



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**

## 1.2. SimCity

### TAD SIMCITY

#### igualdad observacional

$$(\forall s, s' : \text{SimCity}) \left( s =_{\text{obs}} s' \iff \left( \begin{array}{l} \text{mapa}(s) =_{\text{obs}} \text{mapa}(s') \wedge_{\text{L}} \\ \text{casas}(s) =_{\text{obs}} \text{casas}(s') \wedge \\ \text{comercios}(s) =_{\text{obs}} \text{comercios}(s') \wedge \\ \text{popularidad}(s) =_{\text{obs}} \text{popularidad}(s') \end{array} \right) \right)$$

**géneros**      SimCity

**exporta**      **completar**

**usa**            **completar**

#### observadores básicos

mapa            : SimCity  $\longrightarrow$  Mapa  
casas            : SimCity  $\longrightarrow$  dicc(Pos, Nivel)  
comercios       : SimCity  $\longrightarrow$  dicc(Pos, Nivel)  
popularidad    : SimCity  $\longrightarrow$  Nat

#### generadores

iniciar            : Mapa  $\longrightarrow$  SimCity  
avanzarTurno : SimCity  $s \times$  dicc(Pos  $\times$  Construcccion)  $cs \longrightarrow$  SimCity  
 $\left\{ \begin{array}{l} (\forall p : \text{Pos}) (\text{def?}(p, cs) \Rightarrow_{\text{L}} (\neg p \in \text{claves}(\text{construcc}(s)) \wedge \\ \neg \pi_0(p) \in \text{horizontales}(\text{mapa}(s)) \wedge \neg \pi_1(p) \in \text{verticales}(\text{mapa}(s)) \wedge \\ (\text{obtener}(p, cs) =_{\text{obs}} 1 \vee \text{obtener}(p, cs) =_{\text{obs}} 2))) \end{array} \right\}$   
unir               : SimCity  $a \times$  SimCity  $b \longrightarrow$  SimCity  
 $\left\{ \begin{array}{l} (\forall p : \text{Pos}) (\text{def?}(p, \text{construcc}(a)) \Rightarrow_{\text{L}} \\ (\neg \pi_0(p) \in \text{horizontales}(\text{mapa}(b)) \wedge \neg \pi_1(p) \in \text{verticales}(\text{mapa}(b)) \wedge \\ (p \in \text{masNivel}(a) \Rightarrow \neg p \in \text{construcc}(b)))) \wedge \\ (\forall p : \text{Pos}) (\text{def?}(p, \text{construcc}(b)) \Rightarrow_{\text{L}} \\ (\neg \pi_0(p) \in \text{horizontales}(\text{mapa}(a)) \wedge \neg \pi_1(p) \in \text{verticales}(\text{mapa}(a)) \wedge \\ (p \in \text{masNivel}(b) \Rightarrow \neg p \in \text{construcc}(a)))) \end{array} \right\}$

#### otras operaciones

turnos            : SimCity  $\longrightarrow$  Nat  
construcc        : SimCity  $\longrightarrow$  dicc(Pos, Nivel)  
unirDicc         : dicc(Pos  $\times$  Nivel)  $\times$  dicc(Pos  $\times$  Nivel)  $\longrightarrow$  dicc(Pos, Nivel)  
masNivel         : SimCity  $\longrightarrow$  conj(Pos)  
masNivelAux     : dicc(Pos  $\times$  Nivel)  $\times$  Nat  $\longrightarrow$  conj(Pos)  
nivelMaximo     : dicc(Pos  $\times$  Nivel)  $\longrightarrow$  Nat  
agCasas          : dicc(Pos  $\times$  Nivel)  $\times$  dicc(Pos  $\times$  Construcccion)  $\longrightarrow$  dicc(Pos, Nivel)  
agComercios     : dicc(Pos  $\times$  Nivel)  $\times$  dicc(Pos  $\times$  Construcccion)  $\longrightarrow$  dicc(Pos, Nivel)  
nivelCom         : Pos  $\times$  SimCity  $\longrightarrow$  Nat  
conjManhatt     : Pos  $\times$  dicc(Pos  $\times$  Nivel)  $\longrightarrow$  dicc(Pos, Nivel)  
distManhatt     : Pos  $\times$  Pos  $\longrightarrow$  Nat  
sacarRepes       : dicc(Pos  $\times$  Construcccion)  $\times$  dicc(Pos  $\times$  Construcccion)  $\longrightarrow$  dicc(Pos, Construcccion)

**axiomas**       $\forall s, s' : \text{simcity}, \forall cs, cs' : \text{dicc}(\text{Pos}, \text{Construcccion})$

mapa(iniciar(m))  $\equiv$  m  
mapa(avanzarTurno( $s, cs$ ))  $\equiv$  mapa( $s$ )  
mapa(unir( $s, s'$ ))  $\equiv$  crear(horizontales( $s$ )  $\cup$  horizontales( $s'$ ),  
verticales( $s$ )  $\cup$  verticales( $s'$ ))  
casas(iniciar(m))  $\equiv$  vacio  
casas(avanzarTurno( $s, cs$ ))  $\equiv$  agCasas(casas( $s$ ),  $cs$ )  
casas(unir( $s, s'$ ))  $\equiv$  agCasas(casas( $s$ ), sacarRepes(construcc( $s$ ), construcc( $s'$ )))

