TAD IP **es** String
TAD Interfaz **es** Nat
TAD Prioridad **es** Nat
TAD Id **es** Nat
TAD Paquete **es** Tupla(id:Id, prioridad:Prioridad, origen:Compu, destino:Compu)
TAD Compu **es** Tupla(ip:IP, interfaces:conj(Interfaz))

TAD Red

géneros red

exporta red, igualdad observacional, observadores basicos, generadores, vecinos, usaInterfaz?, caminosMinimos, hayCamino?

igualdad observacional

$$(\forall r1, r2 : \text{red}) \left(\begin{array}{l} r1 =_{\text{obs}} r2 \iff \\ \quad \text{computadoras}(r1) =_{\text{obs}} \text{computadoras}(r2) \wedge_{\text{L}} \\ \quad (\forall c_1, c_2 : \textit{compu})(c_1 \in \text{computadoras}(r_1) \wedge c_2 \in \text{computadoras}(r_1)) \Rightarrow_{\text{L}} \\ \quad\quad (\text{conectadas?}(r_1, c_1, c_2) =_{\text{obs}} \text{conectadas?}(r_2, c_1, c_2)) \wedge_{\text{L}} \\ \quad \text{conectadas?}(r_1, c_1, c_2) \Rightarrow_{\text{L}} \\ \quad\quad (\text{interfazUsada}(r_1, c_1, c_2) =_{\text{obs}} \text{interfazUsada}(r_2, c_1, c_2)) \end{array} \right)$$

observadores básicos

computadoras	: red	$\longrightarrow$	conj(compu)	
conectadas?	: red r $\times$ compu c_1 $\times$ compu c_2	$\longrightarrow$	bool	$\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$
interfazUsada	: red r $\times$ compu c_1 $\times$ compu c_2	$\longrightarrow$	interfaz	$\{\text{conectadas?}(r, c_1, c_2)\}$

generadores

iniciarRed	:	$\longrightarrow$	red	
agregarComputadora	: red r $\times$ compu c	$\longrightarrow$	red	$\{(\forall\ c'\text{:compu})\ (c' \in \text{computadoras}(r)\ \Rightarrow\ \text{ip}(c) \neq \text{ip}(c'))\}$
conectar	: red r $\times$ compu c_1 $\times$ interfaz i_1 $\times$ compu c_2 $\times$ interfaz i_2	$\longrightarrow$	red	$\left\{ \begin{array}{l} c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r) \wedge \\ \text{ip}(c_1) \neq \text{ip}(c_2) \wedge \neg\text{conectadas?}(r, c_1, c_2) \wedge \\ \neg\text{usaInterfaz?}(r, c_1, i_1) \wedge \neg\text{usaInterfaz?}(r, c_2, i_2) \end{array} \right\}$

otras operaciones

vecinos	: red r $\times$ compu c	$\longrightarrow$	conj(compu)	$\{c \in \text{computadoras}(r)\}$
auxVecinos	: red r $\times$ compu c $\times$ conj(compu)	$\longrightarrow$	conj(compu)	$\{c \in \text{computadoras}(r)\}$
usaInterfaz?	: red r $\times$ compu c $\times$ interfaz	$\longrightarrow$	bool	$\{c \in \text{computadoras}(r)\}$
interfacesUsadas	: red r $\times$ compu c $\times$ conj(compu)	$\longrightarrow$	conj(interfaz)	$\{c \in \text{computadoras}(r)\}$
caminos	: red r $\times$ compu c_1 $\times$ compu c_1	$\longrightarrow$	conj(secu(compu))	$\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$
auxCaminos	: red r $\times$ compu c_1 $\times$ compu c_1 $\times$ secu(compu) $\times$ conj(compu)	$\longrightarrow$	conj(secu(compu))	$\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$
hayCamino?	: red r $\times$ compu c_1 $\times$ compu c_2	$\longrightarrow$	bool	$\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$
caminosMinimos	: red r $\times$ compu c_1 $\times$ compu c_2	$\longrightarrow$	conj(secu(compu))	$\{c_1 \in \text{computadoras}(r) \wedge c_2 \in \text{computadoras}(r)\}$
auxMinimos	: conj(secu(compu))	$\longrightarrow$	conj(secu(compu))	

