



DEPARTAMENTO
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

TP de Especificación y Diseño

Modelado de SimCity

1 de Junio de 2022

Algoritmos y Estructuras de Datos II

Grupo 01 - hasTADlaVista, turno mañana

Integrante	LU	Correo electrónico
Lakowsky, Manuel	511/21	mlakowsky@gmail.com
Vekselman, Natán	338/21	natanvek11@gmail.com
Arienti, Federico	316/21	fa.arianti@gmail.com



Facultad de Ciencias Exactas y Naturales

Universidad de Buenos Aires

Ciudad Universitaria - (Pabellón I/Planta Baja)

Intendente Güiraldes 2610 - C1428EGA

Ciudad Autónoma de Buenos Aires - Rep. Argentina

Tel/Fax: (+54 +11) 4576-3300

<http://www.exactas.uba.ar>

1. Especificación

1.1. Mapa

TAD MAPA

igualdad observacional

$$(\forall m, m' : \text{Mapa}) \left(m =_{\text{obs}} m' \iff \left(\text{horizontales}(m) =_{\text{obs}} \text{horizontales}(m') \wedge_L \text{verticales}(m) =_{\text{obs}} \text{verticales}(m') \right) \right)$$

géneros Mapa

exporta completar

usa completar

observadores básicos

horizontales : Mapa \longrightarrow conj(Nat)

verticales : Mapa \longrightarrow conj(Nat)

Mapa

generadores

crear : conj(Nat) \times conj(Nat) \longrightarrow Mapa

axiomas $\forall hs, vs: \text{conj}(\text{Nat})$

horizontales(crear(hs, vs)) \equiv hs

verticales(crear(hs, vs)) \equiv vs

Fin TAD


```

comercios(iniciar(m))           ≡ vacío
comercios(avanzarTurno(s, cs))  ≡ agComercios(s, comercios(s), cs)
comercios(unir(s, s'))          ≡ agComercios(s,
                                comercios(s),
                                sacarRepes(construcc(s), construcc(s')))
agComercios(s, cn, cs) ≡ if vacío?(claves(cs)) then
    cn
else
    if obtener(dameUno(claves(cs)), cs) =obs 2 then
        agComercios(definir(dameUno(claves(cs)),
                                nivelCom(dameUno(claves(cs)), casas(s), cn),
                                borrar(dameUno(claves(cs)), cs))
    else
        agComercios(cn, borrar(dameUno(claves(cs)), cs))
    fi
fi
nivelCom(p, cn) ≡ if vacío?(claves(cn)) then
    1
else
    if distManhatt(p, dameUno(claves(cn))) ≤ 3 then
        max(obtener(dameUno(claves(cn)), cn),
            nivelCom(p, borrar(dameUno(claves(cn)), cn)))
    else
        nivelCom(p, borrar(dameUno(claves(cn)), cn))
    fi
fi
distManhatt(p, q) ≡ if π0(p) < π0(q) then q - p else p - q fi
+
if π1(p) < π1(q) then q - p else p - q fi
popularidad(iniciar(m)) ≡ 0
popularidad(avanzarTurno(s, cs)) ≡ popularidad(s)
popularidad(unir(s, s')) ≡ popularidad(s) + 1
turnos(iniciar(m)) ≡ 0
turnos(avanzarTurno(s, cs)) ≡ turnos(s) + 1
turnos(unir(s, s')) ≡ if turnos(s) < turnos(s') then turnos(s') else turnos(s) fi
construcc(s) ≡ casas(s) ∪dicc comercios(s)
d ∪dicc d' ≡ if vacío?(claves(d')) then
    d
else
    definir(dameUno(claves(d')),
            obtener(dameUno(claves(d')), d'),
            d ∪dicc borrar(dameUno(claves(d')), d'))
    fi
sacarRepes(cs, cs') ≡ if vacío?(claves(cs)) then
    cs'
else
    if def?(dameUno(claves(cs)), cs') then
        sacarRepes(borrar(dameUno(claves(cs)), cs),
                    borrar(dameUno(claves(cs)), cs'))
    else
        sacarRepes(borrar(dameUno(claves(cs)), cs), cs')
    fi
fi

