# Model multi-agent del metro de Barcelona

Javi Ledesma

**Resum**—El transport és una operació clau en molts dominis d'aplicacions, des de sistemes de transport públic a les ciutats fins a la logística interna en magatzems i fàbriques. La complexitat d'aquests sistemes és molt elevada a causa del nombre de nodes a les xarxes de trànsit i del nombre de vehicles. També presenten diferències importants, com ara que el trànsit per carretera es mou en entorns molt poc estructurats mentre que els ferrocarrils restringeixen els moviments de vehicles i creen un entorn altament estructurat. Independentment d'això, comparteixen el problema de determinar quin vehicle ha de fer una determinada "ordre de transport", és a dir, d'anar d'un lloc a un altre. Un sistema comú per fer-ho és un sistema de diversos agents on cada agent representa una part interessada en el problema del transport. Per exemple, els agents taxi i els agents passatger són representants de vehicles i persones (o mercaderies), respectivament. Cada passatger emet una "petició de propostes" i tria la de cost més baix entre els taxis disponibles per assignar una ordre de transport.

**Paraules clau**— Agents BDI, sistema de comunicació ACL, simulació, metro, màquines d'estat

**Abstract**—Transportation is a key operation in many application domains, from public transportation systems in cities to in-house logistics in warehouses and factories. The complexity of these systems is very high due to the number of nodes in the traffic networks and the number of vehicles. They also present important differences, such as that road traffic moves in very unstructured environments while railways restrict vehicle movements and create a highly structured environment. Regardless, they share the problem of determining which vehicle should make a certain "transport order", that is, going from one place to another. A common system for doing this is a multi-agent system where each agent represents a stakeholder in the transportation problem. For example, taxi agents and passenger agents are representatives of vehicles and people (or goods), respectively. Each passenger issues a "request for proposals" and chooses the lowest cost among the available taxis to assign a transport order.

**Index Terms**— BDI agents, ACL communication system, simulation, metro, state machines

## 1 Introducció - Context del treball

En la gestió dels sistemes de metro es necessita un software per poder analitzar i estudiar la quantitat necessària de metros per cadascuna de les línies[5]. A partir de les dades extretes de les entrades horàries de passatgers a les diferents estacions es pot planificar i reestructurar el transport davant de noves condicions socials, com ara la pandèmia actual. El propòsit final d'aquests sistemes és el de poder automatitzar l'anàlisi de la concurrència de cada línia i actuar segons els resultats proporcionant i fomentant una mobilitat més sostenible per apropar-nos a la creació de les smart cities.

Degut a la complexitat del treball, hem creat un entorn de simulació basat en Lua [6] a partir d'un sistema multi agent (MAS) on diversos agents representaran el comportament dels metros recorrent el mapa de metro de Barcelona. Per dur a terme la simulació hem desenvolupat un sistema complex amb programes orientats a agents belief-desire-intention (BDI). Aquesta programació ens permet crear un entorn on cada agent actuarà de forma autònoma i intel·ligent percebent el que té al voltant i actuant davant dels conflictes que hi trobi dins l'entorn.


- *E-mail de contacte: javi.ledesma@e-campus.uab.cat*
- *Menció realitzada: Enginyeria de Computadors*
- *Treball tutoritzat per: Lluís Ribas-Xirgo (Departament de Microelectrònica i sistemes)*
- *Curs 2020/21*


A partir de màquines d'estat controlarem el comportament dels agents que fan referència als metros de les diferents línies de la ciutat comtal. S'ha partit d'un model base on només comptarem amb les línies L6, que va de Plaça Catalunya a Reina Elisenda i L7, que també inicia el recorregut a Plaça Catalunya i el finalitza a Avinguda Tibidabo. A partir de la generació de mapa (secció 3) podríem generar tot el mapa de Barcelona.

Pel que fa a la comunicació entre els diferents agents utilitzarem una llibreria de Lua ja implementada per a les funcions d'Agent Communication Language (ACL). Serà imprescindible controlar la comunicació a l'hora de prendre decisions per tenir una millor qualitat de servei i evitar possibles errors de sincronització entre els diferents agents.

En aquest treball farem referències a màquines d'estat dels tipus EFSM i EFS²M, i un sistema bàsic que representarà les línies de metro que podria millorar-se a partir de dades reals. D'aquesta manera, poder valorar la quantitat de metros necessaris en cadascuna de les línies existents a Barcelona o altra ciutat que compti amb un metro. Adaptarem un model de flota de taxis que funciona a partir d'un model de subhasta [1, 2] de les propostes dels passatgers utilitzant el mapa de metro actual i modificant la interacció dels agents amb l'entorn.

## 2 Objectius

L'objectiu principal consisteix en obtenir un model de totes línies de metro que hi ha en l'actualitat a Barcelona.

És necessari tenir a l'abast dades reals referents a la quantitat de gent que accedeix a una estació, determinant quants metros han d'estar operatius, tant a les hores puntes on es necessitarà una major freqüència de metros, com a les restants durant cada dia de la setmana.

Cadascuna de les diferents línies necessitarà una densitat concreta d'agents de metro per línia. Per tant, serà necessari fer un estudi individual de cadascuna d'aquestes. Donada la complexitat del treball s'ha partit d'un model base amb les línies de metro L6 i L7 de Barcelona i una simulació a partir d'un sistema basat en agents recorrent el mapa. La implementació d'aquests sistemes és molt útil en simulacions de qualsevol dels transports públics a les ciutats fins a la logística interna en magatzems i fàbriques.

