ser especificado o caminho absoluto da entrada da PLA;
* *Output Directory*: Deverá ser especificado o caminho absoluto da saída da PLA otimizada;

* + 1. **RF005 – Otimização utilizando Padrões de Projeto (interface do item 3.1.1.3)**

Este requisito será encarregado de armazenar operadores de mutação para aplicar os padrões de projeto Strategy, Bridge, Façade e Mediator em arquiteturas de software (incluindo PLA) na atividade de Otimização Multiobjetivo da MOA4PLA (Mariani, 2015).

O OPLA-Patterns deve se comunicar com o módulo OPLA-Core para possibilitar que seus operadores de mutação sejam utilizados em conjunto com os operadores já existentes no módulo. Essa funcionalidade deverá conter as seguintes operações:

* Design Pattern Selection: Selecionar o padrão de projeto utilizado pelo OPLA-Patterns para adicionar operadores de mutação, caso o arquiteto assim queira.
* Scope Selection: O usuário poderá ativar o módulo randômico, onde o sistema deve escolher de forma aleatória qual padrão será adotado. Assim como poderá definir os elementos com mesmo padrão ou nenhum.

* + 1. **RF006 - Resultados (interface do item 3.1.1.4)**

Após a execução da otimização da PLA pelo módulo OPLA-Core, os resultados deverão ser informados ao usuário de uma maneira detalhada, com informação referente ao tempo despendido por cada execução, lista com alternativas de PLA resultantes e suas respectivas identificações, algoritmos utilizados, nomes e data em que foram criadas.

Também deve ser possível verificar o valor das funções objetivo filtrando pelo ID da solução. Este requisito é dividido da seguinte maneira:

* *Executions*: Deverá haver uma descrição dos resultados que apresentará as informações sobre cada uma das execuções realizadas pela OPLA-Tool até o momento. As informações relacionadas à execução que deverão ser exibidas ao usuário são: Identificador da execução, nome da PLA, algoritmo e a data;
* *Runs*: A descrição das execuções terá a função de listar todas as rodadas do algoritmo. Essa listagem deve incluir as seguintes informações: identificador da rodada, tempo que a rodada levou para ser executado, número de soluções encontradas e número de soluções não dominadas.

O valor apresentado será o mesmo para todas as rodadas, pois, se refere às soluções não dominadas da execução, e não da rodada;

* Tabela de funções objetivo: Deverão ser listados valores obtidos das funções objetivos de uma determinada solução selecionada pelo usuário;
* *Non-Dominated Solutions*: Deverão ser selecionadas e apresentadas ao usuário, soluções não dominadas através da curva de Pareto que divide soluções dominadas das não dominadas.

* + 1. **RF007 – Comparação de Execuções (interface do item 3.1.1.5)**

A ferramenta deverá fornecer ao arquiteto de software uma maneira de se realizar comparações entre as execuções de forma rápida e fácil. O intuito é a exibição de soluções não dominadas no espaço de busca.

Para a realização da comparação entre as execuções, é necessário que estas tenham as seguintes características:

1. Ambas as execuções devem utilizar o mesmo modelo de avaliação (mesmas funções objetivo);
2. Ambas as execuções devem ser referentes à mesma PLA; e
3. Somente duas funções de fitness podem ser selecionadas por vez.

A comparação deverá ser feita por meio da exibição de um gráfico da distância euclidiana no formato de barra ou de linha.

* + 1. **RF008 - Logs (interface do item 3.1.1.6)**

A ferramenta deverá ter um log com o histórico das rotinas executadas para fácil visualização do usuário, indicando possíveis erros quando identificados.

Esses eventos começam desde a inicialização do aplicativo e devem conter também informações sobre resultado do processo, por exemplo, se concluiu em erro ou sucesso.

* 1. **FUNC003 Decodificação das alternativas de PLA**
     1. **RF009 Decodificação da Saída**

As soluções não dominadas contidas na saída gerada pelos algoritmos de busca no módulo OPLA-Core, precisam ser convertidas para um formato legível ao arquiteto de software (formato XMI).

Essa decodificação deverá ser realizada para que os usuários possam visualizar e editar a PLA no Papyrus, podendo depois, reaproveitar a solução como entrada para outra otimização.

O processo de decodificação deve seguir o esquema mostrado na Figura 9, apresentando os passos a seguir (a numeração apresentada no texto corresponde à numeração da Figura 9):

Inicialmente, no passo 1, a arquitetura que se encontra em memória (instância do objeto *Architecture*) é fornecida como entrada para o módulo OPLA-Decoding, conforme passo 2. A primeira atividade a ser realizada, conforme passo 3, é a cópia de três arquivos (.uml, .notation e .di) denominados *templates* (arquivos XMI) e que possuem apenas o conteúdo padrão a ser utilizado na construção da nova arquitetura. Eles são, então, copiados para o diretório selecionado.

No passo 4, são copiados também os perfis utilizados pela arquitetura, como o módulo OPLA-Encoding que recebe um objeto do tipo *Architecture*, cada elemento da arquitetura é recuperado (passo 5). Para cada elemento encontrado é realizada a geração do mesmo (passo 6) sobre os arquivos XMI, exibidos no passo 7. Essa geração consiste em vários passos dependendo do tipo de elemento a ser gerado. Um aspecto importante nessa fase, é que todas as classes e interfaces devem ser geradas antes de qualquer tipo de relacionamento, visto que esses últimos estão diretamente ligados aos primeiros.

Neste processo de geração, dependendo do tipo do elemento, é usada uma classe para realizar a geração do elemento em questão. A cada mudança nesses arquivos, (.uml e .notation) é feita uma exportação para o diretório configurado pelo usuário, resultando assim, na arquitetura de saída (passo 8).

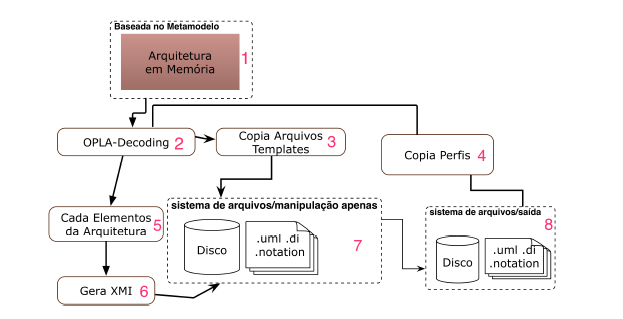


Figura 9 - Processo de Decodificação (Féderle, 2014)

1. **Requisitos Não-Funcionais**
   1. **Requisitos de Desempenho**
      1. **RNF001 Tempo para Feedback da Execução**

Os usuários deverão receber feedbacks da execução do experimento na tela de Logs no menor tempo possível (menor quantidade de minutos possíveis), a fim de, permitir a atualização do usuário sobre o andamento das execuções. Muitas vezes o tempo de execução é longo e caso não sejam fornecidos feedbacks sobre a execução, o usuário poderá tomar uma ação precipitada como a interrupção do processamento, neste caso, invalidaria todo o processamento executado até o momento da interrupção tendo este que ser reiniciado.

* 1. **Requisitos de Segurança**
     1. **RNF002 Consistência de Dados**

O sistema não deverá alterar ou remover dados originais existentes, como entradas e/ou saídas geradas por outros experimentos. A alteração de entrada ou resultado de outro experimento pode acarretar na impossibilidade de comparação entre experimentos.

* 1. **Atributos de Qualidade do Software**
     1. **RNF003 Qualidade de Software**

Para cada funcionalidade deve ser realizado um conjunto de testes de unidade e funcionais com relatório de testes.

* 1. **Usabilidade** 
     1. **RNF004 Usabilidade**

A OPLA-Tool deverá contar com uma interface intuitiva, agradável e responsiva. Essa funcionalidade será passará por avaliação onde será submetida à utilização para que sejam efetuadas coletas de opiniões.

1. **Outros Requisitos**

Não se aplica.

1. **Workflow de Requisitos**

Esta seção apresenta as atividades realizadas no Workflow de Requisitos do Processo Unificado para o Sistema OPLA-Tool. Foram desenvolvidos os seguintes artefatos na execução deste workflow: Visão de Negócio, Modelo de Objetos de Negócio, Modelo de Casos de Uso e suas descrições e Modelo da Arquitetura Inicial do Sistema.

* 1. **Visão de Negócio**

A OPLA-Tool é um sistema que tem como objetivo instanciar uma abordagem de otimização multiobjetivo automatizada para mensurar e aperfeiçoar um projeto de PLA (*Product Line Architecture*), concedendo apoio ao arquiteto de software ou pesquisador da área.

Como dito na Seção 2.2, as principais funcionalidades do produto são:

* **FUNC001 OPLA-Encoding**: Conversão do arquivo de entrada no formato especificado pelo metamodelo.
* **FUNC002 OPLA-Core e OPLA-Patterns**: Seleção de configurações do projeto e realização de execução do experimento de acordo com suas necessidades.
* **FUNC003 OPLA-Decoding**: Responsavel pela transformação da saída do sistema em um formato reconhecido pelo Papyrus.

O Sistema OPLA-Tool possui relação direta com os arquivos XMI(PLA e profiles), Papyrus e um banco de dados. Com base nas entradas (arquivos XMI) deverá gerar o arquivos de hypervolume . Abaixo segue a Visão de Negócio do Sistema:

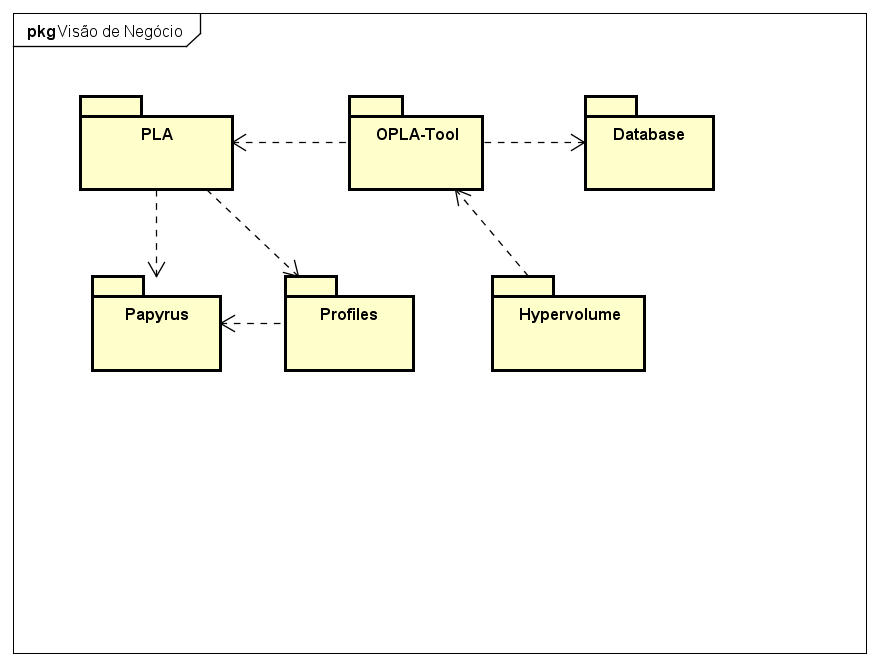


Figura 10 - Diagrama de Visão de Negócio

* 1. **Modelo de Objetos de Negócio**

A figura abaixo apresenta o modelo de objetos de negócio da OPLA-Tool, considerando as principais entidades do sistema e seus relacionamentos entre si:

* Arquiteto de Software e Pesquisador vão herdar do objeto de negócio Usuário;
* O usuário pode criar ou selecionar uma PLA de entrada;
* O usuário pode iniciar o processo de otimização;
* O usuário seleciona funções objetivo;
* As funções objetivo possuem métricas atrelas;
* A otimização é efetuada através de um algoritmo evolutivo multiobjetivo (MOA4PLA);
* A otimização recebe as funções objetivo;
* A otimização gera populações(conjunto de possíveis soluções) e dentro dessas populações, são selecionados as melhores indivíduos(soluções) com base na avaliação das métricas atreladas a cada função objetivo selecionada pelo usuário;
* A otimização gera arquivos de hypervolume;

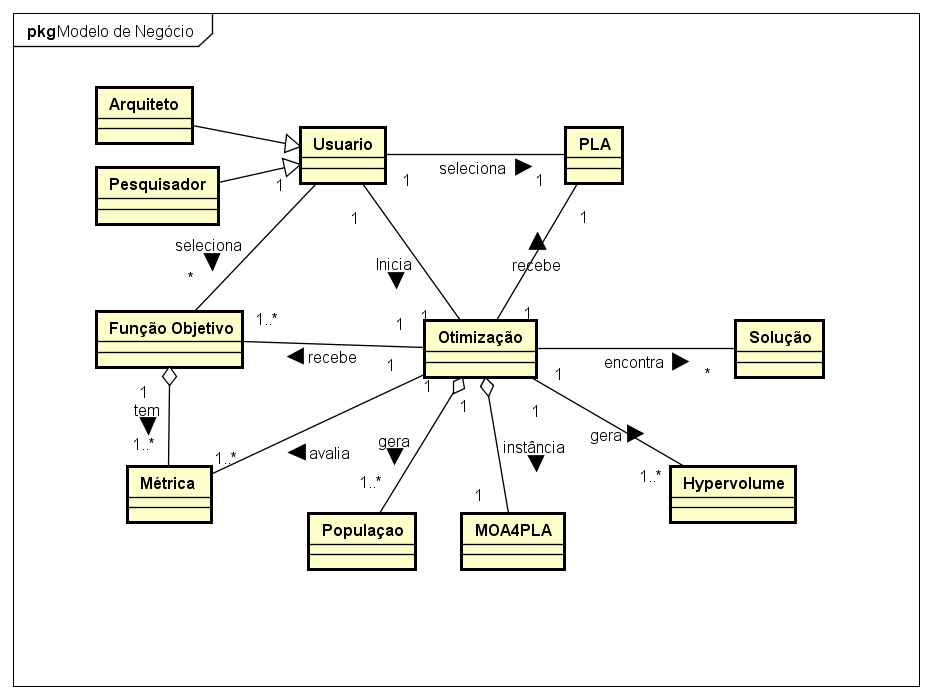


Figura 11 - Diagrama de Modelo de Objeto de Negócio

* 1. **Modelo de Caso de Uso**

Nesta seção são apresentados os casos de uso do sistema e logo em seguida é realizada a sua especificação.

