Discrete events model for dual mode transport system simulation and evaluation

Sami Mahari
IMARA team, INRIA, France

Arnaud de La Fortelle *
Robotics Lab, Mines ParisTech, France

13 May 2011

ABSTRACT

The European project CATS — City Alternative Transport System — is developing and evaluating a new vehicle system using a single type of vehicle for two different usages: individual use or collective transport. Real experiments will necessarily take place with a limited number of vehicles and stations. Hence there is a need for evaluation using simulations. INRIA is developing a discrete events simulator for that purpose, based on a previous work done for collective taxis. We present in this paper the model we use for the CATS project. This model rely on an adapted events/decision graph that extends previous graphs. The new feature of this model is the way we deal with two modes that can be extended to many other modes. This work therefore shows on a concrete example a method to efficiently merge multiple modes into one model.
Keywords: *Modeling, discrete events, simulation, graph, multimodality.*

1 INTRODUCTION

The CATS project objective is the final development and experimentation of a new urban transport service based on a new generation vehicle. Its major innovation is the utilization of a single type of vehicle for two different usages: individual use or collective transport. This new transport service is aimed at filling the gap between public mass transport and private individual vehicles. It is based on two operating principles: the *self service* concept where small and clean urban vehicles are offered on a short term rental basis, and the *flexible shuttle service* where a variable length of vehicles convoy, driven by a professional driver, operates at fixed hours along a line on a permanent basis or on a case by case basis. Both these principles are integrated in a single service (composed of vehicles and stations) called Cristal.

*Corresponding author – Mines ParisTech, Robotics Lab, Mathématiques et Systèmes, 60 Bd St Michel 75272 Paris Cedex 06, France, E-Mail: arnaud.de_la_fortelle@mines-paristech.fr

The final aim of this new service is a more efficient mobility in cities through a more balanced use of small clean vehicles and mass transport. This inclusive new transport system is well adapted to the needs of people with reduced mobility, young passengers and tourists. Four Cristal vehicles and two stations will be made available by Lohr Industrie to the project for experiments. CATS will complement the design and manufacture of the Cristal vehicle via a detailed definition of its operating principles and by a design of its urban settings (stations, infrastructures,) in accordance with cities and citizens needs.

Since such a system has never been deployed before, CATS needs tools to help the evaluation of the operating principles. There are indeed known operating principles for the two kinds of services, self service and flexible shuttle service, but it is unclear how to switch from one to the other. How do you get the vehicles to build the shuttle when it is time to go for buses? You clearly need to stop the self service and wait customers to give back the vehicles. But how long are we ready to wait? When do we decide to stop the self service? How long is the transition? What happen if a customer drives a long time and gives back its vehicle too late?

This is a short list of the problems raised by CATS and the design of the operating principles. An even more crucial question is: how do you know you have considered all scenarios? For sure we do not know, but using a model helps a lot defining completeness. In our case we use a graph to model the events and decisions that define the dynamic of the system. Once the model is defined, we derive an implementation of it. The resulting simulator then allows extensive evaluation of various scenarios and to optimize decision algorithms. This is the scheme we follow within the CATS project. Implementation and evaluation by simulation will not be presented here but they follow the same path as what has been already done with collective taxis [1, 2, 3, 4, 5].

2 SYSTEM DESCRIPTION

The system described in this paper consists of several components.

1. Vehicles: The vehicles used in this system are compact electric vehicles called Cristal. They can be used in manual driving, in convoys (platooning or towing without contact) or remote controlled by a local operator.

2. Locomotives: A special type of vehicles used exclusively by the company's professional drivers. They will be used to redistribute empty vehicles between system's stations and also to form convoys for the flexible shuttle mode. This is made possible thanks to the platooning technology of the Cristal vehicles. When not used, the driver puts the locomotive in the company's depot.

3. Stations: They are also provided by the Cristal Project. They are located in several places in the city and form a whole network. The vehicles are able to recharge when they are idle at the station. Some of the self-service stations will be used also as shuttle stops in the flexible shuttle operating mode, routes will be fixed to those shuttles.

4. Operator Center: The fleet of vehicles is controlled and managed by the system's operation center. This center controls the state and the location of each vehicle and relocates them when necessary. Also, it is responsible for switching between the two operating modes in order to satisfy the demand.

5. Clients: End users of the intelligent transportation system proposed by CATS.

3 EVENTS GRAPHS

The CATS system as described above is a highly complex stochastic system for which it is almost impossible to write a mathematical model describing its behavior. Thus, a computer simulation model is built for the system. This queuing-based discrete event model will be used to evaluate operational issues such as the transition between the two modes, vehicle relocation and vehicle availability.