A partir de la generació del mapa actual de metro de l'àrea metropolitana de Barcelona crearem dos tipus d'agents: els conductors i els metros. Per facilitar la interacció entre els diferents agents dins l'escenari de simulació adaptarem aquest comportament d'un sistema MAS de flota de taxis.

Cada agent actuarà de forma autònoma i prendrà les decisions que cregui convenient segons l'entorn i a partir de la comunicació amb els altres agents. Per programar el comportament dels agents utilitzarem màquines d'estat finites esteses (EFSM) per tractar la part reactiva i màquines de pila d'estats finits estesa (EFS²M) per tractar la part deliberativa a partir del model de software d'agents Belief–desire–intention (BDI).

## 3 GENERACIÓ DEL MAPA DE METRO

El dia 30 de desembre de 1924 es va inaugurar el primer tram de metro a Barcelona que unia Lesseps amb plaça Catalunya[3]. Els enginyers Octavio Zaragoza i Pablo Muller van ser els pares d'aquest gran projecte i van assistir al primer de molts viatges que realitzaria a partir d'ara aquest nou mitjà de transport. Més de noranta anys després d'aquest succés hem vist un augment tant en la quantitat de línies com d'estacions.

Per poder realitzar la simulació i comprovar el comportament dels diferents agents crearem un model base a partir de les línies de metro L6, que va de Plaça Catalunya a Reina Elisenda i L7, que també inicia el recorregut a Plaça Catalunya i el finalitza a Avinguda Tibidabo.

Cal tenir en compte que els mapes de metro són simbòlics, és a dir, que no es representen les vies d'acord amb la seva disposició geogràfica exacta, per facilitar la lectura del mapa.

La generació del mapa de metro Barcelona ha requerit la creació d'un software per poder detectar cadascuna de les estacions de metro i connectar-les a partir de caràcters que representaran la direcció de les vies. A continuació, s'explicarà més en detall cadascun dels passos realitzats.

Abans de començar amb aquest primer pas, haurem de realitzar una cerca on escollirem una imatge de les línies de metro de Barcelona que sigui de l'actualitat i es puguin visualitzar correctament les línies amb les estacions operatives. Hi ha plànols que ens mostren línies no operatives. Aquests s'utilitzen per informar sobre futures línies o estacions dins de l'àrea metropolitana.

Inicialment, es va voler utilitzar un software que transformés els píxels que no fossin blancs, és a dir, aquells píxels que fossin color en píxels negres. El motiu pel qual es va decidir fer-ho d'una altra manera va ser per l'alta definició de les imatges. Les imatges cada cop són representades amb un nombre major de píxels, a major quantitat de píxels més ombres es formen. Les ombres es creen quan tens una línia i augmentes el zoom per poder arribar a veure-la píxel a píxel. A més, els plànols del metro contenen molts caràcters (nom d'estacions i identificador de sector), per tant, es va decidir que no era una opció gaire viable. La solució utilitzada es mostra en les figures 1 i 2, aquesta va consistir en fer servir l'aplicació de Microsoft Office Power Point enganxar a una diapositiva la imatge del plànol de metro (figura 1 part esquerra) i resseguir les línies L6 i L7. Un cop resseguides eliminar la imatge de fons, d'aquesta forma teníem les línies fetes i ens estalviàvem haver d'esborrar els caràcters (figura 1 part dreta).

*Figura 1: Passos inicials de la generació del mapa*

Un cop tenim les línies fetes es necessita un programa on puguis reduir la resolució de la imatge (Photoshop, Paint...) . D'altra banda, si a l'hora de fer la reducció comprovem que hi ha píxels que tenen un color diferent de negre o blanc, aleshores haurem de pintar aquells píxels de manera manual. L'últim pas dins del programa

d'edició serà el de marca píxels de color vermell les zones on volem assignar les estacions (figura 2 part esquerra), recordant que seran parelles d'estacions, és a dir, en una via tindrem l'anada (per exemple: estació de Muntaner direcció Reina Elisenda) i a l'altra la tornada (per exemple: estació de Muntaner direcció Plaça Catalunya).

*Figura 2: Passos finals de la generació del mapa*

El programa que s'encarregarà de transformar el dibuix que tenim actualment a una matriu de caràcters. Primer carregarà la imatge del pas anterior, a continuació comprovarà píxel a píxel els seus valors RGB. Per reduir la possibilitat d'errors quan un píxel tingui valors superiors a 220;220;220 interpretarem que serà un píxel blanc i per tant, guardarem la posició dins d'una matriu assignant el valor # (paret). Si el valor RGB del píxel és inferior a 30;30;30, aleshores interpretarem que serà un píxel de color negre i el guardarem dins de la matriu assignant el valor A (via de metro). Finalment, en el cas que el píxel tingui un valor intermedi com per exemple 128;200;150 interpretarem que és un píxel vermell i el guardarem dins de la matriu assignant el valor @ (estació de metro). Un cop recorreguts tots els píxels de la imatge, l'últim pas serà el de crear un fitxer on escriurem tots els valors emmagatzemats dins de la matriu.

Un cop extraiem el fitxer de caràcters, modificarem el sentit de les vies, és a dir, canviarem els caràcters segons el sentit en què voldrem que els agents dels trens es desplacin mentre es dur a terme la simulació (figura 2 part dreta). Els comportaments dels diversos agents que hem creat es citen en el següent apartat.