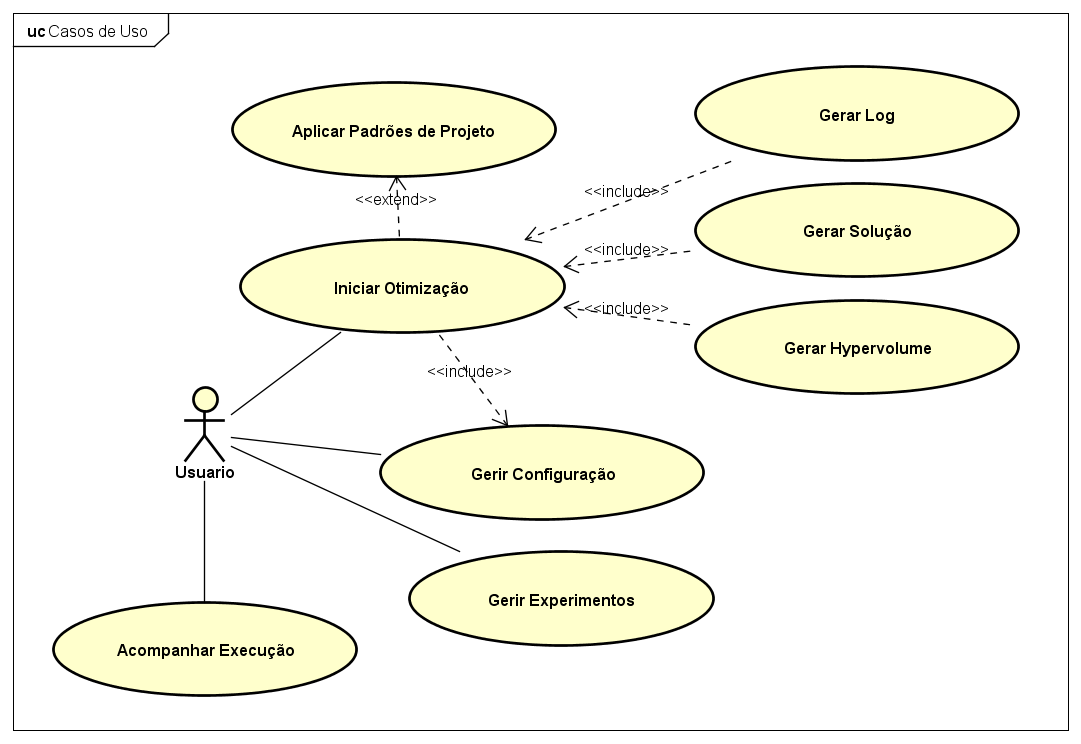


Figura 12 - Diagrama de Caso de Uso

|  |
| --- |
| **Caso de Uso:** Gerir Configuração |
| **Identificação:** UC001 |
| **Descrição:** A ferramenta permite ao usuário especificar configurações para execução, sendo:   * Configuração Geral: Caminho onde se encontra a PLA alvo da otimização, local das pastas de perfis, manipulação de arquivos, arquivos temporários e diretório para saída de resultados. * Configuração de Execução: O usuário deverá informar os parâmetros para execução da otimização, sendo: Algoritmo, tamanho da população, número de gerações, número de rodadas, operadores de mutação e cruzamento com respectivas probabilidades e funções objetivo. * Configuração para uso de Padrões de Projeto: O usuário poderá selecionar se deseja ou não que sejam aplicações padrões de projeto durante o processo de otimização. Ao decidir pelo uso de padrões, o usuário deverá selecionar um ou mais padrões dentro das opções disponíveis e definir o escopo em que o padrão será utilizado. |
| **Ator(es):** Usuário (Arquiteto de Software e Pesquisadores). |
| **Pré-Condição:** O usuário deve ter disponível uma PLA para entrada. Esta deve ser válida e gerada pelo Papyrus com os seus respectivos perfis necessários. |
| **Pós-Condição:** O sistema registra as informações no banco de dados. |
| **Curso Normal:**   1. O usuário deve navegar pelas pastas da maquina onde a sistema está instalado, procurar e especificar local das pastas e arquivos. 2. O usuário deve preencher as informações obrigatórias (campos marcados com \*), para que o sistema possa iniciar o processo de otimização. |
| **Tratamento de Exceções:**   1. Arquivo com formato incorreto especificado:    1. A ferramenta deve mostrar um aviso de erro. |

|  |
| --- |
| **Caso de Uso:** Iniciar Otimização |
| **Identificação:** UC002 |
| **Descrição:** O usuário poderá iniciar a execução do processo de otimização, com isso, o sistema deve permanecer com todas as abas, campos de entrada e botões bloqueados, sendo exibido apenas informações sobre o processo, localizados na tela de Logs, e deverá permanecer assim, até que o processo esteja completo. |
| **Ator(es):** Usuário (Arquiteto de Software e Pesquisadores). |
| **Pré-Condição:** UC001. |
| **Pós-Condição:** O sistema irá ler parâmetros de entrada fornecidos pelo usuário e efetuar o bloqueio das telas deixando apenas a tela de Logs.  O usuário então deve aguardar o término da execução. |
| **Curso Normal:**   1. O Usuário deve clicar no botão de execução e confirmar a execução do experimento; 2. O sistema abre a tela de Log e apresenta informações gerais sobre a configuração do experimento; 3. O sistema começa a execução dos experimentos e impossibilita a utilização da ferramenta (apenas habilitando a aba Logs) até que todo o processo tenha sido concluído; 4. O usuário deve aguardar o término da execução; 5. Após cada execução, são atualizadas informações na tela de Log, ao final de todo o processo, um log geral sobre as melhorias é apresentado. 6. O sistema gera diverso arquivos XMI para cada solução não dominada encontrada pelo processo de otimização; 7. O sistema gera um arquivo de hypervolume para cada solução. |
| **Tratamento de Exceções:**   1. Nenhuma função objetivo selecionada;    1. A ferramenta deve mostrar um aviso de erro e não permitir a execução do experimento. 2. Parâmetro não especificado ou com formato incorreto;    1. A ferramenta deve mostrar um aviso de erro e não permitir a execução do experimento. 3. Pasta temporária ou saída incorreta;    1. A ferramenta deve mostrar um aviso de erro e não permitir a execução do experimento. 4. Pasta temporária com conteúdo já incluído;    1. A ferramenta deve mostrar um aviso de erro e não permitir a execução do experimento. 5. Configuração ou PLA incorreto, como arquivo contendo dados errôneos, corrompido ou que não esteja na estrutura padrão do Papyrus:    1. Deve mostrar erro ao usuário e não permitir a execução; 6. A execução de uma rodada entra em loop infinito:    1. Deve mostrar erro e onde ocorreu e encerrar a execução; 7. Queda de energia ou fechamento da ferramenta:    1. Caso ocorra fechamento inesperado do sistema, o usuário deve reiniciar todo o processo. Apenas parâmetros referentes a caminhos de arquivos e pastas, ficam armazenados e são recuperados na próxima vez que abre a ferramenta. |

|  |
| --- |
| **Caso de Uso:** Acompanhar Execução |
| **Identificação:** UC003 |
| **Descrição:** A ferramenta permite que o usuário acompanhe o processo de execução por meio da tela de Logs do sistema. |
| **Ator(es):** Usuário (Arquiteto de Software e Pesquisadores). |
| **Pré-Condição:** UC001 e UC002. |
| **Pós-Condição:** Um relatório textual sobre o estado do sistema, contendo notificações das execuções e alterações efetuadas. |
| **Curso Normal:**   1. O sistema atualiza a tela de Log toda vez que a ferramenta encontrar alguma alteração importante, seja a troca de configurações ou até erros e dados básicos de soluções; 2. O usuário pode verificar o histórico de eventos em formato textual e verificar dados importantes do sistema, tais como horário de início de uma execução e configurações utilizadas para a execução. Esta verificação é realizada pela aba Logs. |
| **Tratamento de Exceções:** não se aplica. |

|  |
| --- |
| **Caso de Uso:** Gerir Soluções |
| **Identificação:** UC004 |
| **Descrição:** A ferramenta permite que o usuário verifique informações detalhadas de resultados e analise de informações sobre experimentos concluídos. |
| **Ator(es):** Usuário (Arquiteto de Software e Pesquisadores). |
| **Pré-Condição:** O usuário deve ter pelo menos uma solução gerada pela ferramenta em alguma execução finalizada. |
| **Pós-Condição:** Geração dos dados (mostrados em tabela) ou gráficos das soluções para análise do usuário. |
| **Curso Normal:**   * **Tela de Resultados:**  1. O usuário deve selecionar uma execução; 2. O sistema deve carregar as rodadas da execução selecionada; 3. O usuário deve escolher uma das rodadas; 4. O sistema deve carregar em um *combobox* as soluções referentes à execução selecionada; 5. O sistema deve carregar as funções objetivo utilizadas na execução em outro *combobox*; 6. O usuário deve então filtrar por soluções e função objetivo (os dois *combobox*); 7. O sistema carrega os valores das funções objetivo relativos à solução selecionada no *combobox*; 8. O sistema carrega os valores das métricas que compõem a função objetivo selecionada no *combobox*; 9. Ao clicar em Non-Dominated Solutions o sistema deve carregar uma nova janela contendo informações somente das soluções não dominadas.  * **Tela de Experimentos:**  1. O usuário seleciona a aba *Experiments* (interface 3.1.1.5); 2. O sistema deve mostrar o histórico das execuções em uma tabela; 3. O usuário deve selecionar as execuções a serem comparadas na tabela; 4. O sistema deve recuperar as funções objetivo utilizadas por cada execução; 5. O usuário deve marcar as funções objetivo recuperadas; 6. O usuário deve clicar em *Generate Chart*; 7. O sistema deve então montar o gráfico de acordo com as opções selecionadas pelo usuário, possibilitando comparar os resultados de duas execuções diferentes. |
| **Tratamento de Exceções:**   * Tela Results: N/A * Tela Experiments:  1. O usuário seleciona duas execuções consideradas distintas, ou seja, não utilizam as mesmas funções objetivo:    1. O sistema emitirá uma mensagem informando que as funções objetivo devem ser as mesmas para ambas as execuções. |

* 1. **Tabela de Conceitos**

O sistema OPLA-Tool não contém CRUD, não contendo Inserção, Alteração ou Remoção por parte do usuário. Devido a este fato, a tabela de conceitos não foi desenvolvida neste documento.

* 1. **Arquitetura Inicial do Sistema**

Abaixo é apresentada a arquitetura inicial da OPLA-Tool, onde é possível observar o relacionamento do OPLA-GUI (Interface do sistema), OPLA-Core, OPLA-Encoding, OPLA-Decoding e OPLA-Patterns.

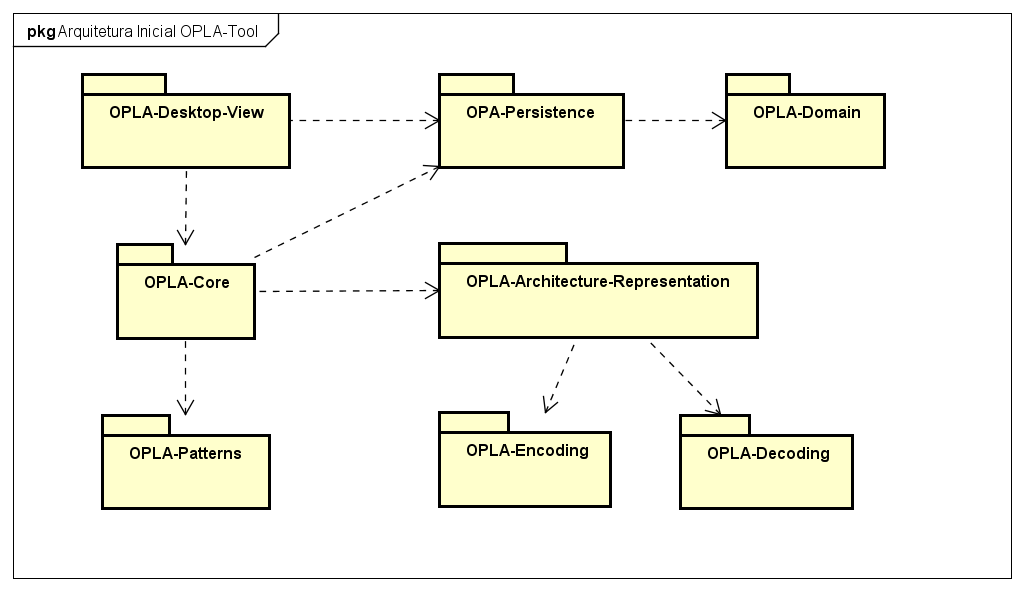


Figura 13 - Arquitetura Inicial da OPLA-Tool

1. **Workflow de Análise**

Esta seção apresenta as atividades realizadas no Workflow de Análise do Processo Unificado para o OPLA-Tool.

Foram desenvolvidos os seguintes artefatos na execução deste workflow: Realização dos Casos de Uso, Comunicação dos Casos de Uso, Análise Arquitetural e a Visão Geral do Sistema. Cada artefato produzido foi apresentado em sua subseção.