axiomas

computadoras(iniciarRed)	$\equiv$	$\emptyset$
computadoras(agregarComputadora(r, c))	$\equiv$	Ag(c, computadoras(r))
computadoras(conectar(r, c_1, i_1, c_2, i_2))	$\equiv$	computadoras(r)
conectadas?(agregarComputadora(r, c), c_1, c_2)	$\equiv$	$\neg(c = c_1 \vee c = c_2) \vee$ conectadas?(r, c_1, c_2)
conectadas?(conectar(r, c_1', i_1, c_2', i_2), c_1, c_2)	$\equiv$	$(c_1 = c_1' \wedge c_2 = c_2') \vee (c_1 = c_2' \wedge c_2 = c_1') \vee$ conectadas?(r, c_1, c_2)

interfazUsada(agregarComputadora(r, c), c_1, c_2) $\equiv$ interfazUsada(r, c_1, c_2)

interfazUsada(conectar(r, c'_1, i_1, c'_2, i_2), c_1, c_2) $\equiv$ **if** $(c_1 = c'_1 \wedge c_2 = c'_2)$ **then**
i_1
else
if $(c_1 = c'_2 \wedge c_2 = c'_1)$ **then**
i_2
else
interfazUsada(r, c_1, c_2)
fi
fi

vecinos(r, c) $\equiv$ auxVecinos(r, c, computadoras(r))

auxVecinos(r, c, cs) $\equiv$ **if** $\emptyset$?(cs) **then**
$\emptyset$
else
if conectados?(r, c, dameUno(cs)) **then**
Ag(dameUno(cs), auxVecinos(r, c, sinUno(cs)))
else
auxVecinos(r, c, sinUno(cs))
fi
fi

usaInterfaz?(r, c, i) $\equiv$ $i \in$ interfacesUsadas(r, c, computadoras(r))

interfacesUsadas(r, c, cs) $\equiv$ **if** $\emptyset$?(cs) **then**
$\emptyset$
else
if conectados?(r, c, dameUno(cs)) **then**
Ag(interfazUsada(r, c, dameUno(cs)),
interfacesUsadas(r, c, sinUno(cs)))
else
FiltrarInterfacesUsadas(SinUno(ci), c)
fi
fi

caminos(r, c_1, c_2) $\equiv$ auxCaminos(r, c_1, c_2, $c_1 \bullet <>$, vecinos(r, c_1))

auxCaminos(r, c_1, c_2, *recorrido*, *candidatos*) $\equiv$ **if** $\emptyset$?(*candidatos*) **then**
$\emptyset$
else
if ult(*recorrido*) $= c_2$ **then**
ag(*recorrido*, $\emptyset$)
else
if $\neg$esta?(dameUno(*candidatos*), *recorrido*)
then
auxCaminos(r, c_1, c_2,
recorrido $\circ$ dameUno(*candidatos*),
vecinos(r, dameUno(*candidatos*)))
$\cup$ auxCaminos(r, c_1, c_2, *recorrido*,
sinUno(*candidatos*))
else
auxCaminos(r, c_1, c_2, *recorrido*,
sinUno(*candidatos*))
fi
fi
fi

hayCamino?(r, $c1$, $c2$) $\equiv$ #(caminos(r, $c1$, $c2$)) > 0

caminosMinimos(r, $c1$, $c2$) $\equiv$ auxMinimos(caminos(r, $c1$, $c2$))

```
auxMinimos(cc) ≡ if ∅?(cc) then
                    ∅
                 else
                    if #(cc) = 1 then
                       ag(dameUno(c), ∅)
                    else
                       if long(dameUno(cc)) < long(dameUno(auxMinimos(sinUno(cc)))) then
                          ag(dameUno(cc), ∅)
                       else
                          if long(dameUno(cc)) = long(dameUno(auxMinimos(sinUno(cc)))) then
                             ag(dameUno(c), auxMinimos(sinUno(cc)))
                          else
                             auxMinimos(sinUno(cc))
                          fi
                       fi
                    fi
                 fi
```