A més de modificar els caràcters on obligarem posteriorment a què els trens segueixin aquestes línies de direcció, a continuació, dins del mateix fitxer del mapa haurem d'assignar els noms de les parades d'origen i el seu destí. Aquestes dades hauran de ser representades de la següent forma: @estació_origen estació_destí. Per facilitar la lectura del programa, utilitzarem al principi de cada línia de text el símbol '@', l'acompanyarem del nom de l'estació actual o estació origen, el concatenarem utilitzant el símbol ' ' i finalment, escriurem el nom de l'estació següent o estació destí.

## 4 SIMULACIÓ

Un cop realitzat el mapa i carregat a memòria (creant les estructures lògiques corresponents) podrem iniciar la simulació. Com hem dit abans aquest és un projecte MAS, on els metros apareixeran en les posicions de la matriu del mapa i recorreran la línia estació a estació. A continuació parlarem de totes les llibreries utilitzades per fer la comunicació entre agents i els comportaments d'aquests.

### 4.1 Llibreries utilitzades en la comunicació entre agents

Per poder realitzar la comunicació entre agents i i la gestió d'aquests, s'han utilitzat dos llibreries ja implementades en el model de sistema multiagent de flotes de taxis de Ll. Ribas-Xirgo [1].

#### 4.1.1 Agent Communication Language (ACL)

Quan volem realitzar una comunicació entre diferents agents existeixen protocols per fer l'intercanvi d'informació entre agents [7]. La forma més simple és la d'utilitzar dos agents, un actuarà com a client i enviarà una consulta a un altre agent que tindrà la funció d'actuar com a servidor i esperar la resposta. En el cas d'aquest projecte, fem ús d'una llibreria ACL on els agents de conductor (secció 4.2.1) es comunicaran amb la resta d'agents de metro (secció 4.2.2) de la simulació. D'altra banda, també existirà una comunicació entre cada metro i la resta dels agents de metro que explicarem més endavant. En la figura 3 es mostra com es comuniquen els agents dels conductors i els metros.

A partir d'aquesta comunicació es vol acabar tenint una relació d'un agent de metro amb un agent de conductor. Mentre es dur a terme el lligam entre els dos agents, es tancaran les comunicacions de cada conductor amb els demés metros i de cada metro amb la resta de conductors.

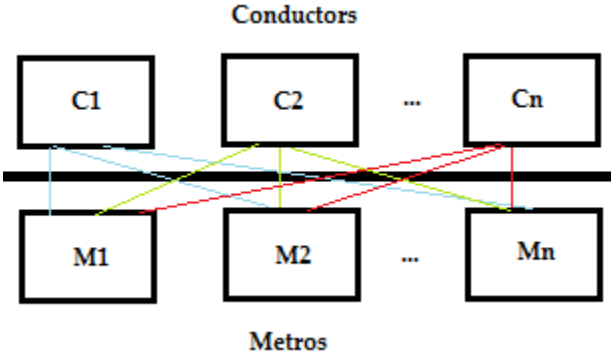


*Figura 3. Comunicació entre agents*

En la comunicació entre clients no s'acostuma a realitzar peticions o respondre peticions amb un missatge complet, sinó que s'utilitza un identificador que permeti al client entendre el que li ha enviat el servidor. Un

exemple seria quan un agent de conductor s'ha creat envia un missatge de "call for proposals" per interactuar amb els agents de metro. A continuació els cada agent de metro escoltarà les peticions dels conductors, s'avaluarà la situació de cada metro i el que tingui un cost de desplaçament major acceptarà la proposta feta pel conductor i es convertirà en la proposta guanyadora, mentre que la resta d'agents de metro declinaran la proposta i esperaran la d'un altre agent conductor. El cas d'aquest exemple de comunicació entre agents necessita el suport de la llibreria BDI-agent que explicarem en el següent apartat.

Comunicar-se entre agents resultar complicat a l'hora d'intentar obtenir de forma ràpida respostes. Tant els agents de conductor com els de metro tenen un comportament definit a partir de màquines d'estat (secció 4.2), això significa que en moltes ocasions haurem d'esperar que els dos agents estiguin en l'estat corresponent per poder realitzar una unió a partir de la comunicació. Les respostes de cada agent són asíncrones i arriben en intervals irregulars, per tant el client no sap quan arribarà cada missatge de resposta i pot estar ocupat executant altra tasca quan arribi la resposta del servidor.

#### 4.1.2 Agents belief-desire-intention (BDI)

Necessitem l'ajut d'aquesta classe ja implementada per a executar tasques de control i gestió d'alt nivell en entorns dinàmics complexos on tenim agents amb estats deliberatius. L'arquitectura dels agents BDI[4] conté les beliefs que es poden interpretar com el component informatiu de l'estat del sistema. Com estem en un entorn canviant l'agent no pot assegurar que realitzarà una acció en un element que pot canviar en el futur. Cada agent comptarà amb una llista de beliefs. A continuació tenim els desires que representen l'estat motivacional de l'agent, és a dir, els objectius o situacions que aquest vol aconseguir o provocar (per exemple: l'objectiu d'un conductor per trobar l'agent de metro més proper per sol·licitar que el reculli i el porti al seu destí). Finalment, les intentions representen l'estat deliberatiu de l'agent, és a dir, dur a terme allò que ha escollit fer.