* 1. **Realização e Comunicação dos casos de uso**
     1. **UC001 – Gerir Configuração**

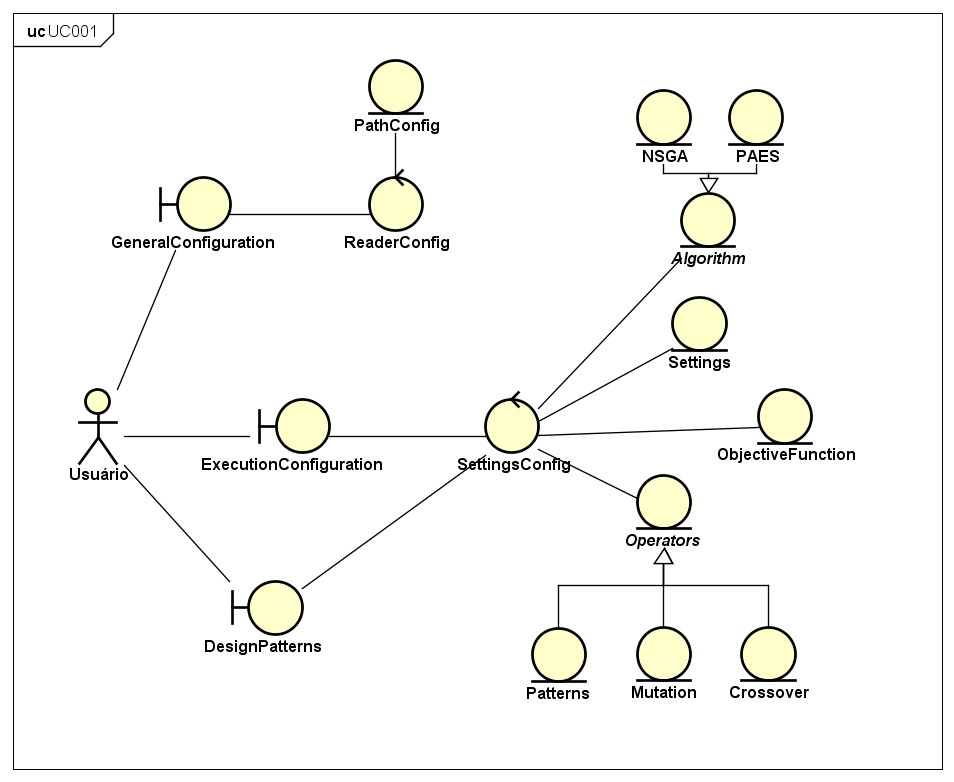
****

Figura 14 - Realização do UC001

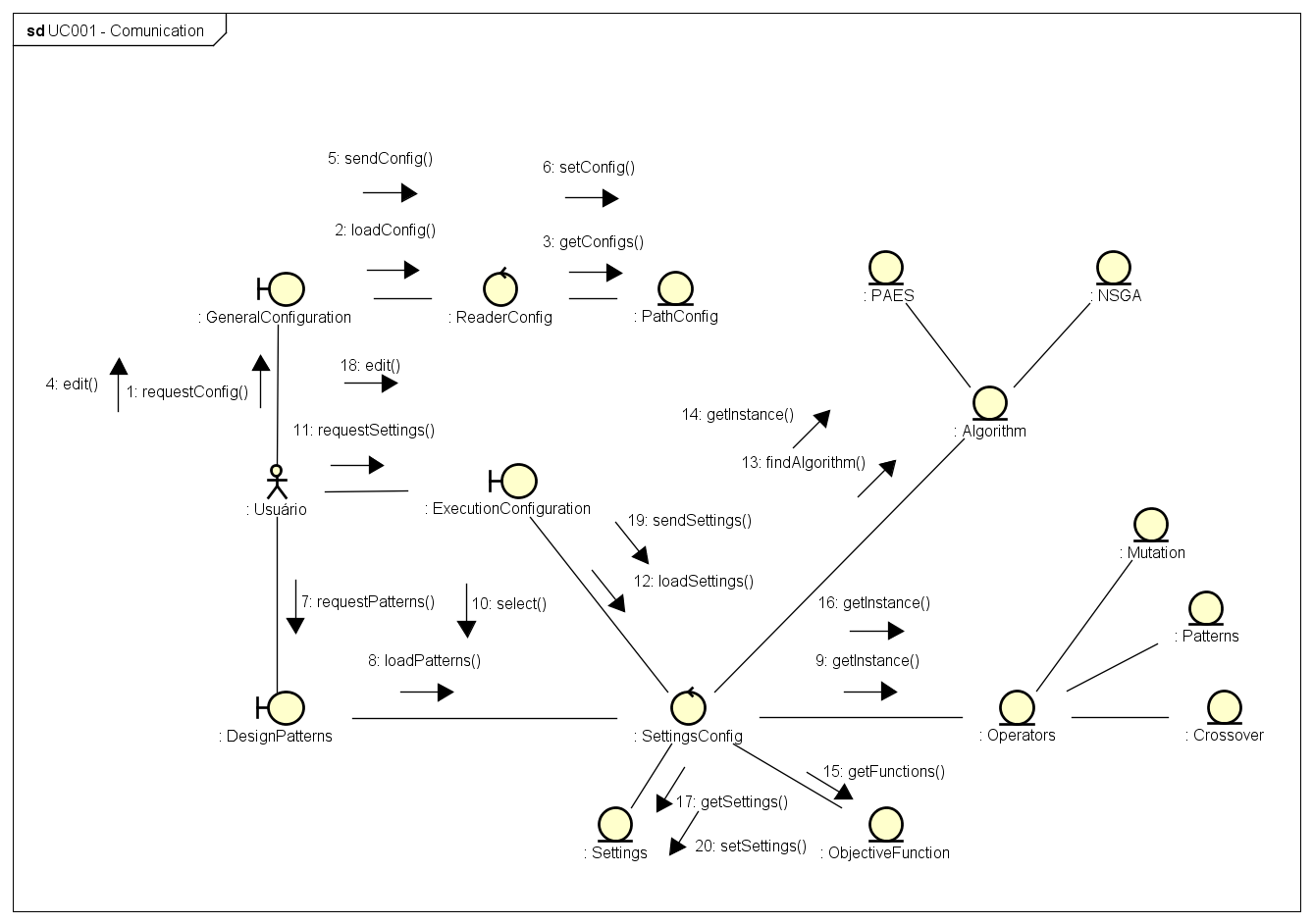


Figura 15 - Comunicação do UC001

* + 1. **UC002 – Iniciar Otimização**

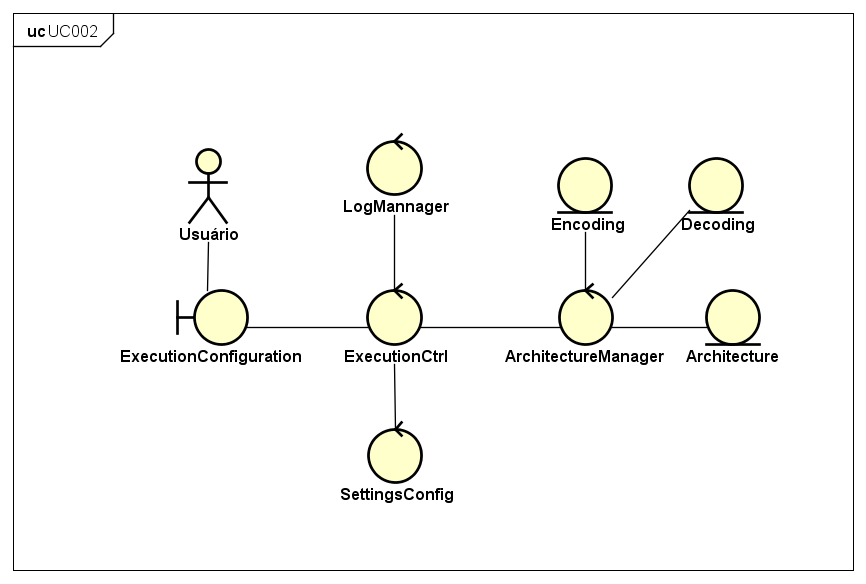


Figura 16 - Realização do UC002

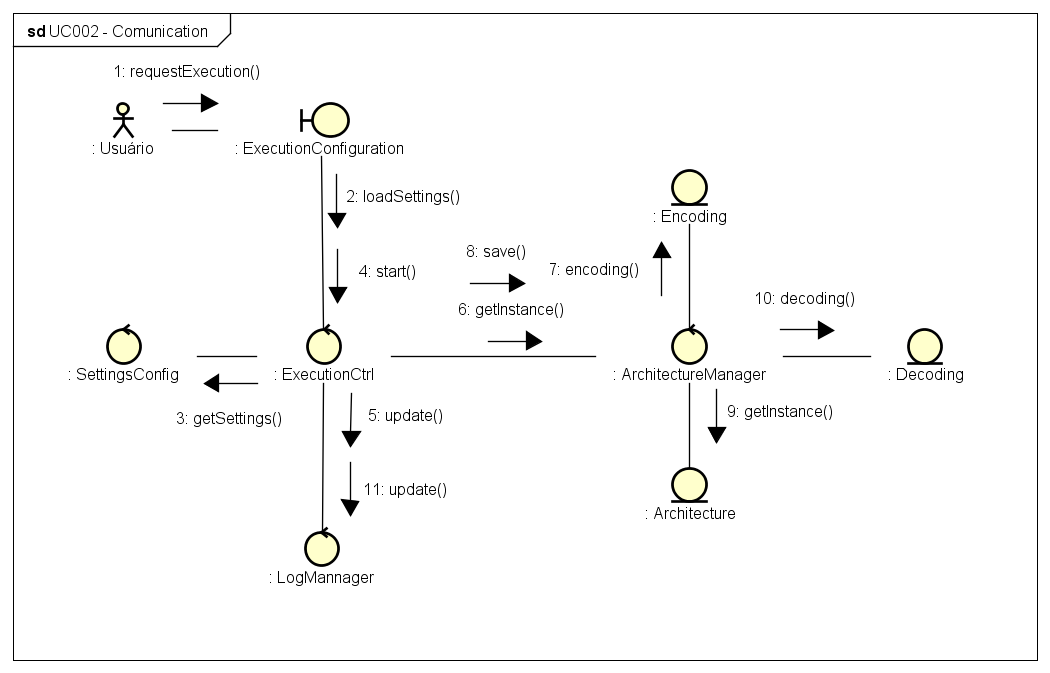


Figura 17 - Comunicação do UC002

* + 1. **UC003 – Acompanhar Execução**

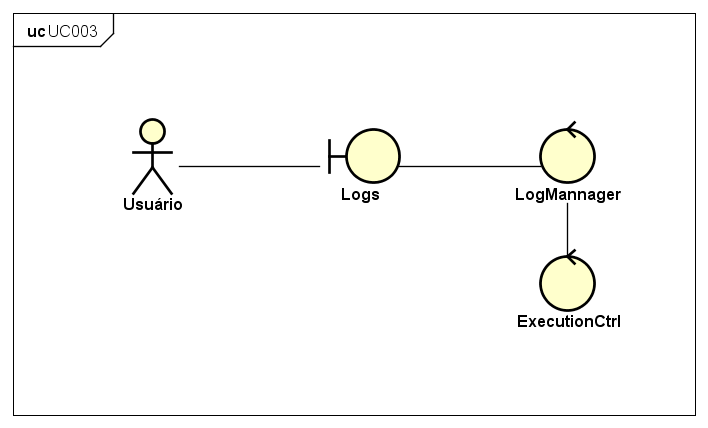


Figura 18 - Realização do UC003

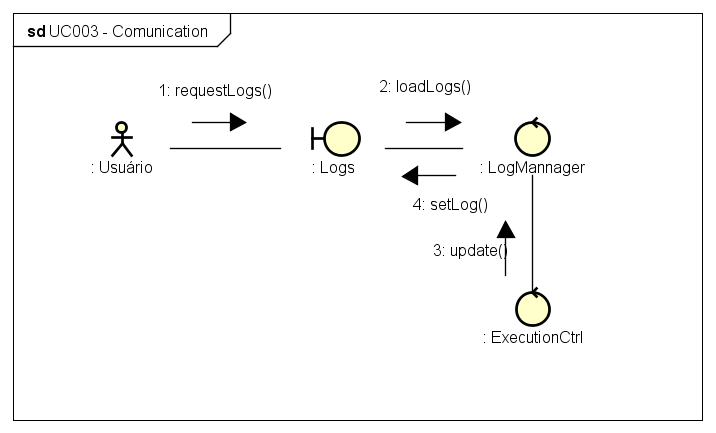


Figura 19 - Comunicação do UC003

* + 1. **UC004 – Gerir Experimentos**

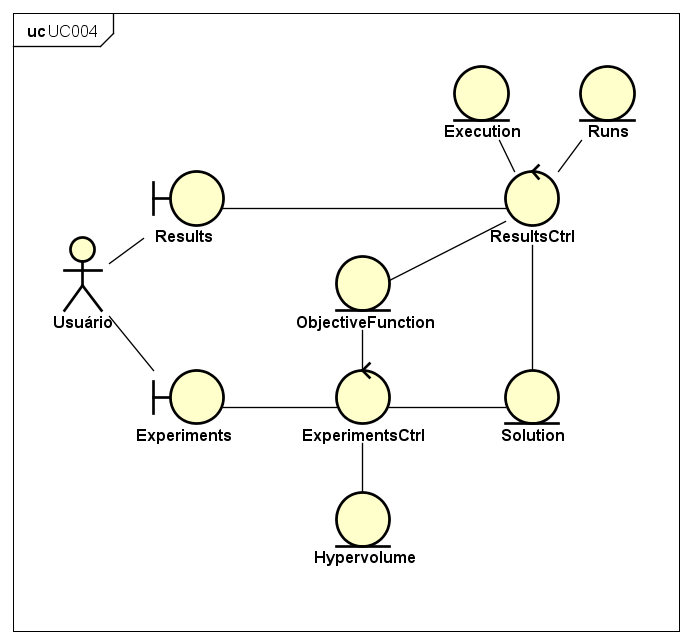


Figura 20 - Realização do UC004

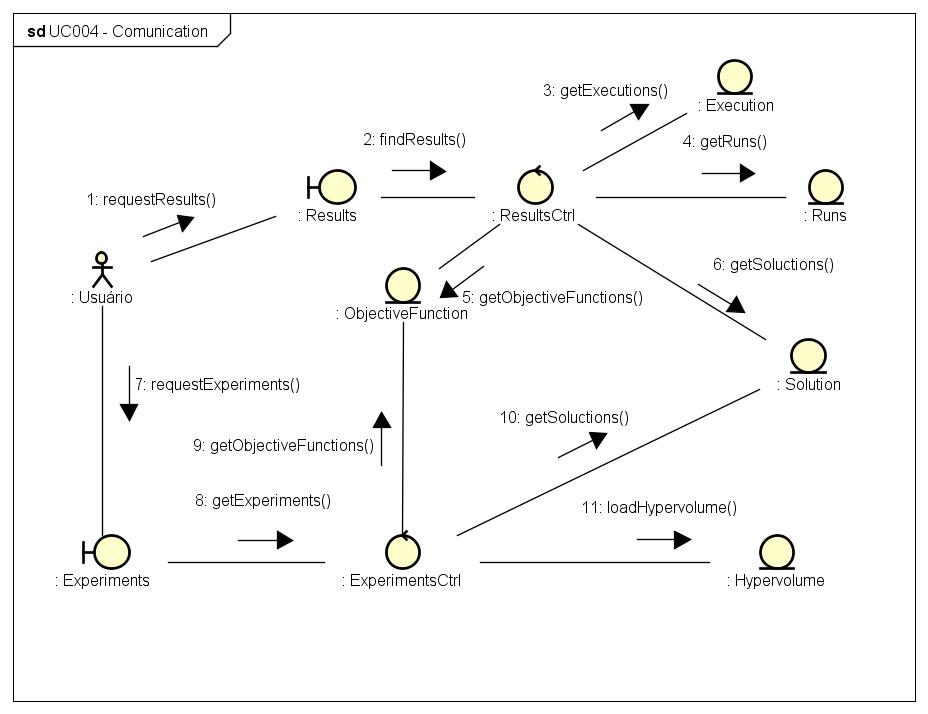


Figura 21 - Comunicação do UC004

* 1. **Análise Arquitetural do sistema**