```

Fin TAD

*donde:

avanzarTurnoValido : SimCity $s \times \text{dicc}(\text{Pos} \times \text{Construccion}) \text{ cs} \longrightarrow \text{boolean}$

$$\begin{aligned} \text{avanzarTurnoValido}(s, cs) \equiv & \neg \text{vacio?}(\text{claves}(cs)) \wedge \\ & (\forall p : \text{Pos})(\text{def?}(p, cs) \Rightarrow_{\text{L}} \\ & \quad (\neg p \in \text{claves}(\text{construcc}(s)) \wedge \\ & \quad \neg \pi_0(p) \in \text{horizontales}(\text{mapa}(s)) \wedge \neg \pi_1(p) \in \text{verticales}(\text{mapa}(s)) \wedge \\ & \quad (\text{obtener}(p, cs) =_{\text{obs}} 1 \vee \text{obtener}(p, cs) =_{\text{obs}} 2)) \\ &) \end{aligned}$$

unirValido : Simcity $a \times \text{SimCity } b \longrightarrow \text{boolean}$

$$\begin{aligned} \text{unirValido}(a, b) \equiv & (\forall p : \text{Pos})(\text{def?}(p, \text{construcc}(a)) \Rightarrow_{\text{L}} \\ & \quad (\neg \pi_0(p) \in \text{horizontales}(\text{mapa}(b)) \wedge \neg \pi_1(p) \in \text{verticales}(\text{mapa}(b)) \wedge \\ & \quad (\neg (\exists \text{otra} : \text{Pos})(\text{def?}(\text{otra}, \text{construcc}(a)) \wedge_{\text{L}} \\ & \quad \quad \text{obtener}(\text{otra}, \text{construcc}(a)) > \text{obtener}(p, \text{construcc}(a)) \Rightarrow_{\text{L}} \\ & \quad \quad \neg \text{def?}(p, \text{construcc}(b))))) \\ &) \wedge \\ & (\forall p : \text{Pos})(\text{def?}(p, \text{construcc}(b)) \Rightarrow_{\text{L}} \\ & \quad (\neg \pi_0(p) \in \text{horizontales}(\text{mapa}(a)) \wedge \neg \pi_1(p) \in \text{verticales}(\text{mapa}(a)) \wedge \\ & \quad (\neg (\exists \text{otra} : \text{Pos})(\text{def?}(\text{otra}, \text{construcc}(b)) \wedge_{\text{L}} \\ & \quad \quad \text{obtener}(\text{otra}, \text{construcc}(b)) > \text{obtener}(p, \text{construcc}(b)) \Rightarrow_{\text{L}} \\ & \quad \quad \neg \text{def?}(p, \text{construcc}(a))))) \end{aligned}$$

1.3. Servidor

TAD SERVIDOR

géneros server

exporta observadores, generadores, verMapa, verCasas, verComercios, verPopularidad y verTurno

usa SimCity, Mapa, Nombre, Pos, Construcccion, Nivel, Nat, bool, $\text{dicc}(\alpha, \beta)$, $\text{conj}(\alpha)$

igualdad observacional

$$(\forall s, s' : \text{server}) \left(s =_{\text{obs}} s' \iff \left(\text{partidas}(s) =_{\text{obs}} \text{partidas}(s') \wedge \text{congeladas}(s) =_{\text{obs}} \text{congeladas}(s') \right) \right)$$

observadores básicos

partidas : server $\rightarrow \text{dicc}(\text{Nombre}, \text{SimCity})$

congeladas : server $\rightarrow \text{conj}(\text{Nombre})$

generadores

nuevoServer : $\rightarrow \text{server}$

nuevaPartida : server $s \times \text{Nombre } p \times \text{Mapa} \rightarrow \text{server} \quad \{\neg \text{def?}(p, \text{partidas}(s))\}$

unirPartidas : server $s \times \text{Nombre } p1 \times \text{Nombre } p2 \rightarrow \text{server} \quad \{*\text{unionValida}(s, p1, p2, cs)\}$

avanzarTurnoPartida : server $s \times \text{Nombre } p \times \text{dicc}(\text{Pos} \times \text{Construcccion}) cs \rightarrow \text{server} \quad \{*\text{avanzarTurnoValido}(s, p, cs)\}$