Les aplicacions en temps real com seria aquesta, tenen una sèrie de característiques que fan que sigui de gran utilitat l'ús d'aquests agents. En primer lloc, ens podem trobar amb un entorn que es va modificant; per exemple la creació o eliminació d'un agent com ara el conductor o en el cas de tenir un agent passatger, aquests constantment apareixerien en les estacions. Una altra característica és la d'executar accions o procediments que alterin el comportament de l'agent; per exemple quan un metro arribi a una estació farà una parada en la "gate" de l'estació i després continuarà el rumb fins a la següent estació. Les accions i procediments es realitzaran a partir de l'entorn independentment de l'estat intern del sistema. L'entorn només es podrà detectar de forma local, per aquest motiu l'agent de conductor comptarà amb una variable que contingui el mapa.

Com hem mencionat anteriorment, disposarem de dos tipus d'agents: els conductors i els metros. En el següent apartat veurem màquines de pila d'estats finits estesa (EFS²M) que representen la part deliberativa. Són màquines d'estat que necessiten agents creats a partir de l'arquitectura d'agents belief-desire-intention.

### 4.2 Elements de la simulació

#### 4.2.1 Inicialització dels agents i l'entorn

A l'hora d'iniciar la simulació carregarem el fitxer que conté el mapa que els agents de metro recorreran. Com cada agent actuarà de forma autònoma, necessitem un sistema en el qual tots els agents es puguin sincronitzar provocant el menor nombre de conflictes possibles. Els metros es mouran gràcies al model de subhastes realitzat a partir de les peticions dels agents conductors.

Quan s'inicia l'execució del programa el primer que veiem no és el mapa amb els agents, és necessari especificar el valor d'algunes variables prèviament.

En primer lloc, s'ha d'especificar la quantitat de cada agent per poder-los crear, és a dir, es pregunta sobre la quantitat de metros que hi haurà durant la simulació i, a continuació haurem de decidir quants agents conductors crearem. Com cada tren necessita un conductor per poder desplaçar-se, aleshores en crearem tants com metros de forma simultània hi hagi en la simulació. Per tant, com sabem que el nombre de metros és equivalent al nombre de conductors només es preguntarà un cop.

En segon lloc, decidirem la freqüència en la qual s'actualitzarà el mapa de la simulació. Si utilitzem una freqüència d'un tick, a l'hora de visualitzar la simulació podrem observar com els metros es desplacen parcel·la a parcel·la i com s'aturaran a cadascuna de les estacions.

Finalment, comptem amb una variable que permetrà en un futur poder realitzar múltiples simulacions. Aquesta variable consisteix a definir el nombre màxim d'arrencades del programa. En cada iteració s'executarà una simulació del programa, com els agents apareixen de forma aleatòria pel mapa, es pot intentar fer proves de forma automàtica per trobar possibles errors en l'entorn, a més d'utilitzar la funcionalitat de poder fer una anàlisi dels metros necessaris per tenir la màxima concurrència sense que hi hagi problema entre els diferents agents.

#### 4.2.2 Els agents i el seu comportament

El comportament dels agents és causat per una comunicació entre agents de tipus conductor i agents de tipus metro. Aquesta comunicació segueix un protocol de xarxa de contracte. A més, de forma interna cada agent compta amb una comunicació interna entre la capa deliberativa i la capa reactiva que seran explicades seguidament.

##### 4.2.2.1 Agent conductor

Els agents dels conductor seran creats arreu del mapa. Per donar-li realisme a la simulació, un conductor només podrà ser creat en una casella de porta (representació dins el mapa: '*'), és a dir, aquelles caselles que estan davant

d'una estació (representació dins el mapa: '@').

El conductor s'encarregarà d'avisar a tots els agents de metro, sol·licitant els serveis d'aquell que estigui més a prop. Per dur a terme aquest procés es necessita tenir una bona comunicació i gestió de control dels dos tipus d'agents (secció 4.1). La representació del comportament de l'agent conductor es pot veure en les dos figures que tenim a continuació.

El comportament de la part deliberativa d'aquest agent es veu en la figura 4. La representació es fa a partir d'una màquina de pila d'estats finits estesa (EFS²M). Aquest tipus de màquines d'estat utilitza l'arquitectura de belief-desire-intention (BDI) que hem explicat anteriorment (secció 4.1.2).

Aquest tipus d'agent es transformarà en un "agent d'autoritat de ruta", és a dir, el conductor serà el responsable que un tren es desplaci o no. Inicialment, com es pot veure en la següent imatge (Figura 4.), com estarà esperant en una porta d'una estació el destí escollit per aquest serà la següent estació.

A continuació, entrarem en el primer estat que té el nom de find_a_taxi, recordem que els diferents estats estan heretats del programa de flota de taxis [1]. Aquest primer estat indica que volem demanar un taxi i per realitzar la cerca guardarem a la llista d'intencions els passos per trobar un taxi mitjançant subhasta. Es farà una planificació de la cerca de metro a partir de tres intencions que seran emmagatzemades en l'ordre invers, és a dir, de l'última a la primera. Això és degut al fet que utilitzarem una estructura de pila, per tant, per buidar la pila ho farem des de la primera intenció inserida i no pas la darrera.