Esta subseção apresenta a arquitetura do sistema, organizada em pacotes de análise (estereótipo <<*Analysis*>>) e de serviço (estereótipo <<*Service*>>), com suas respectivas dependências.

As classes de análises e dos casos de uso de cada pacote estão referenciadas dentro dele.

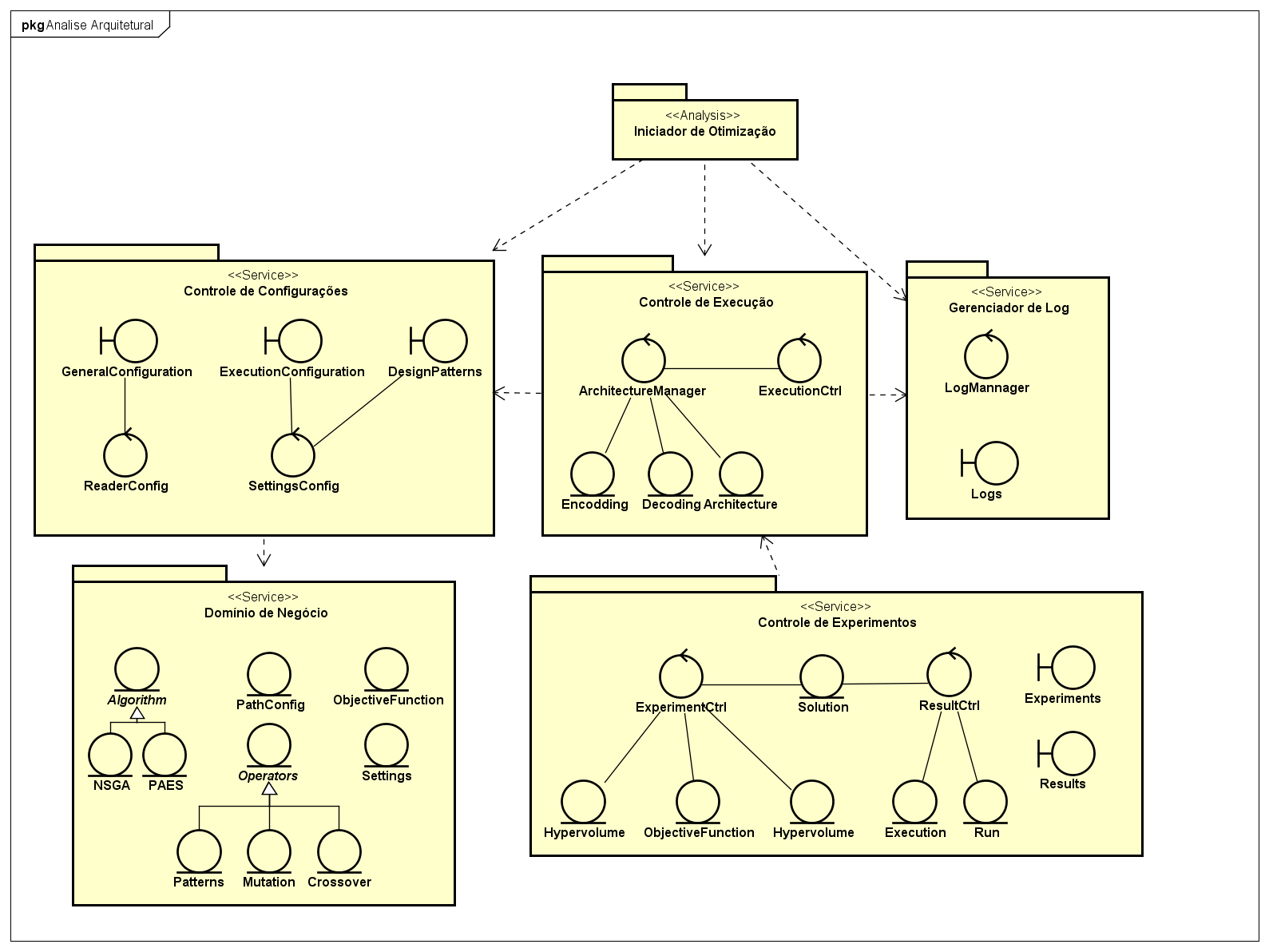


Figura 22 - Análise Arquitetural

* 1. **Visão Global**

Esta subseção apresenta o diagrama com a visão global do sistema, contendo todas as classes de análise e suas relações.

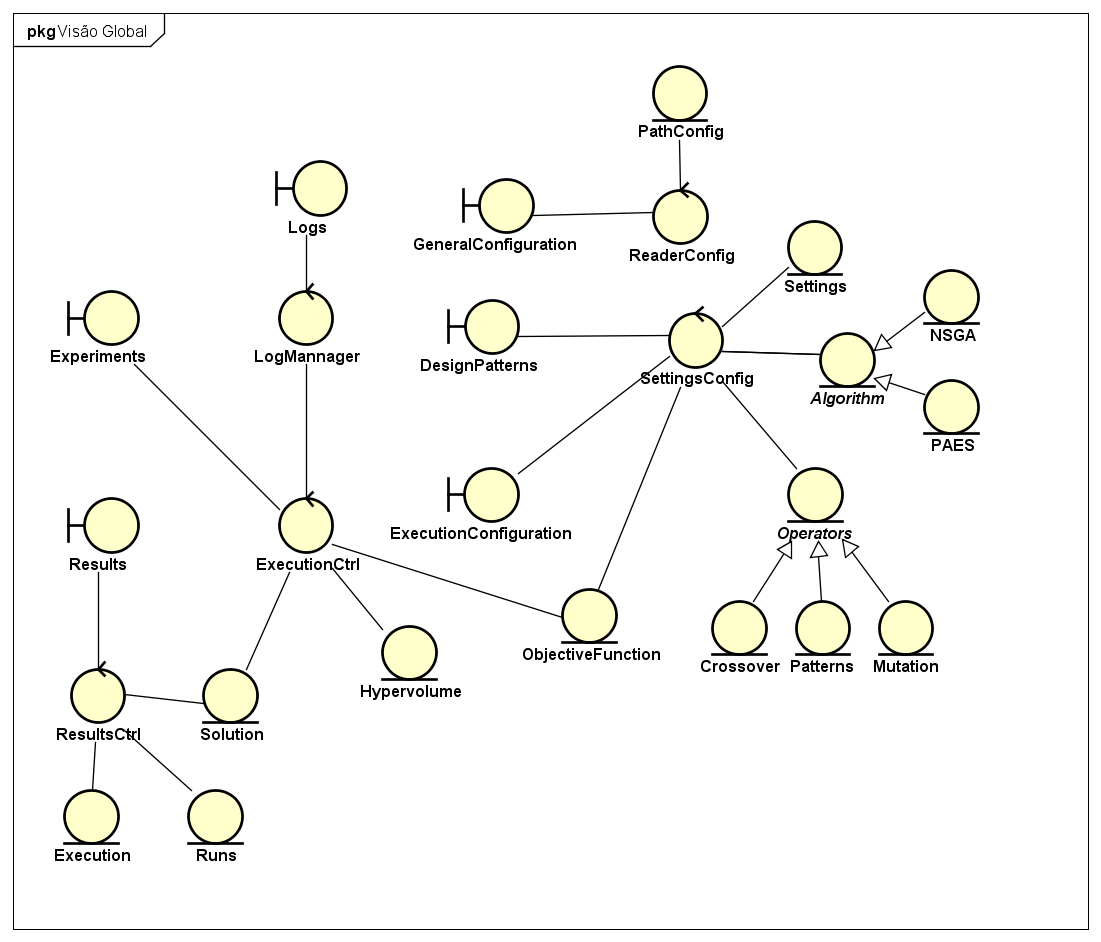


Figura 23 - Visão Global

1. **Workflow de Projeto**

Esta seção apresenta as atividades realizadas no workflow de Projetos do Processo Unificado para a OPLA-Tool. Foram desenvolvidos os seguintes artefatos: Realização dos Casos de Uso considerando os diagramas de classe, de sequência e um diagrama de estados. Também foi realizada a descrição da arquitetura considerando os subsistemas de projeto e de serviço e as interfaces de comunicação entre eles. Por fim, o modelo de instalação do sistema.

