

ESPECIFICAÇÃO FORMAL

INTRODUÇÃO

Este documento apresenta a especificação formal dos requisitos do jogo Eco Runner.

PLANEJAMENTO

A especificação formal dos requisitos será realizada utilizando notação Z e redes de petri. Para cada requisito funcional e não funcional, haverá uma especificação.

REQUISITOS FUNCIONAIS

RFO1

- Notação Z
- Tipos Básicos:

[NICKNAME, SENHA]

RespostaLogin ::= valido | erro_nick_invalido | erro_senha_invalida

ContaUsuarios

Usuarios : NICKNAME \rightarrow SENHA

\forall nick : dom Usuarios •

$(3 \leq \#nick \leq 12 \wedge 8 \leq \#(Usuarios\ nick) \leq 64)$

- Esquema de estado:
- Esquemas de operação:

LoginSucesso

\exists ContaUsuarios

nick? : NICKNAME

senha? : SENHA

resp! : RespostaLogin

nick? \in dom Usuarios

Usuarios(nick?) = senha?

resp! = valido

NickInvalido

\exists ContaUsuarios

nick? : NICKNAME

senha? : SENHA

resp! : RespostaLogin

nick? \notin dom Usuarios

resp! = erro_nick_invalido

SenhaInvalida

\exists ContaUsuarios

nick? : NICKNAME

senha? : SENHA

resp! : RespostaCadastro

nick? \notin dom Usuarios

Usuarios nick \neq senha?

resp!= erro_senha_invalida

- Operação completa:

$\text{Login} \triangleq \text{LoginSucesso} \vee \text{NickInvalido} \vee \text{SenhaInvalida}$

RFO2

- Notação Z
- Tipos Básicos:

$[\text{NICKNAME}, \text{SENHA}]$

$\text{RespostaCadastro} ::= \text{valido} \mid \text{erro_nick_existe} \mid \text{erro_formato}$

- Esquema de estado:

ContaUsuarios

Usuarios : NICKNAME \rightarrow SENHA

$\forall \text{nick} : \text{dom Usuarios} \bullet$

$(3 \leq \#\text{nick} \leq 12 \wedge 8 \leq \#(\text{Usuarios nick}) \leq 64)$

- Esquemas de operação:

CadastrarSucesso

Δ ContaUsuarios

nick? : NICKNAME

senha? : SENHA

resp! : RespostaCadastro

$3 \leq \#nick? \leq 12$

$8 \leq \#senha? \leq 64$

nick? \notin dom Usuarios

Usuarios' = Usuarios \cup {nick? \mapsto senha?}

resp! = valido

FormatoInvalido

\exists ContaUsuarios

nick? : NICKNAME

senha? : SENHA

resp! : RespostaCadastro

$(3 > \#nick? \vee \#nick? > 12) \vee$

$(8 > \#senha? \vee \#senha? > 64)$

resp! = erro_formato

NickJaExiste

\exists ContaUsuarios

nick? : NICKNAME

senha? : SENHA

resp! : RespostaCadastro

$3 \leq \#nick? \leq 12$

$8 \leq \#senha? \leq 64$

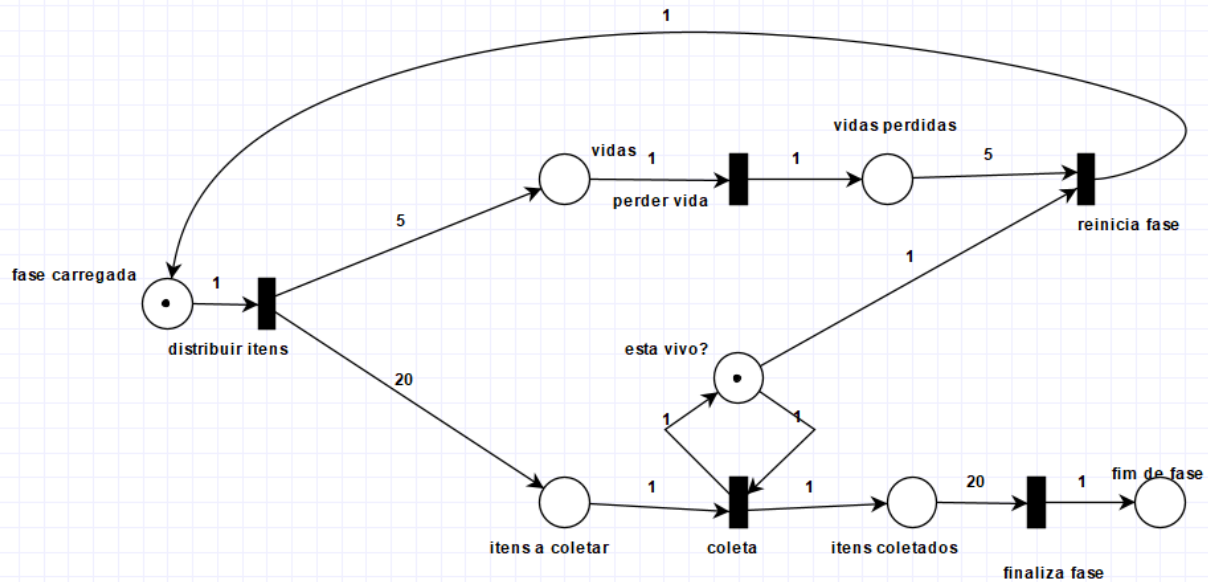
nick? \in dom Usuarios |

resp! = erro_nick_existe

- Operação completa:

$\text{Cadastrar} \triangleq \text{CadastrarSucesso} \vee \text{FormatoInvalido} \vee \text{NickJaExiste}$

RF03



RF04

- Notação Z
- Tipos básicos

$\text{TIPO_LIXO} ::= \text{papel} \mid \text{plastico} \mid \text{vidro} \mid \text{metal}$

$\text{RespostaSort} ::= \text{valido} \mid \text{erro_vidas_zero} \mid \text{erro_inventario_vazio}$

- Esquema de estado:

EstadoFase2

$\text{vidas} : \mathbb{N}$

$\text{inventario} : \text{TIPO_LIXO} \rightarrow \mathbb{N}$

$\text{lixeras} : \text{TIPO_LIXO} \rightarrow \mathbb{N}$

$\text{game_over} : \mathbb{B}$

$\text{fase_completa} : \mathbb{B}$

$\text{vidas} \leq 5$

- Esquemas de operação:

SortItemCorreto

Δ EstadoFase2

item? : TIPO_LIXO

lixreira? : TIPO_LIXO

resp! : RespostaSort

$\neg \text{game_over} \wedge \neg \text{fase_completa}$

item? = lixeira?

inventario(item?) > 0

lixearas(lixreira?) < 5

inventario' = inventario \oplus {item? \mapsto inventario(item?) - 1}

lixearas' = lixearas \oplus {lixreira? \mapsto lixearas(lixreira?) + 1}

vidas' = vidas

Resp! = valido

SortItemErrado

Δ EstadoFase2

item? : TIPO_LIXO

lixreira? : TIPO_LIXO

resp! : RespostaSort

$\neg \text{game_over} \wedge \neg \text{fase_completa}$

item? \neq lixeira?

