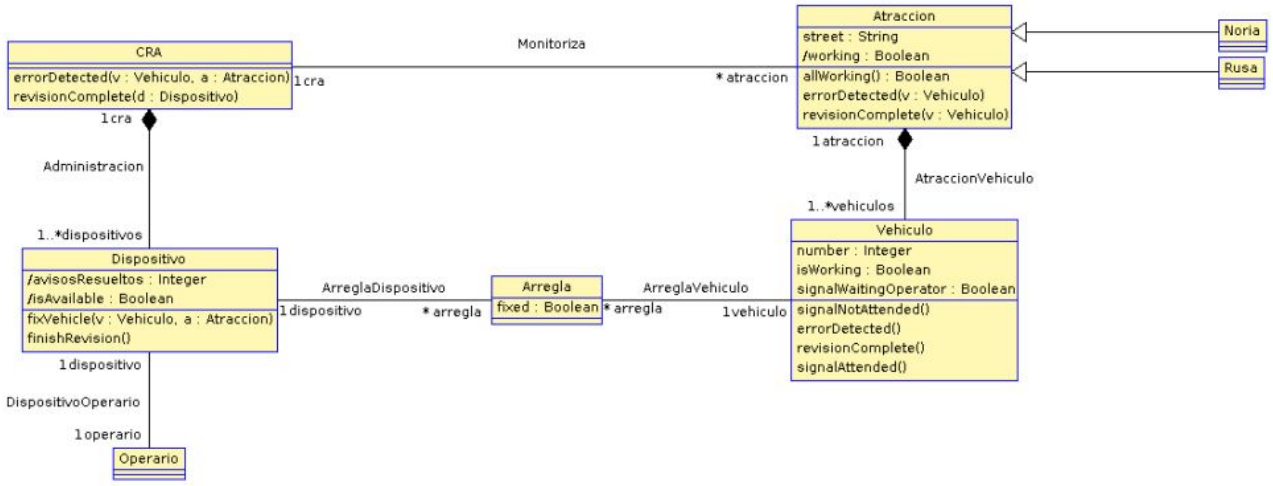
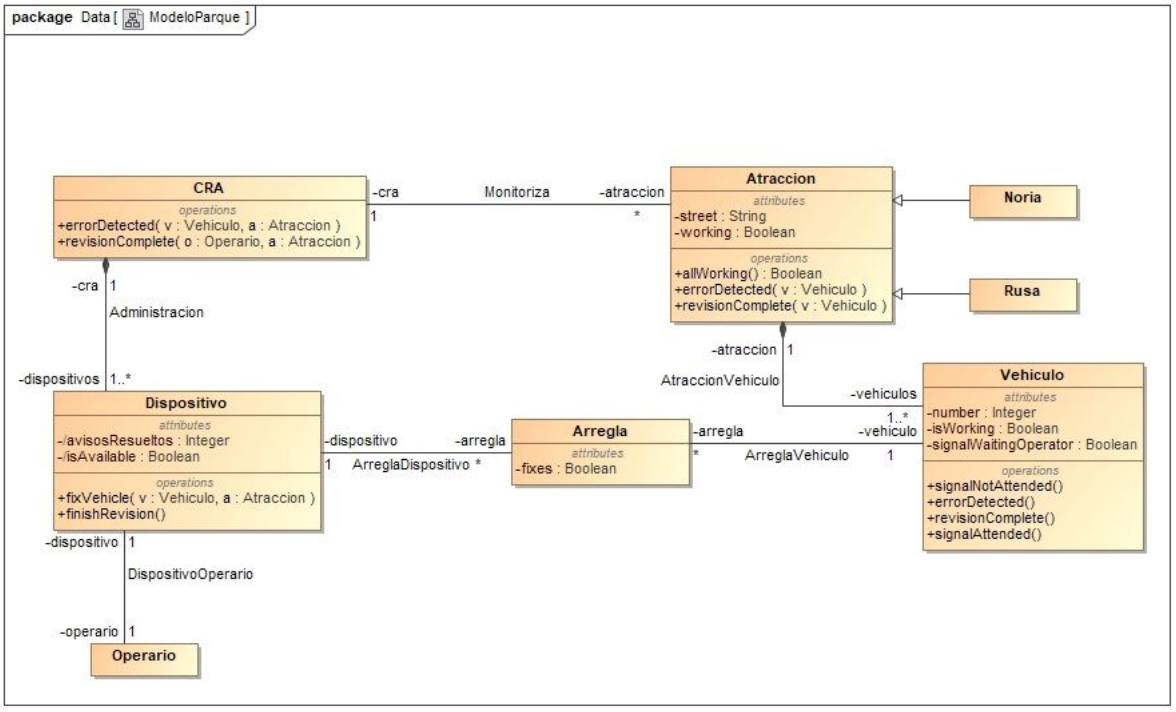
MODELADO DE UN SISTEMA DE REPARACIONES DE UN PARQUE DE ATRACCIONES