* 1. **Diagrama de Classes**
     1. **UC001 – Gerir Configuração**

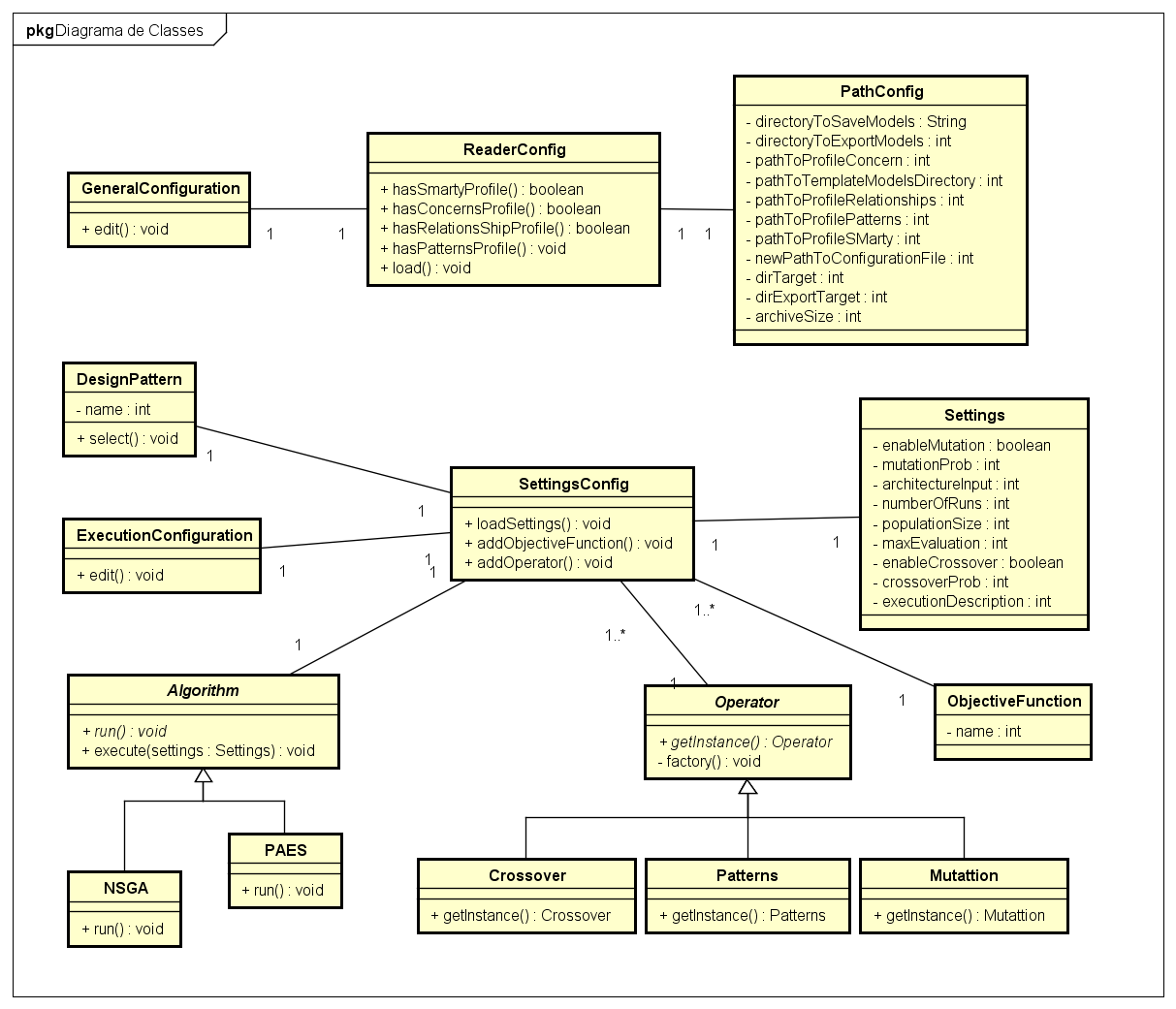
****

Figura 24 - Diagrama de Classes UC001

* + 1. **UC002 – Iniciar Otimização**

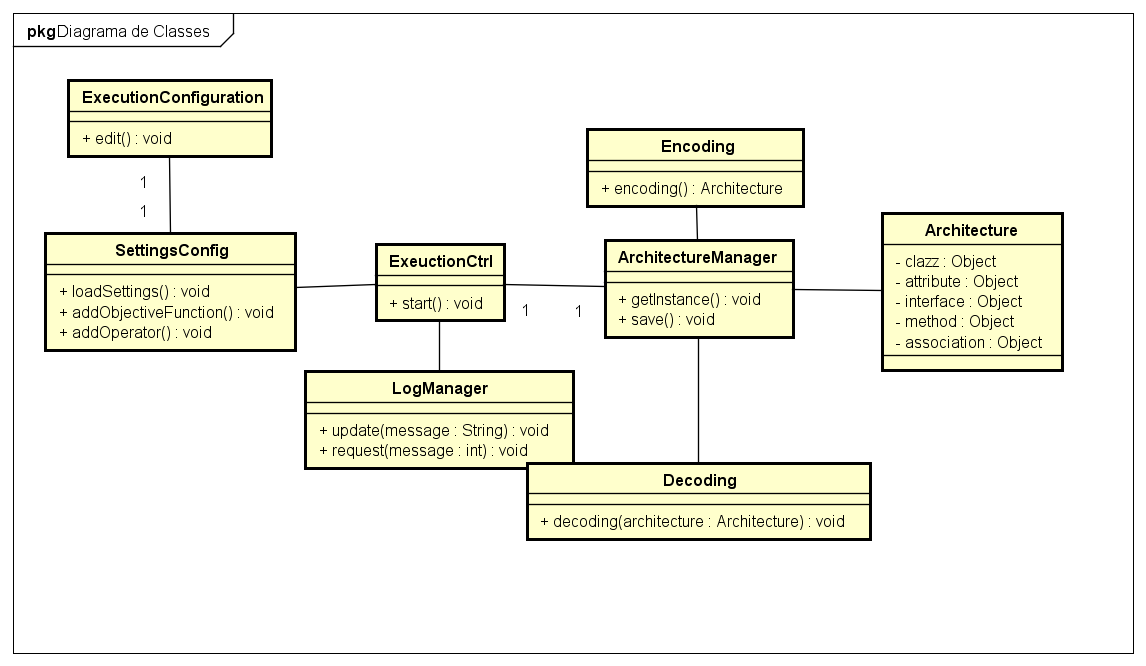


Figura 25 - Diagrama de Classes UC002

* + 1. **UC003 – Acompanhar Execução**

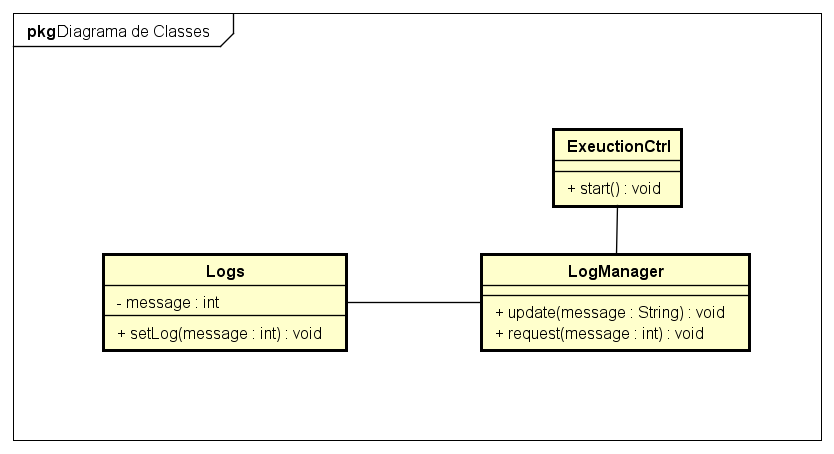
****

Figura 26 - Diagrama de Classes UC003

* + 1. **UC004 – Gerir Experimentos**

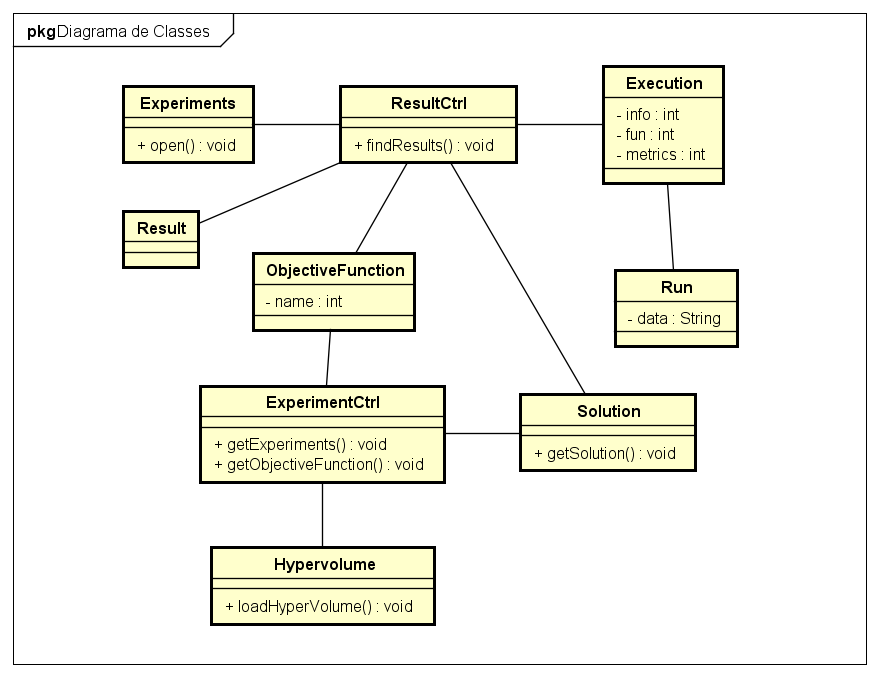
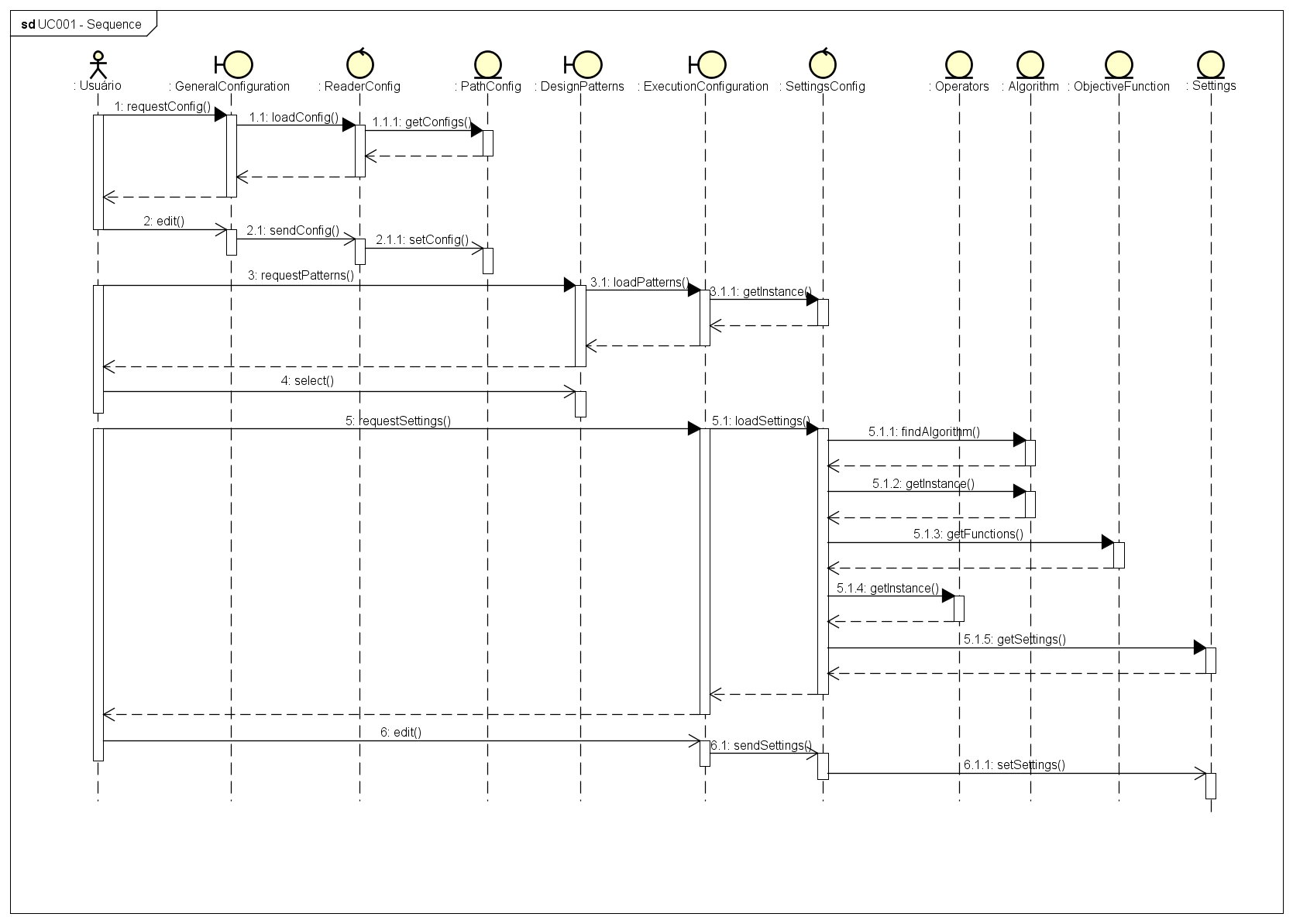
****