La primera intenció que s'executarà serà la de send_CfP_to_taxis, l'objectiu d'aquesta és envia una proposta o petició a tots els agents de metro. Primer es comprova que hi ha metros i enviem a l'agent metro la nostra petició de destinació. A continuació, s'executarà la segona intenció que té el nom de collect_proposals en la qual s'envia un missatge que estàs recopilant les propostes a tots als agents de metro i busques un metro que estigui lliure i pugui acceptar la proposta de recollida. Els noms de metros que responguin de forma positiva es guardaran en la llista de beliefs. Finalment, esperarem que hi ha moviment de taxis, si no es detecta aquest moviment durant 3 ticks, llavors s'informa que estem en estat de deadlock i es tornarà a l'estat de find_a_taxi. L'última intenció correspon a evaluate_proposals_and_send_replies on s'avaluen les propostes emmagatzemades com beliefs i ens quedem amb la millor, és a dir, aquell metro que està més a prop i per tant, a priori arribaria més aviat. Si tot va bé entrarem en l'estat waiting_for_taxi i esperarem l'arribada del taxi. En cas contrari, si no hi ha cap proposta tornarem a llançar la intention de cercar taxis (find_a_taxi).

Mentre estem esperant a l'arribada del taxi, poden sorgir imprevistos, estarem pendents de la comunicació amb el metro que ha sortit escollit com a guanyador entre totes les propostes de metros lliures. Si el metro trobés algun impediment li enviaria al conductor un missatge de fallada i tornarem a fer la crida per trobar un nou metro lliure.

Finalment, si tot va bé entrarem en l'estat de waiting_for_arrival. En aquest estat esperarem que el metro arribi a la porta de l'estació de destí. En arribar, sortirem del metro i en el següent tick, tornarem a entrar-hi per anar a la següent estació i així simular la conducció per la línia de metro corresponent.

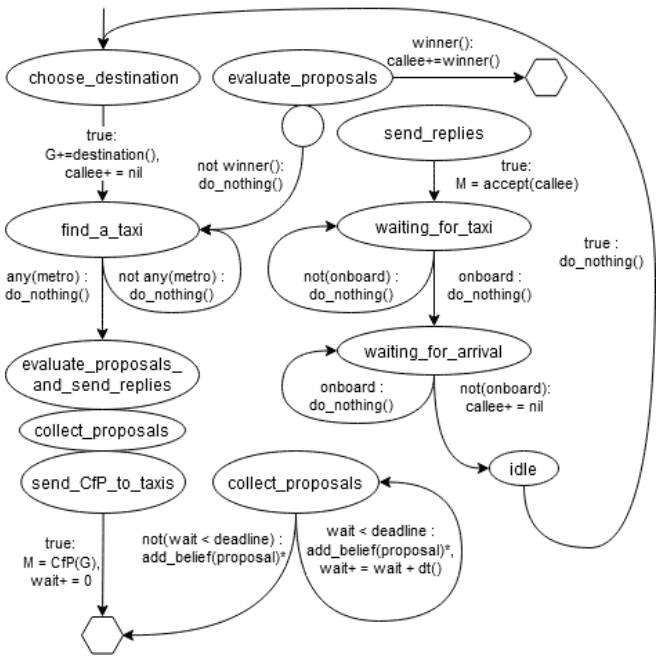


*Figura 4. Comportament deliberatiu de l'agent conductor representat a partir d'una EFS²M*

De forma simultània, l'agent de conductor també haurà de controlar el comportament de la part reactiva que es veu en la figura 5. La part reactiva compta amb quatre estats: "CALL", "WAIT", "EN ROUTE" i "ARRIVED".

Inicialment, l'agent de conductor estarà en l'estat de "CALL". En aquest estat serà on realitzarem la comunicació i esperarem a tenir una proposta de metro guanyadora. Si estem en l'estat "CALL" i esperant un taxi, llavors passarem a l'estat "WAIT". Ens mantindrem quiets fins que arribi el taxi a la casella on està el conductor esperant, és a dir, la porta de l'estació d'origen. Un cop el metro arribi ens posem en l'estat "EN ROUTE". En aquest estat es comprova a cada tick si hem arribat a la destinació, actualment el conductor no està visible al mapa, consta com a conductor a bord d'un metro. Finalment, quan arribem a la destinació sortirem de metro, es podrà tornar a veure l'agent conductor en el mapa i deixarà de comptar-se com a conductor a bord d'un metro i canviarem l'estat a "Arrived". En aquest estat el conductor es tornarà a visualitzar en el mapa.

Com el conductor ha de seguir recorrent el mapa, un cop arribi a la porta de l'estació de destí, aquesta passarà a ser la porta de l'estació origen i l'agent conductor sol·licitarà al metro que estigui més a prop anar a la següent estació. Donat que el metro més proper serà justament en el que havíem viatjat, aquest agent de metro s'encarregarà de portar-nos al següent destí, així de forma successiva.

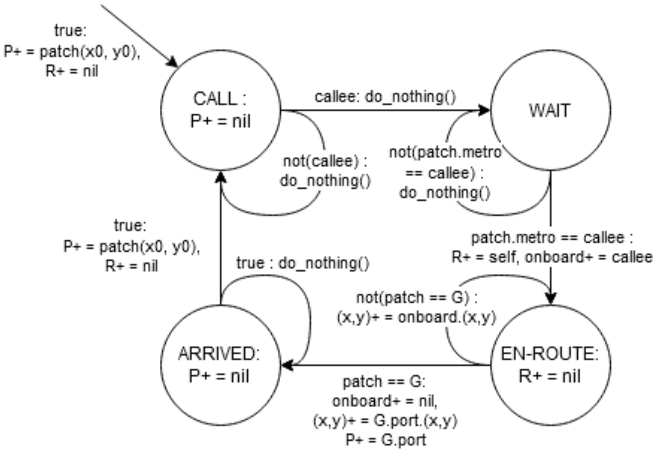


*Figura 5. Comportament reactiu de l'agent conductor representat a partir d'una EFSM*

#### 4.2.2.2 Agent metro

A diferència dels conductors, els agents de metro poden ser creats en qualsevol part del mapa que sigui transitable, és a dir, mentre la casella sigui diferent del tipus port (representació dins el mapa: '@') o paret (representació dins el mapa: "#").