otras operaciones

verMapa : server $s \times \text{Nombre } p \rightarrow \text{Mapa} \quad \{\text{def?}(p, \text{partidas}(s))\}$

verCasas : server $s \times \text{Nombre } p \rightarrow \text{dicc}(\text{Pos}, \text{Nivel}) \quad \{\text{def?}(p, \text{partidas}(s))\}$

verComercios : server $s \times \text{Nombre } p \rightarrow \text{dicc}(\text{Pos}, \text{Nivel}) \quad \{\text{def?}(p, \text{partidas}(s))\}$

verPopularidad : server $s \times \text{Nombre } p \rightarrow \text{Nat} \quad \{\text{def?}(p, \text{partidas}(s))\}$

verTurno : server $s \times \text{Nombre } p \rightarrow \text{Nat} \quad \{\text{def?}(p, \text{partidas}(s))\}$

axiomas $\forall s: \text{server}, \forall p, p1, p2: \text{Nombre}, \forall m: \text{Mapa}, \forall cs: \text{conj}(\text{Pos})$

partidas(nuevoServer) $\equiv \text{vacio}$

partidas(nuevaPartida(s, p, m)) $\equiv \text{definir}(p, \text{iniciar}(m), \text{partidas}(s))$

partidas(unirPartidas(s, p1, p2)) $\equiv \text{definir}(p1, \text{unir}(\text{obtener}(p1, \text{partidas}(s)), \text{obtener}(p2, \text{partidas}(s))), \text{partidas}(s))$

partidas(avanzarTurnoPartida(s, p, cs)) $\equiv \text{definir}(p, \text{avanzarTurno}(\text{obtener}(p, \text{partidas}(s)), cs), \text{partidas}(s))$

congeladas(nuevaPartida) $\equiv \emptyset$

congeladas(nuevaPartida(s, p, m)) $\equiv \text{congeladas}(s)$

congeladas(avanzarTurnoPartida(s, p, cs)) $\equiv \text{congeladas}(s)$

congeladas(unirPartidas(s, p1, p2)) $\equiv \text{Ag}(p2, \text{congeladas}(s))$

// oo

verMapa(s, p) $\equiv \text{mapa}(\text{obtener}(p, \text{partidas}(s)))$

verCasas(s, p) $\equiv \text{casas}(\text{obtener}(p, \text{partidas}(s)))$

verComercios(s, p) $\equiv \text{comercios}(\text{obtener}(p, \text{partidas}(s)))$

verPopularidad(s, p) $\equiv \text{popularidad}(\text{obtener}(p, \text{partidas}(s)))$

verTurno(s, p) $\equiv \text{turnos}(\text{obtener}(p, \text{partidas}(s)))$

Fin TAD

*donde:

unionValida : server s × Nombre p1 × Nombre p2 → boolean

$$\begin{aligned} \text{unionValida}(s, p1, p2) \equiv & \text{def?}(p1, \text{partidas}(s)) \wedge \text{def?}(p2, \text{partidas}(s)) \wedge \\ & p1 \notin \text{congeladas}(s) \wedge_L \% \\ & (\forall pos : Pos)(pos \in \text{claves}(\text{constr1}) \Rightarrow_L \\ & \quad \neg \text{sobreRio}(pos, \text{sim2}) \wedge \\ & \quad ((\nexists otra : Pos)(otra \in \text{constr1} \wedge_L \\ & \quad \quad \text{obtener}(pos, \text{constr1}).\text{nivel} < \text{obtener}(otra, \text{constr1}).\text{nivel} \\ & \quad) \Rightarrow_L \neg \text{def?}(pos, \text{constr2})) \\ &) \wedge \\ & (\forall pos : Pos)(pos \in \text{claves}(\text{constr2}) \Rightarrow_L \\ & \quad \neg \text{sobreRio}(pos, \text{sim1}) \wedge \\ & \quad ((\nexists otra : Pos)(otra \in \text{constr2} \wedge_L \\ & \quad \quad \text{obtener}(pos, \text{constr2}).\text{nivel} < \text{obtener}(otra, \text{constr2}).\text{nivel} \\ & \quad) \Rightarrow_L \neg \text{def?}(pos, \text{constr1})) \\ &) \\ \text{donde } \text{sim1} \equiv & \text{obtener}(p1, \text{partidas}(s)), \\ \text{sim2} \equiv & \text{obtener}(p2, \text{partidas}(s)), \\ \text{constr1} \equiv & \text{casas}(\text{sim1}) \cup \text{comercios}(\text{sim1}), \\ \text{constr2} \equiv & \text{casas}(\text{sim2}) \cup \text{comercios}(\text{sim2}) \end{aligned}$$