Fin TAD

TAD DCNet

exporta dcnet, igualdad observacioanl, observadores basicos, generadores, paqueteEnTransito?, laQueMasEnvio

géneros dcnet

igualdad observacional

$$(\forall s1, s2 : \text{dcnet}) \left(\begin{array}{l} s1 =_{\text{obs}} s2 \iff \\ \quad \text{red}(s1) =_{\text{obs}} \text{red}(s2) \wedge_{\text{L}} \\ \quad (\forall c : ip)(c \in \text{computadoras}(\text{red}(s1)) \Rightarrow_{\text{L}} \text{EnEspera}(s1, c) =_{\text{obs}} \text{EnEspera}(s2, c)) \wedge_{\text{L}} \\ \quad (\forall p : \text{paquete})((\text{paqueteEnTransito?}(s1, p)) \Rightarrow_{\text{L}} \\ \quad\quad \text{caminoRecorrido}(s1, p) =_{\text{obs}} \text{caminoRecorrido}(s2, p)) \wedge_{\text{L}} \\ \quad (\forall c : \text{ip})((c \in \text{computadoras}(\text{red}(s1))) \Rightarrow_{\text{L}} \\ \quad\quad \text{cantidadEnviados}(s1, c) =_{\text{obs}} \text{cantidadEnviados}(s2, c)) \end{array} \right)$$

observadores básicos

red	: dcnet	⟶ red	
caminoRecorrido	: dcnet s × paquete p	⟶ secu(compu)	{paqueteEnTransito?(s,p)}
cantidadEnviados	: dcnet s × compu c	⟶ nat	{$c \in$ computadoras(red(s))}
enEspera	: dcnet s × compu c	⟶ conj(paquete)	{$c \in$ Computadoras(Red(s))}

generadores

iniciarDCNet	: red	⟶ dcnet	
crearPaquete	: dcnet s × paquete p	⟶ dcnet	$\left\{ \begin{array}{l} \neg(\ (\exists\ p'\text{:paquete})\ (\text{paqueteEnTransito}(s,p') \wedge \text{id}(p') = \text{id}(p)) \\ \quad \wedge\ \text{origen}(p) \in \text{computadoras}(\text{red}(s)) \\ \quad \wedge_{\text{L}}\ \text{destino}(p) \in \text{computadoras}(\text{red}(s)) \\ \quad \wedge_{\text{L}}\ \text{hayCamino?}(\text{red}(s),\ \text{origen}(p),\ \text{destino}(p)\ ) \end{array} \right\}$
avanzarSegundo	: dcnet	⟶ dcnet	

otras operaciones

proximoPaquete	: dcnet s × compu c	⟶ paquete {$c \in$ computadoras(Red(s)) $\wedge_{\text{L}}$ #(enEspera(s, c)) > 0}
prioritarios	: conj(paquete)	⟶ conj(paquete)
computadoraActual	: dcnet s × paquete p	⟶ compu {paqueteEnTransito?(s, p)}
proximaComputadora	: dcnet s × paquete p	⟶ compu {paqueteEnTransito?(s, p)}
paquetesConOtroDestino	: conj(paquete) × compu	⟶ conj(paquete)
paquetesEntrantes	: dcnet s × compu c × conj(compu) cc	⟶ conj(paquete) {$c \in$ computadoras(red(s)) $\wedge$ $cc \subseteq$ computadoras(red(s))}

paqueteEnTransito? : dcnet × paquete ⟶ bool
laQueMasEnvio : dcnet ⟶ compu
lasQueMasEnviaron : dcnet s × conj(compu) cc ⟶ conj(compu) {$cc \subseteq$ computadoras(red(s))}

axiomas

red(iniciarDCNet(r, n)) ≡ r
red(crearPaquete(s, p)) ≡ red(s)
red(avanzarSegundo(s)) ≡ red(s)

caminoRecorrido(crearPaquete(s, p'), p) ≡ **if** $p = p'$ **then**
origen(p) • <>
else
caminoRecorrido(s, p)
fi
caminoRecorrido(avanzarSegundo(s), p) ≡ **if** proximoPaquete(s, computadoraActual(s, p)) ≠ p **then**
caminoRecorrido(s, p)
else
caminoRecorrido(s, p) ∘ proximaComputadora(s, p)
fi