L'agent de metro s'encarrega d'escoltar i respondre a les peticions que els agents conductors envien. Només es desplaçaran per recollir un conductor o portar-lo al seu destí. En els altres casos l'agent estarà quiet esperant que un conductor pregunti per algun metro lliure. Cada metro anirà del principi al final de la línia i a la inversa. Els metros avançaran pel mapa (matriu mapa) a cada "tick" seguint la via generada (secció 3). Per tant, un metro podrà avançar de manera horitzontal, vertical o en diagonal.

Cal tenir en compte que cada cel·la del mapa es pot fer equivaldre a una determinada distància o millor, a un temps de recorregut o de temps de permanència del tren en el tram. Tenint en compte que el mapa és simbòlic, la longitud visible de cada segment al mapa no té per què coincidir amb la distància física que ha de recórrer el tren. Això es pot solucionar afegint a cada tram un paràmetre que indiqui el temps (ticks) de permanència del tren en el tram corresponent. Per simplificació, no ho tindrem en compte.

Aquest agent també disposa d'una EFS²M que representa el comportament de la part deliberativa i sobre una reactiva representada a partir d'una EFSM. A continuació, tenim les imatges dels dos tipus de màquines d'estat esmentats. En el cas de l'EFS²M, de la mateixa forma que en la part del conductor, també es fa ús de l'arquitectura d'agents BDI.

Com es pot veure en la figura 6, quan un agent de metro es crea el primer estat en el qual entra per defecte és el de look_for_passenger. Aquest consisteix en escolta propostes de conductors. El nom passenger era el de l'agent utilitzat en el programa de flota de taxis [1]. Aquí s'ha adaptat i es comporta com un driver (conductor). Ens mantindrem dins l'estat de look_for_passenger si cap agent conductor sol·licita un tren. Si hi ha conductors que necessiten un metro, aleshores en comunicarem amb l'agent conductor indicant-li la distancia entre el metro i el conductor. Si rebem una resposta en la qual un conductor determina que l'agent metro és l'elegit, aleshores passarem per diversos estats que representaran la planificació d'intentions de l'agent metro.

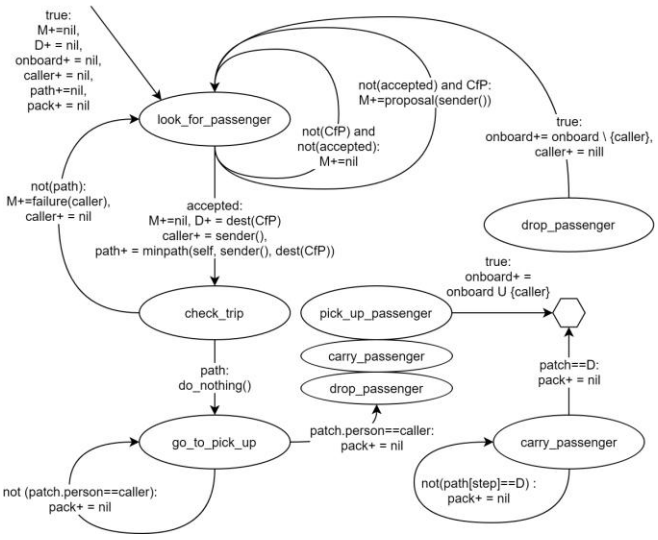


*Figura 6. Comportament deliberatiu de l'agent metro representat a partir d'una EFS²M*

Un cop el conductor hagi demanat que l'agent metro el reculli, el metro canviarà el seu estat a check_trip. Prèviament el metro crearà una llista d'intencions que representen la planificació dels diferents estats en els quals el metro haurà de passar per poder satisfer la petició del conductor. En l'estat de check_trip, si el taxi no detecta una ruta esborrem totes les intencions que puguem tenir, tornant així a un estat inicial en què esperarem al fet que un conductor sol·liciti un metro. A continuació entrarem en l'estat següent que té el nom de go_to_pick_up. En aquest estat s'informarà als altres agents que estarem ocupats i no acceptarem noves ordres, perquè estarem dirigint-nos cap a la parcel·la on hi ha el conductor a recollir. Un cop arribem a la casella on està el conductor esperant, canviarem al següent estat de la pila. El nom d'aquest estat és pick_up_passenger i té l'única funció d'augmentar el comptador de conductor a bord. A continuació, passarem a l'estat de carry_passenger_to_dest. En aquest estat tornarem a informar als altres agents que estarem ocupats i no acceptarem noves ordres, perquè estarem dirigint-nos cap al destí que ens va exigir el conductor. Un cop arribem a la casella de destí on el conduc-

tor sol·licitava anar, passarem a l'últim estat drop_passenger. En aquest estat reduirem el número de conductor a bord i l'agent metro esperarà que un conductor torni a demanar per ells.