inventario(item?) > 0

vidas' = vidas - 1

inventario' = inventario

lixearas' = lixearas

resp! = valido

InventarioVazio

Δ EstadoFase2

item? : TIPO_LIXO

resp! : RespostaSort

inventario(item?) = 0

resp! = erro_inventario_vazio

JogoPerdido

Ξ EstadoFase2

resp! : RespostaSort

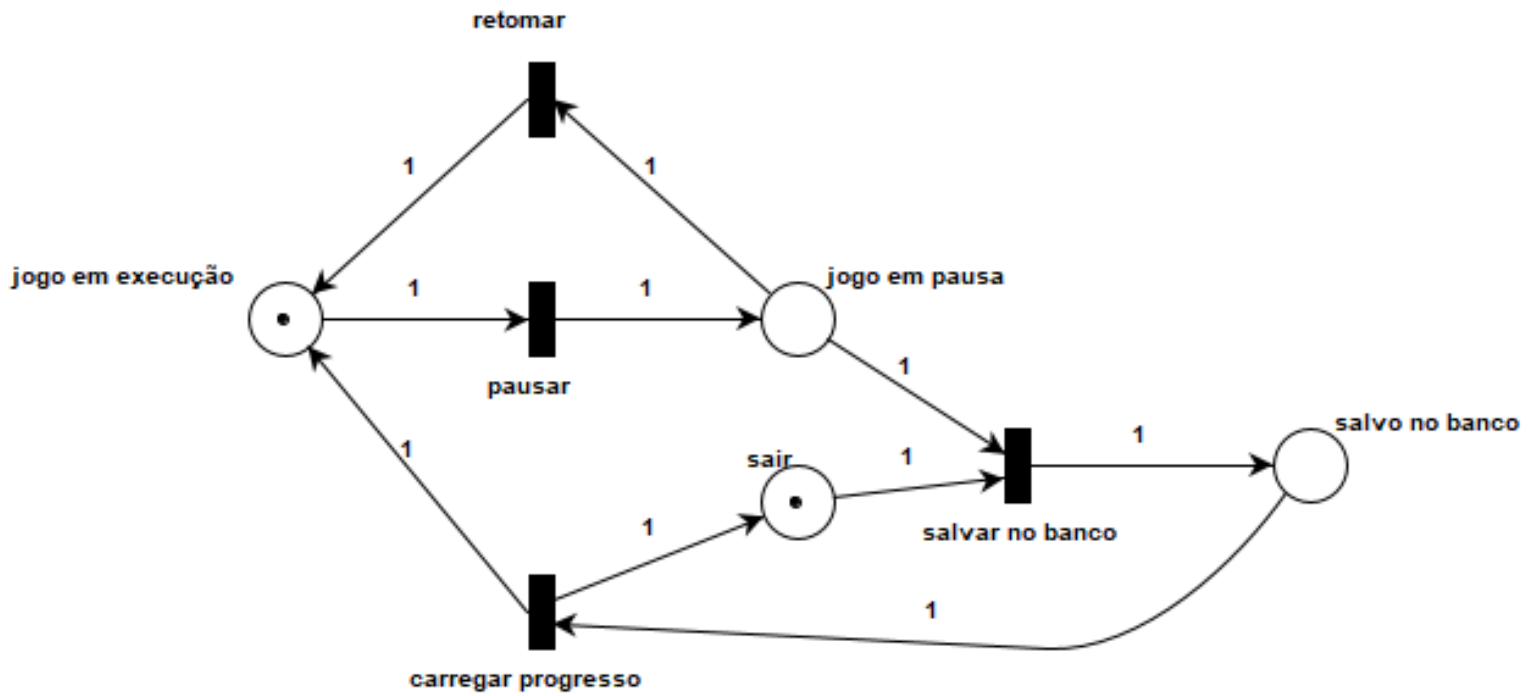
game_over

resp! = erro_vidas_zero

- Operação completa:

$\text{SortItem} \triangleq \text{SortItemCorreto} \vee \text{SortItemErrado} \vee \text{InventarioVazio} \vee \text{JogoPerdido}$

RF06



REQUISITOS NÃO FUNCIONAIS

RNF02

- Notação Z
- Tipos básicos
[TELA, COR, POSICAO, BOTAO]
- Esquema de estado:

Interface

telas : \mathbb{P} TELA
botoes: \mathbb{P} BOTAO
cor_botao_em_tela : (TELA \times BOTAO) \leftrightarrow COR
pos_botao_em_tela : (TELA \times BOTAO) \leftrightarrow POSICAO

$\forall t1, t2 : \text{telas} ; b : \text{botoes} \bullet$
((t1, b) \in dom cor_botao_em_tela \wedge (t2, b) \in dom
cor_botao_em_tela) \Rightarrow cor_botao_em_tela(t1, b) = cor_botao_em_tela(t2, b)
 $\forall t1, t2 : \text{telas} ; b : \text{botoes} \bullet$
((t1, b) \in dom pos_botao_em_tela \wedge (t2, b) \in dom
pos_botao_em_tela) \Rightarrow pos_botao_em_tela(t1, b) = pos_botao_em_tela(t2,
b)

RNF03

- Notação Z
- Tipos básicos:
 - SO_VALIDO ::= windows10 | windows11 | ubuntu22_04
- Esquema de estado

AmbienteDeExecucao

so_atual: SO_VALIDO

RNF04

- Notação Z

- Tipos básicos

[PROGRESSO]

RespostaSave : := sucesso | falha

- Esquema de estado

SistemaDePersistencia

progresso_atual: PROGRESSO

progresso_salvo: PROGRESSO

- Esquemas de operação

SalvarSucesso

Δ SistemaDePersistencia

resp! : RespostaSave

progresso_salvo' = progresso_atual

progresso_atual' = progresso_atual

resp! = ok

SalvarFalha

Δ SistemaDePersistencia

resp! : RespostaSave

progresso_salvo' = progresso_salvo

progresso_atual' = progresso_atual

resp! = falha

CarregarJogo	
Δ SistemaDePersistencia	
progresso_atual' = progresso_salvo	
progresso_salvo' = progresso_salvo	

- Operação completa

$\text{SalvarJogo} \triangleq \text{SalvarSucesso} \vee \text{SalvarFalha}$