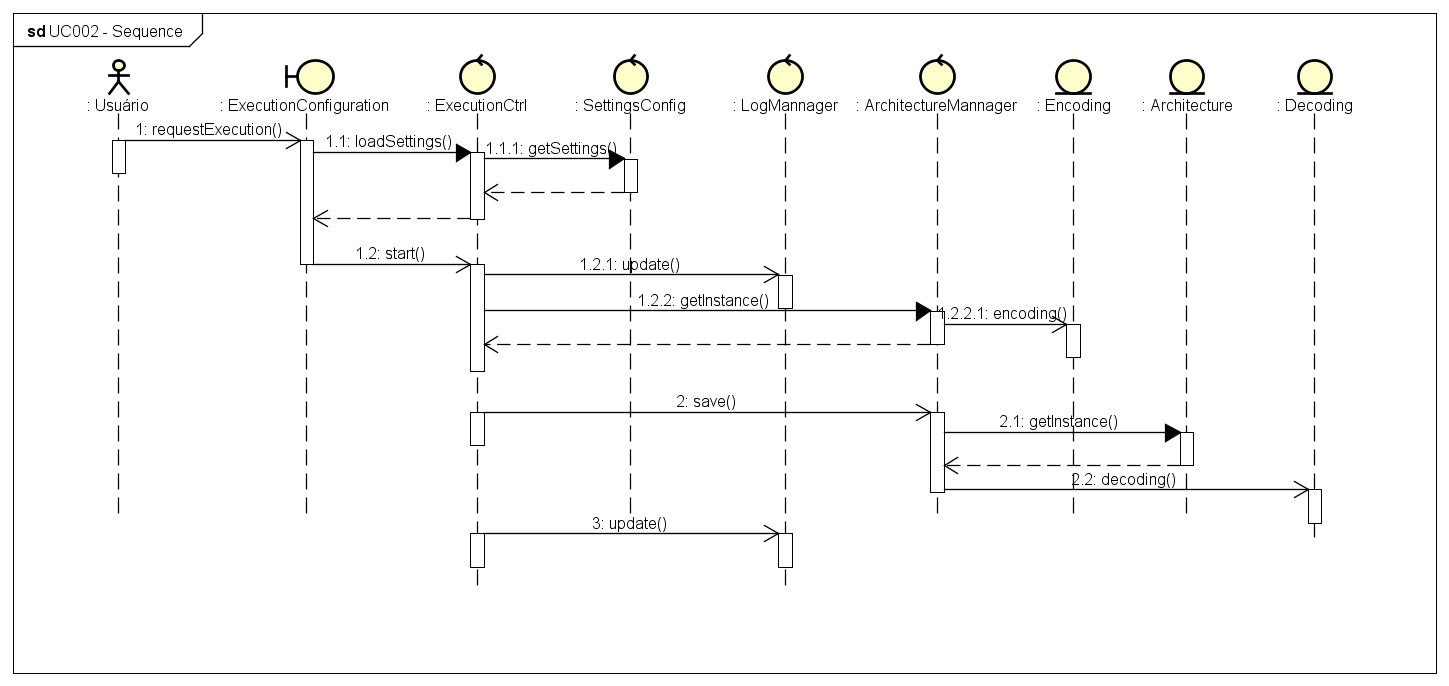
Figura 27 - Diagrama de Classes UC004

* 1. **Diagrama de Sequência**
     1. **UC001 – Gerir Configuração**

****

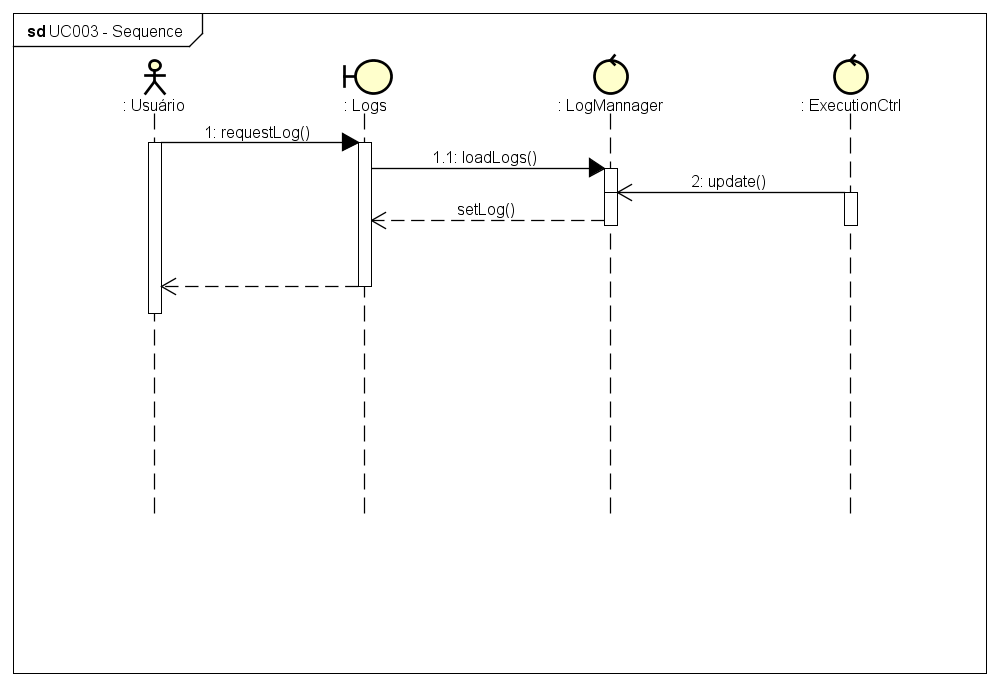
**Figura 28: diagrama de sequência UC001**

* + 1. **UC002 – Iniciar Otimização**

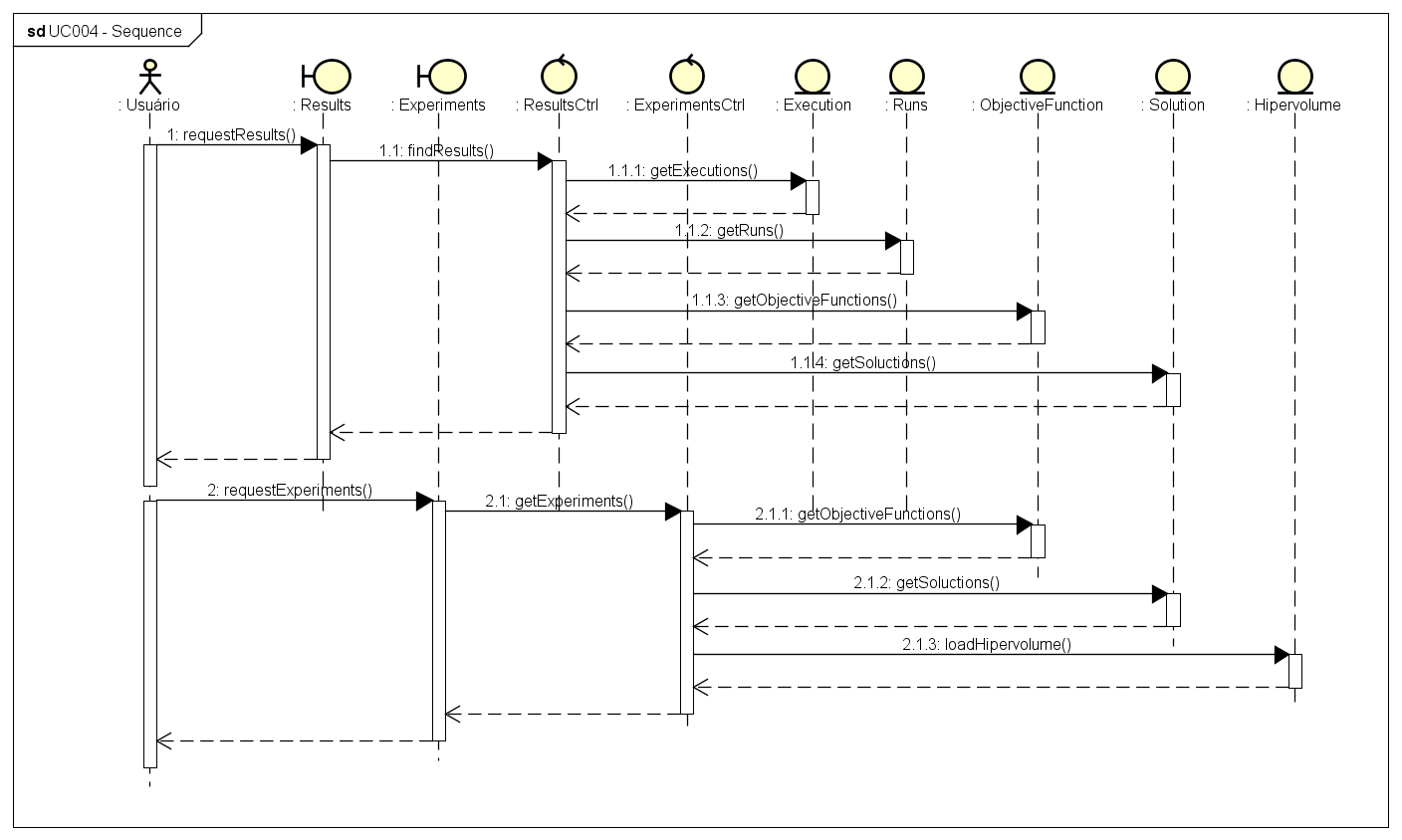


**Figura 29: Diagrama de sequência UC002**

* + 1. **UC003 – Acompanhar Execução**

****

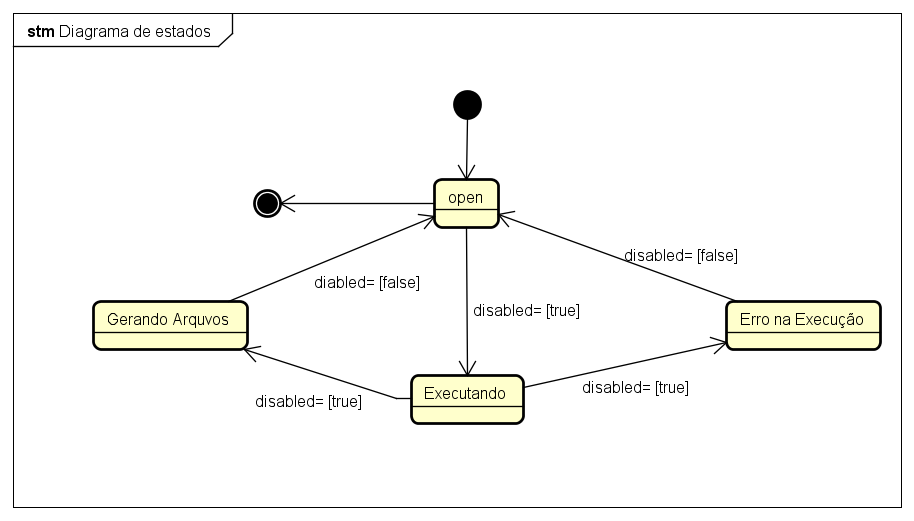
**Figura 30: Diagrama de sequência UC003**

****

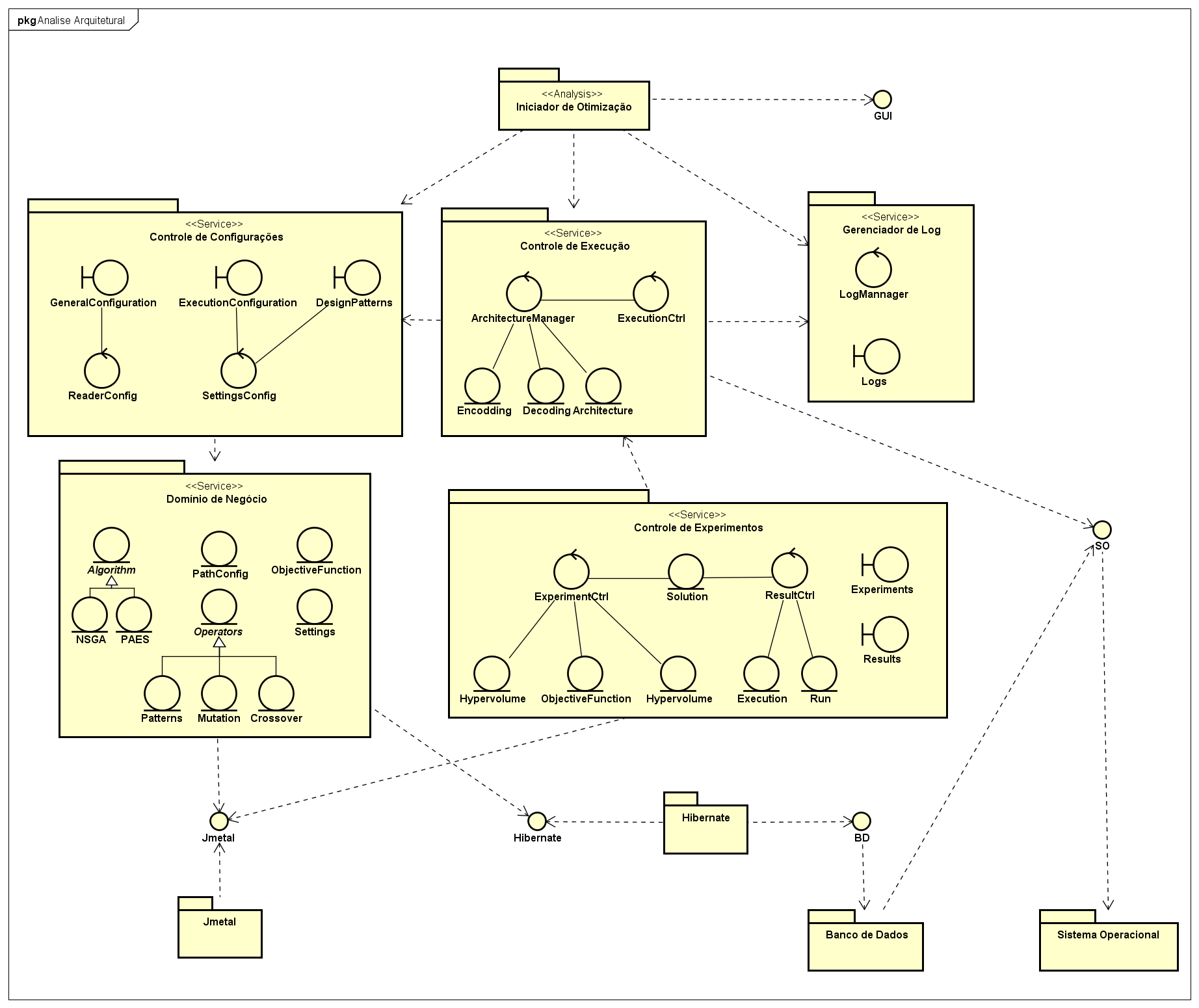
* + 1. **UC004 – Gerir Experimentos**

**Figura 31: Diagrama de sequência UC004**

* 1. **Diagrama de Estado**

**Figura 32: Diagrama de estados**

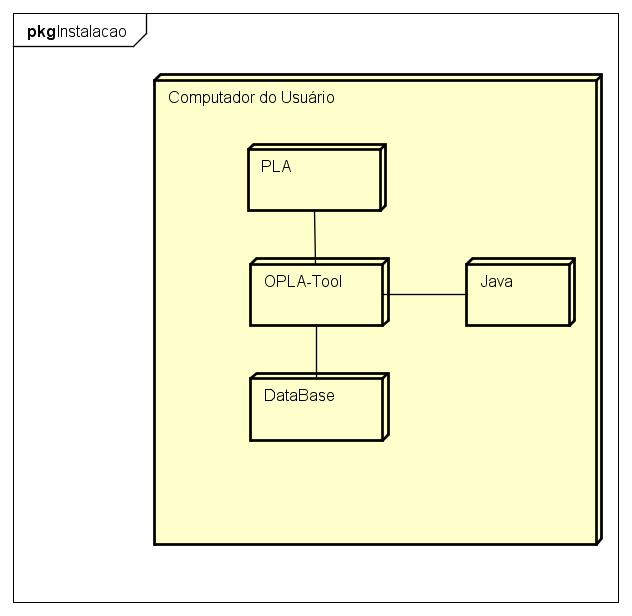
* 1. **Projeto Arquitetural**

****

**Figura 33: Projeto arquitetural**

* 1. **Modelo de Instalação**

O sistema OPLA-Tool funciona de modo off-line, o que o torna uma aplicação de uso local, não necessitando de acesso a rede ou de servidores. A OPLA-Tool possui relação direta com os arquivos XMI(PLA e profiles) e um banco de dados. Abaixo segue o diagrama de instalação:



**Figura 34: Diagrama de instalação**

* 1. **Padrões e Frameworks**

A OPLA-Tool utiliza padrões de projeto: *Strategy*, *Factory*, *Builder*, *Template Method*.