Autores: José Francisco Aldana Martín, Serafín Cortes Ramirez, Tomás Manuel García Ruiz, Jose Moya Vargas, Miriam Rodríguez Álvarez.

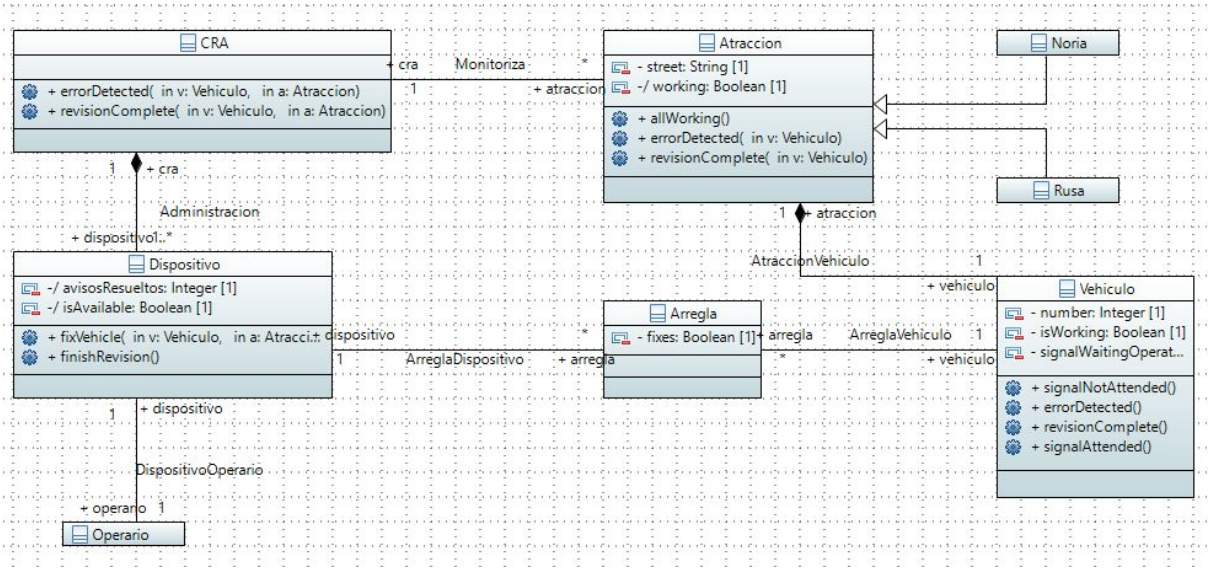
Modelo en USE:



Modelo en MagicDraw:



Modelo en Papyrus:



A continuación se detalla el porqué de las decisiones tomadas en nuestro modelo del sistema y se explicará cada uno de los elementos implicados en el mismo.
En nuestro modelo suponemos una única CRA (Central Receptora de Averías) la cual administra uno o más Dispositivos controlados por Operarios, de los cuales hablaremos más adelante, y monitoriza muchas Atracciones.

Las atracciones pueden ser de tipo Noria o Montaña Rusa (Rusa por abreviar) y tienen, a su vez, uno o más Vehículos. Los Vehículos son los encargados de avisar de una avería en sí mismo a la Atracción y al CRA, a través de la función errorDetected() y cambiando su estado mediante el atributo isWorking. La CRA se encargará de atender dicha señal buscando a un operario libre que vaya a la calle de la atracción (atributo de la clase mencionada) mediante el dispositivo relacionado con el operario y la función errorDetected, en caso de no haber ninguno disponible la CRA enviará una señal de vuelta al Vehículo invocando la función signalNotAttended() la cual cambiará el estado del Vehículo en cuestión mediante signalWaitingOperator a True indicando que espera un operario para su reparación.