De forma simultània, l'agent de metro també haurà de controlar el comportament de la part reactiva que es veu en la figura 7. La part reactiva compta amb cinc estats: "WAIT", "ON-WAY", "PICK-UP", "CARRY" i "DROP-OFF".

Inicialment, l'agent de metro estarà en l'estat de "WAIT" esperant a rebre una intention d'anar a buscar a un conductor. Quan passi això comprovarem la millor ruta (menor distància) per arribar fins on hi ha el conductor i actualitzarem el nostre estat a "ON-WAY". L'agent metro s'anirà movent pel mapa, sempre buscant la millor ruta, és a dir, constantment s'anirà recalculant quina és la millor opció per arribar abans on és el conductor. S'ha de dir que inicialment, com només hi ha un sol camí per anar d'estació origen o estació destí no variarà la ruta en cada tick, però si contempléssim la possibilitat que un tren es pogués avariar en la via o bé, un tram estigués en obres, generant un mapa amb una via alternativa, l'agent de metro podria escollir aquesta opció alternativa. En la fase "PICK-UP" calcularem el cost per arribar a la porta destí que ens ha comunicat anteriorment el conductor. Si podem arribar al port de destí significarà que hi haurà una ruta que podrem realitzar en l'estat següent. Estarem en l'estat "CARRY" fins a arribar a la destinació on passarem a l'estat "DROP-OFF" en què el conductor deixarà el metro. Arribats a aquest punt, tornaríem a començar des de l'estat de "WAIT" escoltant possibles peticions de conductors.

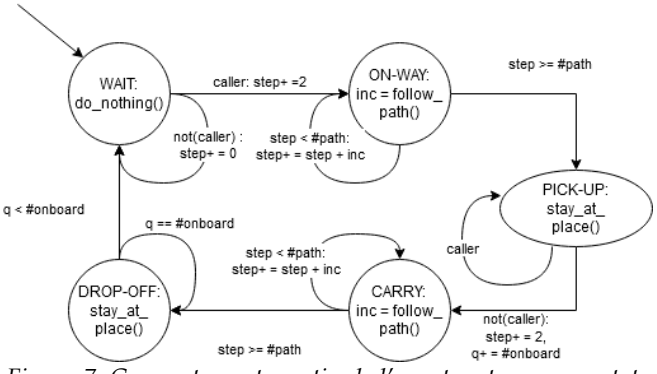


*Figura 7. Comportament reactiu de l'agent metro representat a partir d'una EFSM*

## 5 Conclusions

Com hem mencionat anteriorment, s'ha aconseguit fer una simulació en la qual es poguessin crear diversos agents de metro i agents de conductor. El model de subhastes ens ha permès poder emparellar cada metro amb un conductor.

S'ha fet un model de les línies L6 i L7 del mapa de metro de l'àrea metropolitana de Barcelona. Comprovant que tenir uns resultats satisfactoris on els agents de conductors i metros són capaços de sincronitzar-se correctament, podem afirmar que seguint els passos de la secció 3 d'aquest article es podria fer la representació completa del mapa de metro de Barcelona, o bé, de qualsevol mapa de metro, trens, ferrocarrils o tramvies.

En la secció 4 s'expliquen tots els elements necessaris per a la simulació. He après com funciona la llibreria ACL que funciona a partir de protocol de xarxa de contracte. Abans de fer aquest treball, només havia utilitzat sockets com a elements de comunicació.

He reforçat els coneixements sobre màquines d'estat. Per simular el comportament de cada agent s'ha dissenyat una EFSM per tractar la part reactiva i descobrint les EFS²M per la part deliberativa.

No s'han pogut extreure una anàlisi de les dades per falta de temps. Però donat que els agents de metro es mouen a partir de calcular el cost menor, utilitzant l'algoritme de Dijkstra, es poden formar retencions, ja que no hi ha rutes alternatives, és a dir, tenim una via d'anada i una de tornada. Per tant, si volem millorar el servei de metro arribarà el moment en què no serà eficient afegir més metros a la línia, és a dir, haurem trobat un coll d'ampolla en la simulació. D'altra banda, podem deduir que a mesura que es vagi augmentant la quantitat de metros en una línia de metro, la freqüència de què passi un metro per certa estació serà major.

### 5.1 Línies de futur

Aquest treball encara està en la seva fase inicial, però és un projecte que pot arribar a tenir objectius molt ambiciosos. La finalitat d'aquest projecte seria crear un model multi-agent on afegiríem un tercer agent, els passatgers. Seria molt útil poder realitzar moltes simulacions i retornar resultats de l'ocupació dels trens a partir de dades reals. Com hem esmentat al principi de l'article aquest seria un gran pas per a fomentar el transport públic i tenir una mobilitat sostenible que ens apropi a la creació de smart cities.

## Agraïments

Voldria agrair l'esforç i temps dedicat per part del meu tutor Lluís Ribas-Xirgo. Degut a la pandèmia de la Covid-19, l'últim any entre les restriccions generals i les imposades per la universitat, la meva habitació ha sigut el meu espai de fer feina i d'oci. Com a conseqüència, no he pogut centrar-me al màxim en fer el TFG. En Lluís sempre m'ha donat suport, a més de deixar-me utilitzar el seu entorn per poder realitzar el projecte.