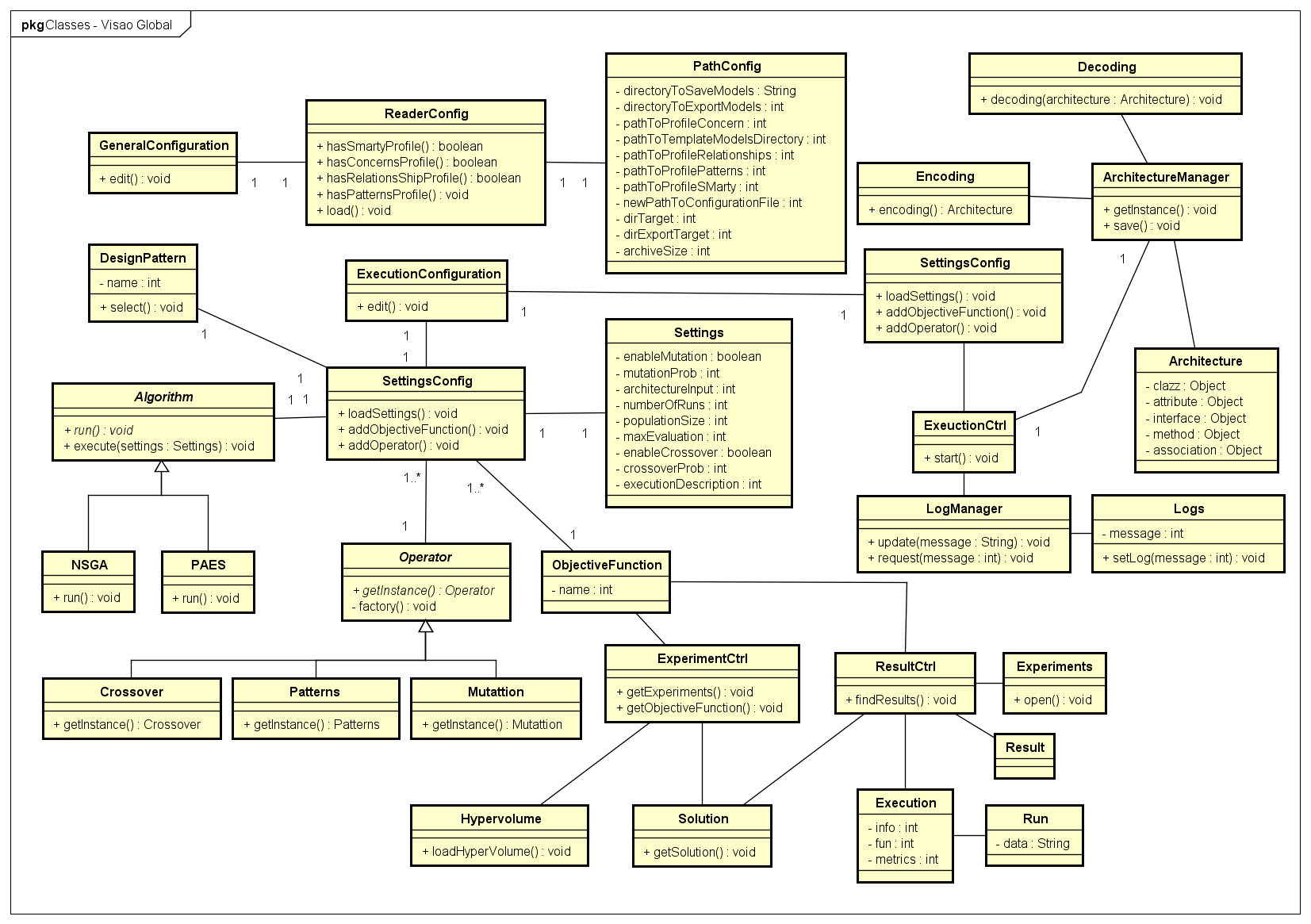
Na OPLA-Tool foram utilizados os frameworks: *Eclipse Modeling Framework* (EFM), *Hibernate* e *JMetal*.

Eclipse EMF é um framework desenvolvido e mantido pela *Eclipse Foundation*. Pode ser usado para representar um modelo de domínio. Dentro do EMF existe uma diferença entre o metamodelo e o modelo real. O meta-modelo é responsável por descrever a estrutura que o modelo real deve seguir, ou seja, o modelo deve estar em conformidade com o meta-modelo. Sendo assim, um modelo é uma instância do meta-modelo. O EMF usa o padrão XMI para persistir os modelos.

A OPLA-Tool estende a implementação padrão do framework *jMetal* e adiciona um novo problema de otimização, novos operadores evolutivos e novas métricas (convencionais e específicas para avaliação de PLAs).

O *Hibernate* foi utilizado para o acesso dos dados (persistência). Ele tem como objetivo a diminuição da complexidade entre os programas Java (modelo orientado a objeto), ao trabalhar com um banco de dados do modelo relacional.

* 1. **Diagrama de Classe – Visão Global**

****

**Figura 35: Diagrama de Classe – Visão Global**

1. **Protótipo**

A seguir é apresentado o protótipo do sistema OPLA-Tool.

* 1. **Configurações Gerais**

A tela apresentada na Figura 1 fornecerá dados relativos às configurações gerais do sistema, bem como, os caminhos referentes à utilização e locais de perfis e locais dos arquivos XMI, assim como, exibição do atual conteúdo do arquivo de configuração.

* Na seção *Profiles Configuration* deve-se selecionar quais arquivos de perfis serão utilizados e seus respectivos caminhos;
* Na seção Template Configuration deve-se especificar o diretório dos arquivos XMI;
* Na seção *Manipulation Directory* deve-se especificar o diretório para manipulação dos arquivos temporários (durante a execução dos experimentos).

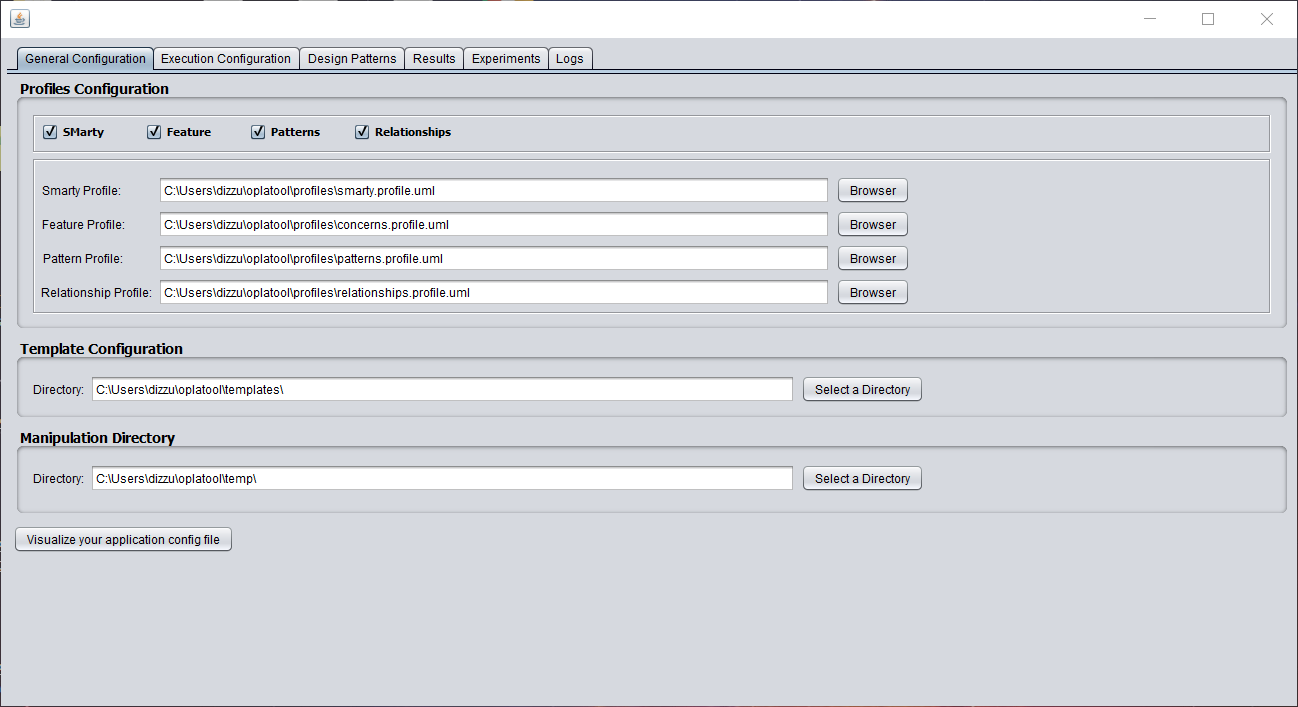


Figura 36 - Tela de Configurações Gerais

* 1. **Configurações de Execução**

A tela apresentada na Figura 2 ilustra a aba *Execution Configuration* que concentrará as configurações relativas à execução de experimentos.

Será possível escolher qual MOEA *(Multiobjective Evolutionary Algorithm)* será usado, assim como, funções objetivo, operadores de mutação, etc..

* Na seção *Settings* o usuário deve selecionar o algoritmo multiobjetivo e seus parâmetros (número de rodadas, número de iterações, o tamanho da população e o tamanho do arquivo);
* Na seção *Objective Functions* o usuário deverá escolher quais funções objetivo serão utilizadas;
* Na seção *Operators*, o usuário seleciona se deseja utilizar o operador de mutação e/ou o operador de cruzamento assim como suas respectivas probabilidades de ocorrência;
* A seção *Mutation Operators* será somente habilitada se o usuário escolher o operador de mutação no checkbox. Esses operadores definem como ocorrerá a mutação de acordo com propriedades da PLA;
* Na seção *Input Architecture* é especificado o caminho do arquivo .uml da PLA alvo da otimização;
* Na seção *Output Directory* é especificado o caminho dos resultados obtidos pela execuçõe.

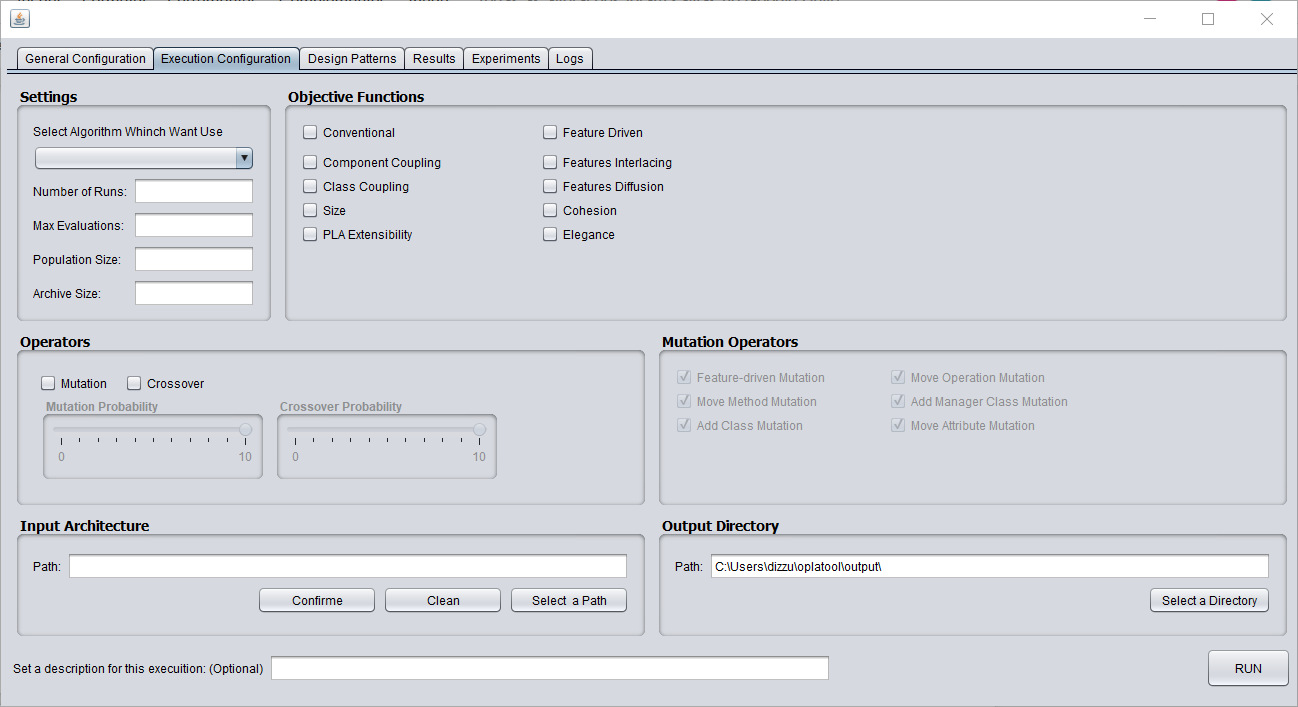


Figura 37 - Tela de Configurações da Execução

* 1. **Padrão de Projeto (*Design Pattern*)**

Na tela apresentada na Figura 3 será possível selecionar quais padrões de projetos deseja-se utilizar. Três opções estão disponíveis, sendo elas: Mediator, Strategy e Bridge. Uma vez selecionado um ou mais padrões de projetos é exibida a opção para ao usuário selecionar opções de customização (utilização randômica ou somente em elementos com mesmos padrões de projeto ou nenhum) de padrões de projeto;