3.1 Discrete events model

There are two principals paradigms for the development of a discrete events model. The *event-scheduling scheme* focuses on the events that instantaneously transform a system's state and/or schedule future events. The *process-oriented scheme*, on the other hand, focuses on processes and entities that flow through the process and interact with resources [6, 7].

The model proposed in this paper is built using the event-scheduling paradigm thanks to the flexibility of the design it offers and to many other features that the process-oriented scheme is not able to handle. Thus, the system is represented as a chronological sequence of events of the form $\{\dots, s_i, e_i, s_{i+1}, e_{i+1}, \dots\}$ where s_i is the state of the system at the time t_i and e_i is a system event happening at the time t_i making changes to the system bringing it to state s_{i+1} and so on.
We have used a Process Flow Diagram to describe graphically our model (Fig. 1). Additional modeling formalisms can be used to describe the interactions between system's entities such as Petri Nets, Network Diagrams... [8].

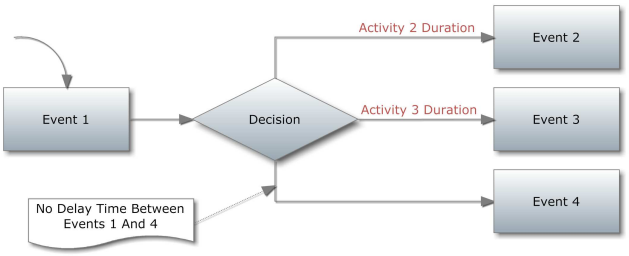

Fig. 1: Process Flow Diagram
Events are instantaneous. If an activity requires time to be performed, then the next event will be scheduled with a delay.

Our system is described based on simulation events. Each event cause the evolution of the system state. As it is described, the system can be divided into four subsystems that are loosely coupled hence they will be modeled separately. Events are associated to the four following sub-models:

1. Clients' behavior: events associated with clients;
2. Self-service operation mode: events associated with shared cars;
3. Flexible shuttle operation mode: events associated with shuttles;
4. Vehicle redistribution: events associated with locomotives.

An overview of these sub-models is given in figures 2, 3, 4 and 5.

3.2 User's events

The user is the center of our simulation model. It is the main force driving the system: a transportation need appears and the system reacts by adjusting to it, possibly simply by computing there is no good solution and doing nothing.

As we can see in the fig. 2, the user's behavior is the same for all transportation systems, so adding a new operation mode later (e.g. collective taxis or a Vélib system) will not cause the entire model to be

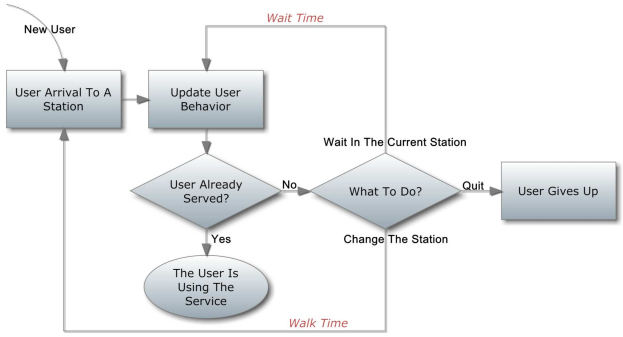

Fig. 2: User's Events
Being in a station and not served yet, the customer is free to decide whether he waits in the same station (in the waiting queue), walks to another station or simply quits the system.

altered and the integration of the new sub-model will be transparent for the user. The major assumption behind this model the fact that users try to optimize a mobility need (that could combine several criteria like time, comfort or cost). There are examples in the real life where this is not the case, e.g. for rendez-vous where synchronization between users appears.

The users part of the events graph begin with a client arrival. This is an event without predecessor so that we need an external model of the users arrivals like Origin-Destination matrices with intensities. This data is very important for the simulations but not for the simulator: we can begin with simplistic models (e.g. Poisson arrivals) and change the inputs without changing anything in our model.

Clients arrive at nodes of the system. First thing for the client is to update its behavior. It means the client get new information (e.g. whether there are available cars in sharing mode) and try to get served.

If he can, he enters a vehicle and the next events concern the vehicle. It will end up (for the client) with arrival to the destination station where the behavior will be again updated; this is indeed necessary when the transport mode is not door-to-door as for the shuttle (but this should happen for self-service only when there was no parking place at the destination station).

Otherwise, if the user is not served, he takes a decision: either waiting, going to another station or quit the system. Note that walking to another station is not necessarily a question of impatience since in shuttle mode there will be arrival at nodes where there is no service. Hence in shuttle mode the general pattern of service is arriving at a node A, walking to a node B where you embark for node C and finish walking to node D. When the user updates its behavior, he computes the best combination of modes to reach his destination (e.g. the optimized A-B-C-D path described above). Finally what we observe is that this model of users is well suited to multimodality even if this model is limited to two transportation modes: walk and cars.