agCasas( $cs, cs'$ )	$\equiv$ <b>if</b> <i>vacio?</i> (claves( $cs'$ )) <b>then</b> $cs$ <b>else</b> <b>if</b> obtener(dameUno(claves( $cs'$ )), $cs'$ ) = <sub>obs</sub> 1 <b>then</b> agCasas(definir(dameUno(claves( $cs'$ )), 1, $cs$ ), borrar(dameUno(claves( $cs'$ )), $cs'$ )) <b>else</b> agCasas( $cs$ , borrar(dameUno(claves( $cs'$ )), $cs'$ )) <b>fi fi</b>
comercios(iniciar( $m$ ))	$\equiv$ vacio
comercios(avanzarTurno( $s, cs$ ))	$\equiv$ agComercios(comercios( $s$ ), $cs$ )
comercios(unir( $s, s'$ ))	$\equiv$ unirConstrucc( $s$ , casas( $s$ ), casas( $s'$ ))
agComercios( $cs, cs'$ )	$\equiv$ <b>if</b> <i>vacio?</i> (claves( $cs'$ )) <b>then</b> $cs$ <b>else</b> <b>if</b> obtener(dameUno(claves( $cs'$ )), $cs'$ ) = <sub>obs</sub> 2 <b>then</b> agComercios(definir(dameUno(claves( $cs'$ )), nivelCom(dameUno(claves( $cs'$ )), $s$ ), $cs$ ), borrar(dameUno(claves( $cs'$ )), $cs'$ )) <b>else</b> agComercios( $cs$ , borrar(dameUno(claves( $cs'$ )), $cs'$ )) <b>fi fi</b>
nivelCom( $p, s$ )	$\equiv$ <b>if</b> $\neg$ <i>vacio?</i> (manhattan( $p$ , casas( $s$ ))) <b>then</b> nivelMaximo(manhattan( $p$ ), casas( $s$ )) <b>else</b> 1 <b>fi</b>
conjManhatt( $p, cs$ )	$\equiv$ <b>if</b> <i>vacio?</i> ( $cs$ ) <b>then</b> $\emptyset$ <b>else</b> <b>if</b> distManhatt( $p$ , dameUno(claves( $cs$ ))) $\leq$ 3 <b>then</b> definir(dameUno(claves( $cs$ )), obtener(dameUno(claves( $cs$ )), $cs$ ), conjManhatt( $p$ , borrar(dameUno(claves( $cs$ )), $cs$ ))) <b>else</b> conjManhatt( $p$ , borrar(dameUno(claves( $cs$ )), $cs$ )) <b>fi</b>
distManhatt( $p, q$ )	$\equiv$ <b>if</b> $\pi_0(p) < \pi_0(q)$ <b>then</b> $q - p$ <b>else</b> $p - q$ <b>fi</b> + <b>if</b> $\pi_1(p) < \pi_1(q)$ <b>then</b> $q - p$ <b>else</b> $p - q$ <b>fi</b>
popularidad(iniciar( $m$ ))	$\equiv$ 0
popularidad(avanzarTurno( $s, cs$ ))	$\equiv$ popularidad( $s$ )
popularidad(unir( $s, s'$ ))	$\equiv$ popularidad( $s$ ) + 1
turnos(iniciar( $m$ ))	$\equiv$ 0
turnos(avanzarTurno( $s, cs$ ))	$\equiv$ turnos( $s$ ) + 1
turnos(unir( $s, s'$ ))	$\equiv$ <b>if</b> turnos( $s$ ) < turnos( $s'$ ) <b>then</b> turnos( $s'$ ) <b>else</b> turnos( $s$ ) <b>fi</b>
construcc( $s$ )	$\equiv$ unirDicc(casas( $s$ ), comercios( $s$ ))
unirDicc( $cs, cs'$ )	$\equiv$ <b>if</b> <i>vacio?</i> (claves( $cs'$ )) <b>then</b> $cs$ <b>else</b> definir(dameUno(claves( $cs'$ )), obtener(dameUno(claves( $cs'$ )), $cs'$ ), unirDicc( $cs$ , borrar(dameUno(claves( $cs'$ )), $cs'$ ))) <b>fi</b>
masNivel( $s$ )	$\equiv$ masNivelAux(construcc( $s$ ), nivelMaximo(construcc( $s$ )))
masNivelAux( $cs, n$ )	$\equiv$ <b>if</b> <i>vacio?</i> ( $cs$ ) <b>then</b> $\emptyset$ <b>else</b> <b>if</b> obtener(dameUno(claves( $cs$ )), $cs$ ) = <sub>obs</sub> $n$ <b>then</b> ag(dameUno(claves( $cs$ )), masNivelAux(borrar(dameUno(claves( $cs$ )), $cs$ ), $n$ )) <b>else</b> masNivelAux(borrar(dameUno(claves( $cs$ )), $cs$ ), $n$ ) <b>fi fi</b>
nivelMaximo( $cs$ )	$\equiv$ <b>if</b> <i>vacio?</i> ( $cs$ ) <b>then</b> 0 <b>else</b> max(obtener(dameUno(claves( $cs$ )), $cs$ ), nivelMaximo(borrar(dameUno(claves( $cs$ )), $cs$ )))

```
sacarRepes(cs, cs')      ≡ if vacio?(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**

### 1.3. Servidor

#### TAD SERVIDOR

**géneros**      server

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

**usa**            SimCity, Mapa, Nombre, Pos, Construccion, 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  $s \times \text{Nombre } p \times \text{dicc}(\text{Pos} \times \text{Construccion}) cs \rightarrow \text{server} \quad \{*\text{avanzarTurnoValido}(s, p, cs)\}$

#### 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**