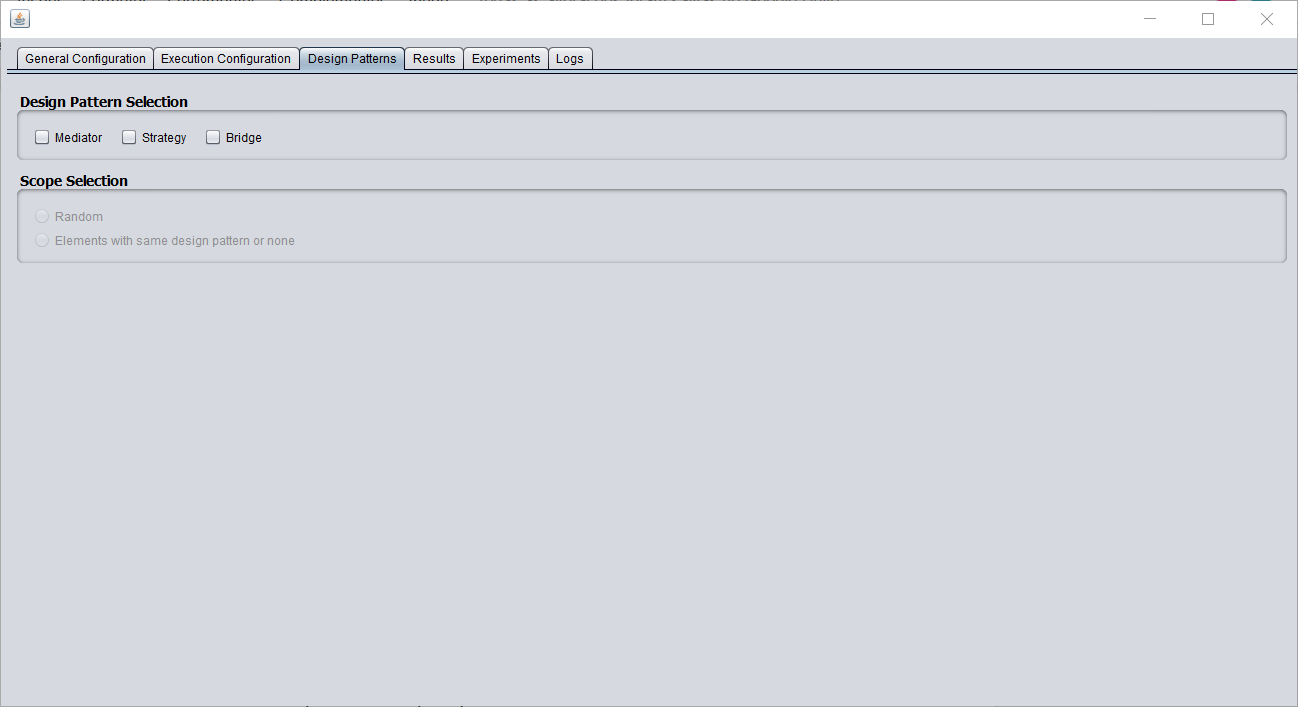


Figura 38 - Tela de Padrões de Projeto

* 1. **Resultados**

Na tela da Figura 4 (tela de resultados) será possível ter acesso a resultados da execução. Serão apresentadas execuções realizadas pela ferramenta OPLA-Tool até o momento.

* Na seção *Executions* o usuário deve selecionar uma execução;
* Na seção *Runs* deve ser carregado automaticamente as rodadas da execução selecionada e o usuário deve escolher uma das rodadas;
* No *combobox* *Solution* deve carregar em um *combobox* soluções referentes à execução selecionada e o usuário deve escolher uma delas (filtro);
* No *combobox Objective Solution*, o sistema deve carregar funções objetivo utilizadas e o usuário deve selecionar uma delas. Abaixo do *combobox* serão carregados os valores das funções objetivo relativos à solução selecionada no *combobox Objective Solution*;
* Na seção entre o *combobox Objective Solution* e o botão *Non-Dominated Solutions* o sistema carrega valores de métricas que compõem a função objetivo selecionada no *combobox*;
* Ao clicar em *Non-Dominated Solutions* o sistema deve carregar uma nova janela contendo informações somente de soluções não dominadas.

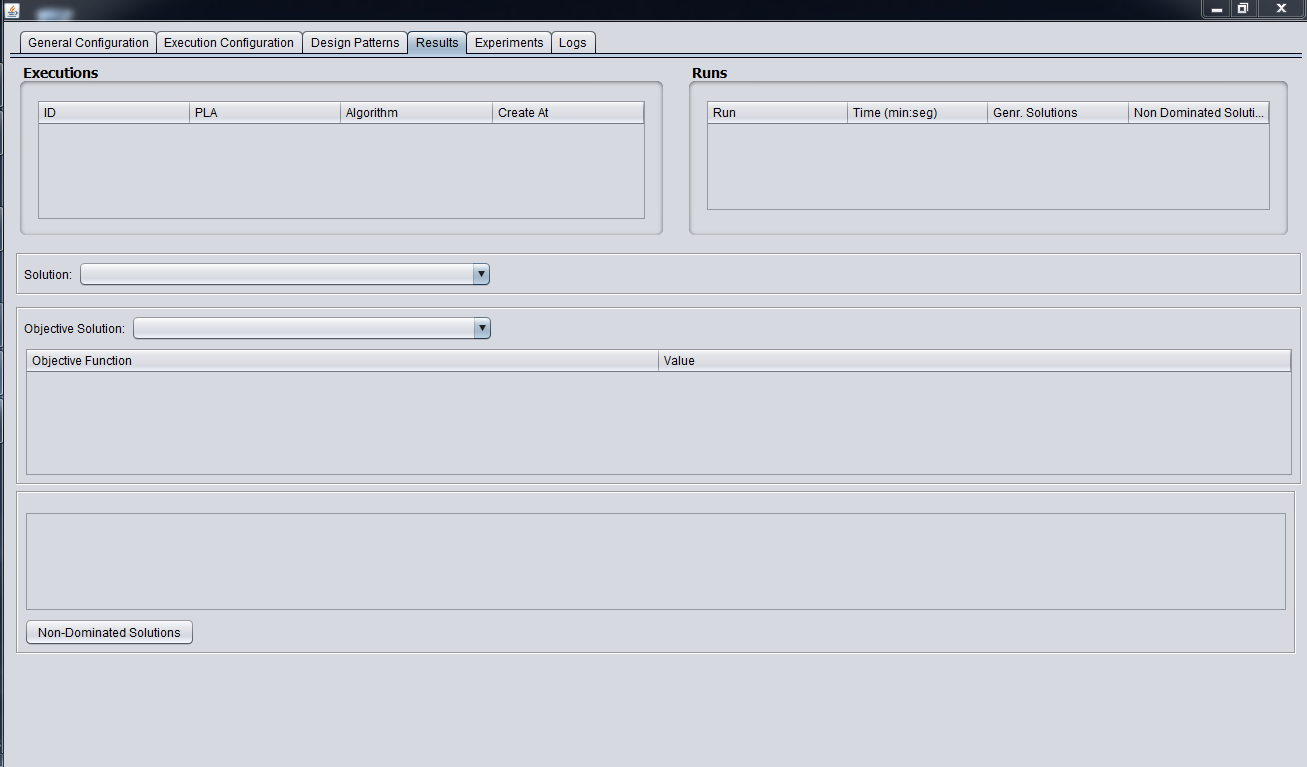


Figura 39 - Tela de Resultados

* 1. **Experimentos**

A tela apresentada na Figura 5 (tela de experimentos) terá como objetivo fornecer uma forma rápida e fácil de realizar comparações entre as execuções.

* Na parte superior da tela o sistema mostra o histórico das execuções em uma tabela e o usuário deve selecionar as execuções a serem comparadas;
* Na seção *Soluctions in the Search Space* o usuário deve marcar as funções objetivo recuperadas e em seguida clicar no botão *Generate Chart*. Com isso o sistema monta o gráfico comparando as soluções no espaço de busca;
* Na seção *Euclidian Distance* o sistema monta o gráfico de números de soluções por distância euclidiana;
* Na seção *Hypervolume* é disponibilizada a opção de calcular o *hypervolume* com dados normalizados ou não.

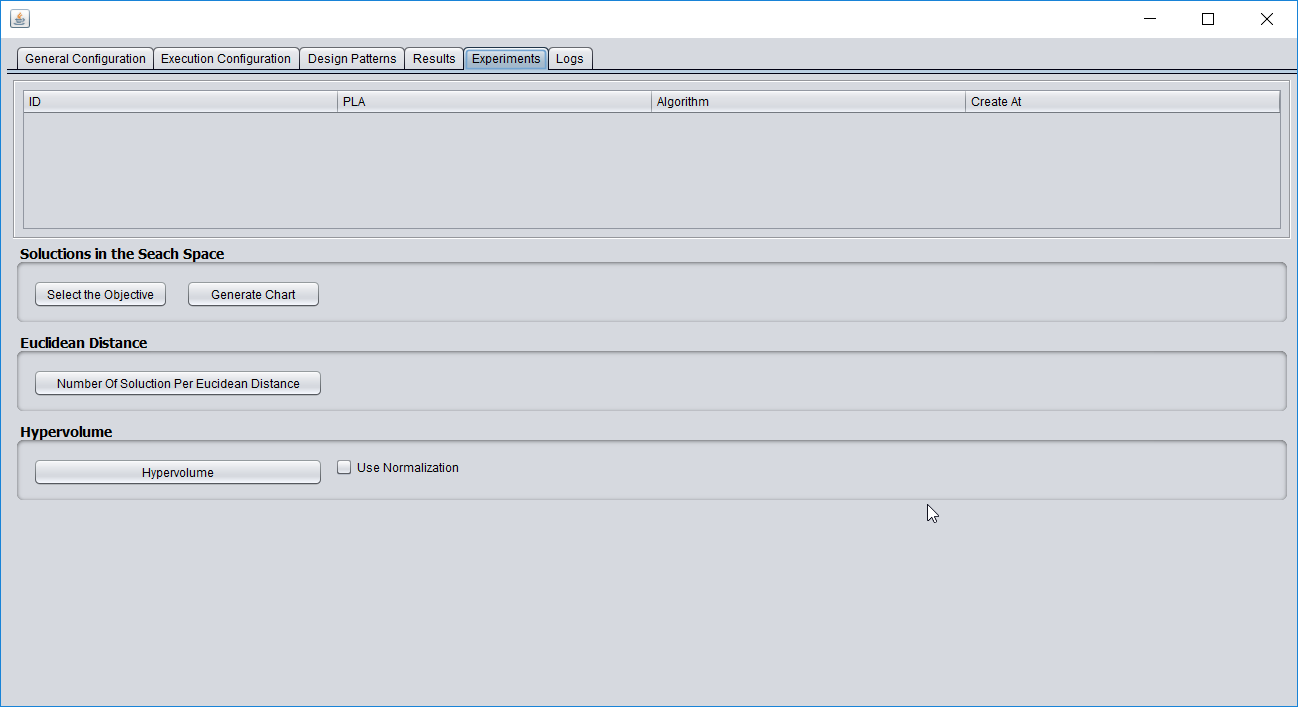


Figura 40 - Tela de Experimentos

* 1. **Logs**

O sistema conterá uma aba chamada Logs, que mostrará informações pertinentes aos procedimentos realizados pela ferramenta, criando um histórico de datas e horários de execuções.

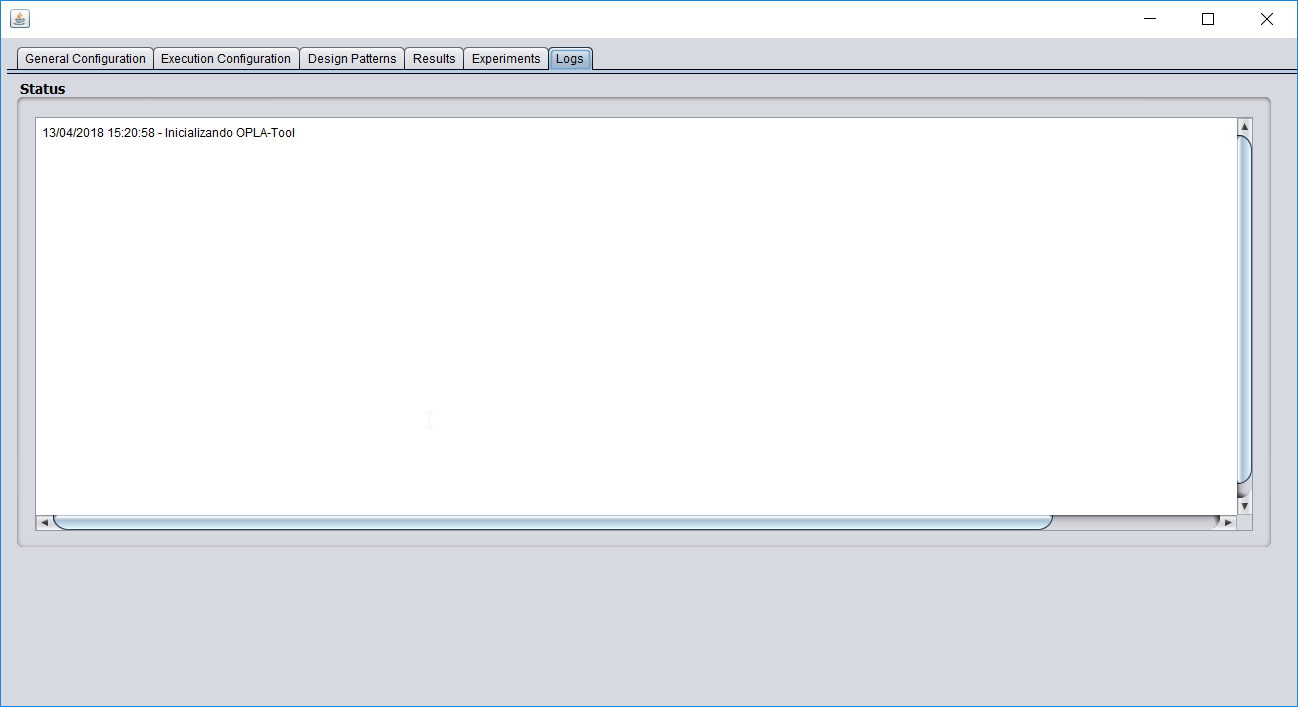


Figura 41 - Tela de Logs do Sistema OPLA-Tool