avanzarTurnoValido : server s × Nombre p × dicc(Pos × Construcccion) cs → boolean

$$\begin{aligned} \text{avanzarTurnoValido}(s, p, cs) \equiv & \text{def?}(p, \text{partidas}(s)) \wedge \\ & p \notin \text{congeladas}(s) \wedge \\ & \neg \text{vacía?}(\text{claves}(cs)) \wedge_L \\ & (\forall pos : Pos)(pos \in \text{claves}(cs) \Rightarrow_L \\ & \quad \text{obtener}(pos, cs) \in \{\text{"casa"}, \text{"comercio"}\} \wedge \\ & \quad \neg \text{sobreRio}(pos, \text{mapa}(\text{sim})) \wedge \\ & \quad \neg \text{def?}(pos, \text{casas}(\text{sim})) \wedge \\ & \quad \neg \text{def?}(pos, \text{comercios}(\text{sim})) \\ &) \\ \text{donde } \text{sim} \equiv & \text{obtener}(p, \text{partidas}(s)) \end{aligned}$$

• ∪ • : dicc(α × β) × dicc(α × β) → dicc(α, β)

a ∪ b ≡ *_definir*(a, b, claves(b))

_union : dicc(α × β) × dicc(α × β) b × conj(α) cs → dicc(α, β) {cs ⊆ claves(b)}

_union(a, b, cs) ≡ **if** *vacío?*(cs) **then**
 a
else
 _union(*definir*(*dameUno*(cs), obtener(*dameUno*(cs), b)), b, *sinUno*(cs))
fi

2. Módulos de referencia

2.1. Módulo Mapa

Interfaz

se explica con: MAPA

géneros: mapa

TP de Especificación y Diseño Operaciones básicas de mapa

CREAR(**in** $hs : \text{conj}(\text{Nat})$, **in** $vs : \text{conj}(\text{Nat})$) $\rightarrow res : \text{mapa}$

Pre $\equiv \{\text{true}\}$

Post $\equiv \{res =_{\text{obs}} \text{mapa}(hs, vs)\}$

Complejidad: $O(\text{copy}(hs), \text{copy}(vs))$

Descripción: crea un mapa

completar

Representación

TP de Especificación y Diseño Representación de mapa

Un mapa contiene rios infinitos horizontales y verticales. Los rios se representan como conjuntos lineales de naturales que indican la posición en los ejes de los ríos.

mapa **se representa con** $estr$

donde $estr$ es $\text{tupla}(\text{horizontales} : \text{conj}(\text{Nat}), \text{verticales} : \text{conj}(\text{Nat}))$

$\text{Rep} : \text{estr} \rightarrow \text{bool}$

$\text{Rep}(e) \equiv \text{true} \iff \text{true}$

$\text{Abs} : \text{estr } m \rightarrow \text{mapa}$

$\{\text{Rep}(m)\}$

$\text{Abs}(m) \equiv \text{horizontales}(m) = \text{estr.horizontales} \wedge \text{verticales}(m) = \text{estr.verticales}$

Algoritmos

crear(**in** $hs : \text{conj}(\text{Nat})$, **in** $vs : \text{conj}(\text{Nat})$) $\rightarrow res : \text{estr}$

1: $\text{estr.horizontales} \leftarrow hs$

2: $\text{estr.verticales} \leftarrow vs$ **return** $estr$

Complejidad: $O(\text{copy}(hs) + \text{copy}(vs))$

completar

2.2. Módulo SimCity

2.3. Módulo Servidor