## Bibliografia

[1] Ll. Ribas-Xirgo. "Multi-agent System Model of Taxi Fleets." In L.M. Bergasa, M. Ocaña, R. Barea, E. López-Guillén and P. Revenga (eds.) *Advances in Physical Agents II. Proceedings of the 21st International Workshop of Physical Agents* (WAF 2020), November

19–20, 2020, Alcalá de Henares, Madrid, Spain. Published in *Advances in Intelligent Systems and Computing* (Springer), vol. 1285, pp. 123–134.

[2] Sandholm, T. "Limitations of the Vickrey Auction in computational multiagent systems". Proceedings of the Second International Conference on Multiagent Systems. In Proceedings of ICMAS-96 (1996). Washington University. Department of Computer Science. One Brookings Drive, Campus Box 1045 https://www.aaai.org/Papers/ICMAS/1996/ICMAS96-038.pdf

[3] Jesús Sancho. "Recorrido inédito por el primer metro de Barcelona" publicat a La Vanguardia el 10/01/2019. https://www.lavanguardia.com/local/barcelona/20190110/454039135388/recorrido-libro-primer-metro-barcelona.html

[4] Anand S. Rao and Michael P. Georgeff. "BDI Agents: From Theory to Practice". Proceedings of the First International Conference on Multiagent Systems. In Proceedings of ICMAS-95 (1995). Australian Artificial Intelligence Institute. Level 6, 171 La Trobe Street. Melbourne, Australia. https://www.aaai.org/Papers/ICMAS/1995/ICMAS95-042.pdf

[5] Irene Mariñas-Collado, Elisa Frutos Bernal, Maria Teresa Santos Martin, Angel Martín del Rey, Roberto Casado Vara and Ana Belen Gil-González. "A Mathematical Study of Barcelona Metro Network". Department of Statistics and Operations Research and Mathematics Didactics, University of Oviedo, 33007 Oviedo, Spain. BISITE Research Group, Department of Statistics, Applied Mathematics, Institute of Fundamental Physics and Mathematics, University of Salamanca, 37007 Salamanca, Spain https://www.mdpi.com/2079-9292/10/5/557/htm

[6] Roberto Ierusalimschy, Luiz Henrique de Figueiredo and Waldemar Celes 10/10/2008 Lua.org, PUC-Rio. "Lua 5.1 Reference Manual" http://public.hajtmar.com/files/TeX/CTM+TE/LuaMan51andLpegSource/ctmandte2010.pdf

[7] Tim Finin and Richard Fritzson. "KQML as an Agent Communication Language". To appear in The Proceedings of the Third International Conference on Information and Knowledge Management (CIKM'94), ACM Press, November 1994. Computer Science Department in University of Maryland Baltimore County, USA. Supported by the Air Force Office of Scientific Research and Advanced Research Projects Agency. https://ebiquity.umbc.edu/_file_directory_/papers/318.pdf

## APÈNDIXS

### A1. EINES USADES

El simulador de metro de Barcelona s'ha realitzat utilitzant el programa zerobrane. Per dur a terme la programació s'ha fet ús del llenguatge de programació Lua.

D'altra banda, per fer la generació del mapa de les línies L6 i L7 de Barcelona s'han utilitzat els programes de Microsoft Power Point i Photoshop. A més, s'ha hagut de programar un software en Java que substituís els píxels d'una imatge per caràcters. El programa utilitzat per generar aquest software ha estat Netbeans.

Finalment, per poder representar gràficament totes les figures de màquines d'estat vistes en la secció 4 de l'article, s'ha fet ús de l'entorn online de draw.io.

### A2. SOFTWARE GENERAT

Tot el software desenvolupat al llarg d'aquest projecte està dins el dossier d'aquest. La part de la generació del mapa està dins de la carpeta "ImageToFile", mentre que el codi emprat per la simulació dels metros es pot localitzar accedint a la carpeta "Dev".

### A3. SOFTWARE DE SIMULACIÓ

El software de simulació funciona de forma automàtica, només fa falta indicar dos valors a l'inici de la simulació, ja explicats dins l'article. Ho podem veure en la figura 8.

```
Model multi-agent del metro de Barcelona
(C) 2020-2021 Lluís Ribas-Xirgo, UAB.
(C) 2021 Javi Ledesma Pardo, UAB.

Step (number of ticks, 0 = pause) = 1
Runs (default: 0, not used) = 1
```

*Figura 8. Iniciació de la simulació*

En les figures següents (figura 9 i figura 10) es pot veure com un tren (que inicialment està reprensentat amb el caràcter "t") segons si va a recollir a un conductor ("T"), o bé, ja ha recollit a un conductor i va fins a la següent estació ("P") la representació d'aquest pot variar.

```
   00000000011111111
   12345678901234567
01 #@###############
02 #C...############
03 #*...######@*P@##
04 #@#..#######..###
05 ##@**@######..###
```

*Figura 9. Figures del mapa (part 1)*

```
   00000000011111111
   12345678901234567
01 #@###############
02 #C...############
03 #*..T######@*C@##
04 #@#..#######..###
05 ##@**@######.T###
06 ###..#######..###
07 ###..######@**@##
08 ###..#######..###
09 ###..@######..###
10 ##@*.*######..###
11 ###....#####..###
12 ####....@##@**@##
13 #####...*###..###
14 #####@*...@#..###
15 #######...*#..###
16 #######@*.....###
17 #########.....###
18 ##########....###
19 ############..###
20 ###########@**@##
```

*Figura 10. Figures del mapa (part 2)*