cantidadEnviados(iniciarDCNet(r, n), c) ≡ 0
cantidadEnviados(crearPaquete(s, p), c) ≡ cantidadEnviados(s, c)
cantidadEnviados(avanzarSegundo(s), c) ≡ (**if** #(enEspera(s, c)) > 0 **then** 1 **else** 0 **fi**) + cantidadEnviados(s, c)

enEspera(iniciarDCNet(r, n), c) ≡ ∅
enEspera(crearPaquete(s, p), c) ≡ **if** origen(p) = c **then**
ag(p, enEspera(s, c))
else
enEspera(s, c)
fi
enEspera(avanzarSegundo(s), c) ≡ (enEspera(s, c) - (**if** #(enEspera(s, c)) > 0 **then** proximoPaquete(s, c) **else** ∅ **fi**)) ∪ paquetesConOtroDestino(paquetesEntrantes(s, c, computadoras(red(s))), c)

proximoPaquete(s, c) ≡ dameUno(prioritarios(enEspera(s, c)))

prioritarios(cp) ≡ **if** ∅?(cp) **then**
∅
else
if #(cp) = 1 **then**
cp
else
if prioridad(dameUno(cp)) < prioridad(dameUno(prioritarios(sinUno(cp))))
then
ag(dameUno(cp), ∅)
else
if prioridad(DameUno(cp)) = prioridad(dameUno(prioritarios(sinUno(cp))))
then
ag(dameUno(cp), prioritarios(sinUno(cp)))
else
prioritarios(sinUno(cp))
fi
fi
fi
fi

computadoraActual(s, p) ≡ ult(caminoRecorrido(s, p))

proximaComputadora(s, p) $\equiv$ prim(fin(dameUno(caminosMinimos(red(s), computadoraActual(s, p), destino(p)))))

paquetesConOtroDestino(cp, c) $\equiv$ **if** $\emptyset?(cc)$ **then**
$\emptyset$
else
if destino(dameUno(cp)) $= c$ **then**
paquetesConOtroDestino(sinUno(cp), c)
else
ag(dameUno(cp), paquetesConOtroDestino(sinUno(cp), c))
fi
fi

paquetesEntrantes(s, c, cc) $\equiv$ **if** $\emptyset?(cc)$ **then**
$\emptyset$
else
(**if** $\#$(enEspera(s, dameUno(cc))) > 0 **then**
if proximaComputadora(s, proximoPaquete(s, dameUno(cc))) $= c$ **then**
ag(proximoPaquete(s, dameUno(cc)), $\emptyset$)
else
$\emptyset$
fi
else
$\emptyset$
fi) $\cup$ paquetesEntrantes(s, c, sinUno(cc))
fi

paqueteEnTransito?(iniciarDCNet(r, n), p) $\equiv$ false
paqueteEnTransito?(crearPaquete(s, p'), p) $\equiv$ $p = p' \vee$ paqueteEnTransito?(s, p)
paqueteEnTransito?(avanzarSegundo(s)) $\equiv$ paqueteEnTransito?(s, p) $\wedge_{\text{L}}$
(proximoPaquete(s, computadoraActual(s, p)) $\neq p \vee$
proximaComputadora(s, p) $\neq$ destino(p))

laQueMasEnvio(s) $\equiv$ dameUno(lasQueMasEnviaron(s, computadoras(Red(s))))

lasQueMasEnviaron(s, cc) $\equiv$ **if** $\emptyset?(cc)$ **then**
$\emptyset$
else
if $\#(cc) = 1$ **then**
cc
else
if cantidadEnviados(s, dameUno(cc)) $>$ cantidadEnviados(s, dameUno(lasQueMasEnviaron(sinUno(cc)))) **then**
ag(dameUno(cc), $\emptyset$)
else
if cantidadEnviados(s, dameUno(cc)) $=$ cantidadEnviados(s, dameUno(lasQueMasEnviaron(sinUno(cc)))) **then**
ag(dameUno(cc), lasQueMasEnviaron(sinUno(cc)))
else
lasQueMasEnviaron(sinUno(cc))
fi
fi
fi
fi

Fin TAD