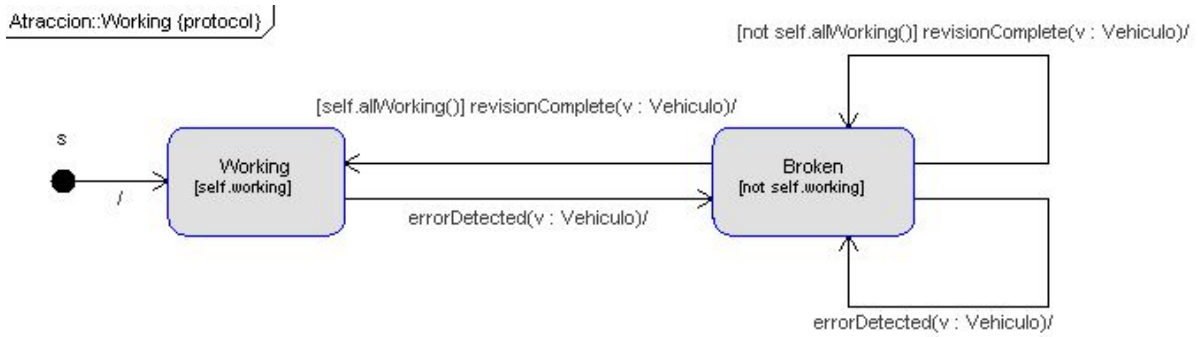
Por otro lado, los dispositivos controlan el número de avisos resueltos (avisosResueltos) calculado de forma derivada realizando la suma de cuantos “Arregla” tiene dicho dispositivo. Arregla simplemente es una entidad para dicho control de qué Dispositivo de un operario repara un Vehículo. Además el dispositivo tiene una variable que indica si el operario se encuentra disponible o no (isAvailable), cuando un operario cambia su estado de ocupado a disponible y existe algún Vehículo con la señal activada la CRA se encargará de enviar a reparar a dicho operario el vehículo estropeado.

Restricciones:

Lenguaje natural	OCL
Un operario solo puede estar arreglando una atracción.	context Dispositivo inv soloArreglaUno: self.arregla->select(a:Arregla|not a.fixed)->isUnique(a:Arregla|not a.fixed)
Un vehículo no puede estar funcionando y la atracción no.	context Vehiculo inv ambosSinFuncionar: (not self.isWorking) implies (not self.atraccion.working)
Una atracción no puede estar funcionando y un vehículo siendo atendido.	context Vehiculo inv atraccionNoFuncionaWhileVehiculoAtendido: self.arregla->exists(a:Arregla|not a.fixed) implies not self.atraccion.working
Si un vehiculo esta siendo arreglado, no puede estar esperando a un operario.	context Vehiculo inv vehiculoArreglandoNoEsperaOperario: self.isWorking implies not self.signalWaitingOperator
Un dispositivo/operario no puede estar ocupado si no está arreglando un vehículo.	context Dispositivo inv OperarioOcupadoArreglando: not self.isAvailable implies self.arregla->select(a|not a.fixed)->size=1
Si una atracción tiene un vehículo que está siendo arreglado, esta no puede estar en funcionamiento.	context Dispositivo inv OperarioOcupadoArreglando: not self.isAvailable implies self.arregla->select(a|not a.fixed)->size=1
Solo hay una CRA	context CRA inv singleton: CRA.allInstances->size()<=1

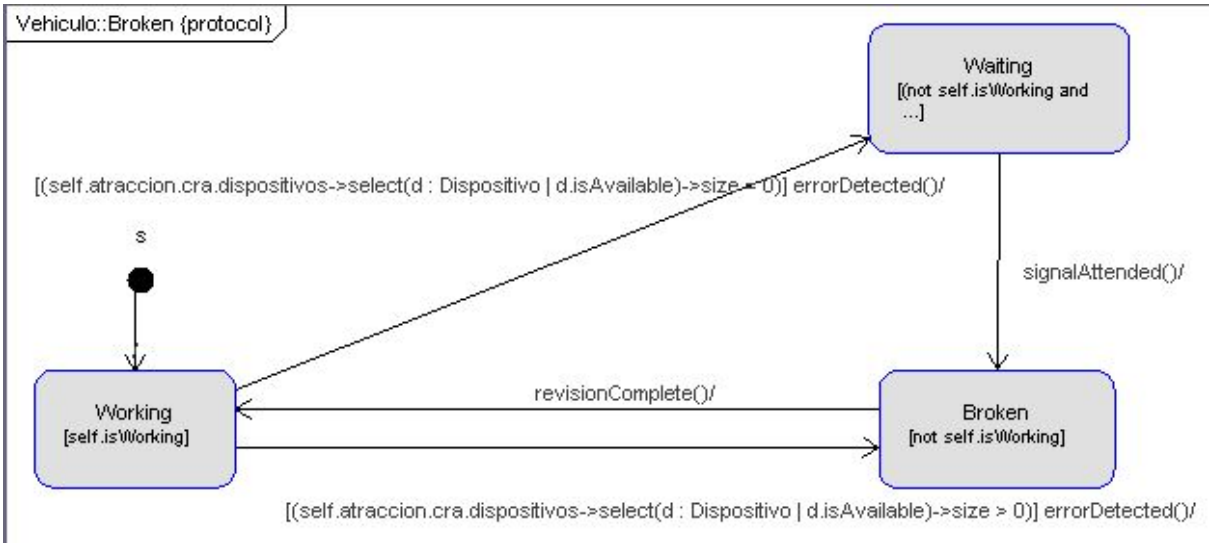
Maquinas de Estado:

Atracción:



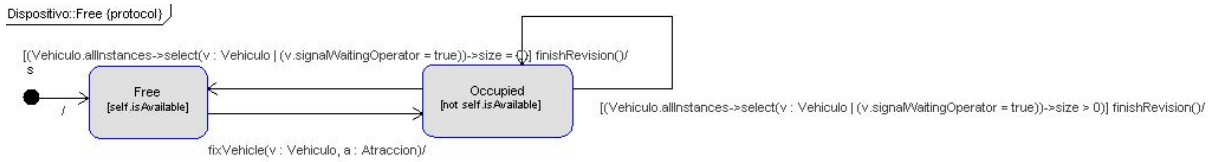

```
statemachines
   psm Working
     states
       s:initial
       Working [self.working]
       Broken [not self.working]
     transitions
       s->Working
       Working->Broken {errorDetected()}
       Broken->Working {[self.allWorking()] revisionComplete()}
       Broken->Broken {[not self.allWorking()] revisionComplete()}
       Broken->Broken {errorDetected()}
   end
end
```

Vehículo:




```
  statemachines
    psm Broken
      states
        s:initial
        Working [self.isWorking]
        Broken [not self.isWorking]
        Waiting [not self.isWorking and self.signalWaitingOperator]
      transitions
        s->Working
        Working->Broken {[self.atraccion.cra.dispositivos->select(d|d.isAvailable)->size()>0]
errorDetected()}
        Working->Waiting {[self.atraccion.cra.dispositivos->select(d|d.isAvailable)->size()=0]
errorDetected()}
        Broken->Working {revisionComplete()}
        Waiting->Broken {signalAttended()}
    end
end
```

Dispositivo:

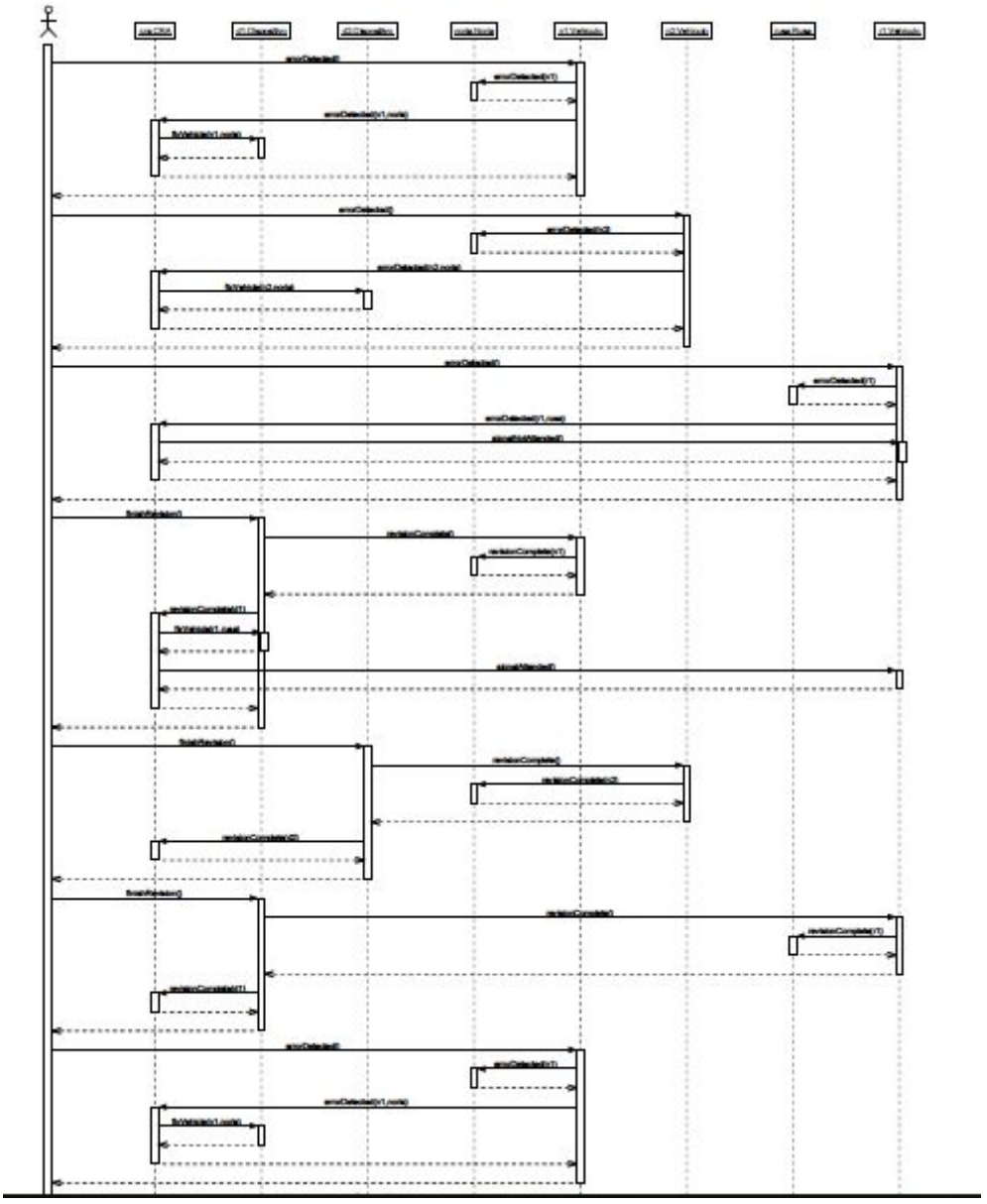



```
statemachines
    psm Free
      states
        s:initial
        Occupied[not self.isAvailable]
        Free[self.isAvailable]
      transitions
       s->Free
       Free->Occupied{fixVehicle()}
Occupied->Free{[Vehiculo.allInstances()->select(v|v.signalWaitingOperator=true)->size()=0 ]
finishRevision()}

Occupied->Occupied{[Vehiculo.allInstances()->select(v|v.signalWaitingOperator=true)->size(
)>0 ] finishRevision()}

  end
end
```

Diagrama de Secuencia para el Soil1:



reset
!new CRA('cra')

!new Operario('op1')
!new Operario('op2')
!new Dispositivo('d1')
!new Dispositivo('d2')

!insert (d1, op1) into DispositivoOperario
!insert (d2, op2) into DispositivoOperario

!insert (cra, d1) into Administracion
!insert (cra, d2) into Administracion

```
!new Noria('noria')
!noria.street:='Calle1'
!insert (cra, noria) into Monitoriza

!new Vehiculo('n1')
!n1.number:=1
!insert (noria, n1) into AtraccionVehiculo
!new Vehiculo('n2')
!n2.number:=2
!insert (noria, n2) into AtraccionVehiculo
!new Vehiculo('n3')
!n3.number:=3
!insert (noria, n3) into AtraccionVehiculo
!new Vehiculo('n4')
!n4.number:=4
!insert (noria, n4) into AtraccionVehiculo
!new Vehiculo('n5')
!n5.number:=5
!insert (noria, n5) into AtraccionVehiculo

-----------------------------------------------
!new Rusa('rusa')
!insert (cra, rusa) into Monitoriza

!new Vehiculo('r1')
!r1.number:=1
!insert (rusa, r1) into AtraccionVehiculo
!new Vehiculo('r2')
!r2.number:=2
!insert (rusa, r2) into AtraccionVehiculo
!new Vehiculo('r3')
!r3.number:=3
!insert (rusa, r3) into AtraccionVehiculo


-----------------------------------------------
--Parque de atracciones completo
--Empiezan las averias
-----------------------------------------------
!n1.errorDetected()
!n2.errorDetected()
!r1.errorDetected()

!d1.finishRevision()
!d2.finishRevision()
!d1.finishRevision()
```

!n1.errorDetected()
!n2.errorDetected()