3.3 Self service mode events

If you put aside the difficult relocation question, the shuttle mode should be extremely simple: a car embarks a user go to its destination where the user disembarks. Actually this is true only if parking is not a problem. This is why fig. 3 contains specific events for dealing with parking search. Note that our model is compatible with many types of parking searches (e.g. using centralized information or not); the events model simply states that sometimes you have to drive to another place whatever the way you chose it.

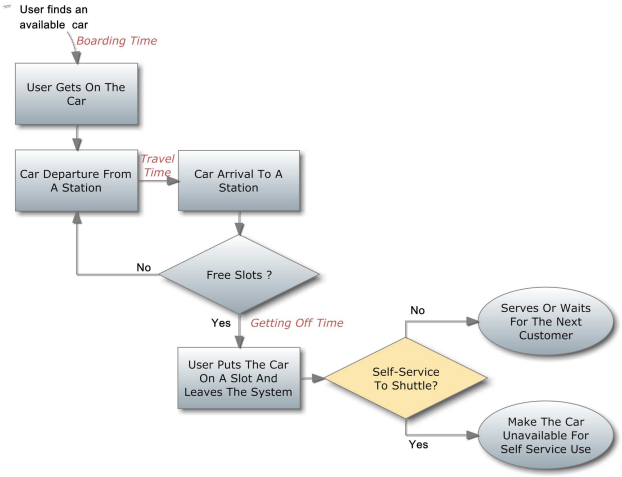

Fig. 3: Self-service operating principles.
When finishing their trips, the users look for an empty place where to put the car. They can be obliged to change the station if there are no free slots. Once an empty car is installed in a slot, the system decides whether it will remain available for self-service use or not.

After completion of the service there is a mode switch decision to take: keep the vehicle in self service or switch to shuttle mode. This decision is usually triggered by a central server (so that there would be a request) but other schemes are possible like a schedule. The event graph shows here the fine modeling we can naturally do. One would first think of a mode transition as an order sent to all cars at some time. However, cars already in use cannot change their mode until there are released. And several transitions may have occured when the car is again available. Again, with an event model, mode switch instants are naturally introduced, whatever complexity the transition has. Here we have only two modes for the vehicles but it is clear we could mix more.

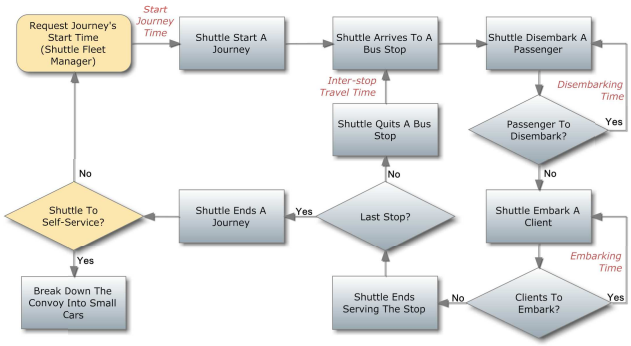

Fig. 4: Shuttle operating principles.
A shuttle's journey consists of serving a known number of stops of the shuttle's route at a given time. Once a transition to the self-service mode is launched, the system waits for the shuttle to end its journey and then breaks it down into individual cars.

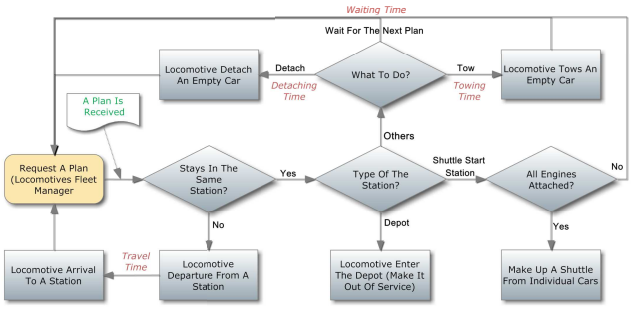

Fig. 5: Relocation process.
When a relocation of cars is needed, the system will give an execution plan for each locomotive. There are many possibilities offered to a locomotive (towing or detaching a car, waiting in the current station, going to the garage, form a shuttle)

3.4 Shuttle mode events

The shuttle mode of fig. 4 contains several loops: the main one concerns a terminal to terminal journey; inside there is a stop to stop loop; finally there are two loops for embarking and disembarking clients. Note that each client go again to the user's events sub-graph when completing its trip so that this graph is obviously linked to the other sub-graphs. We see also the mode switch event at the end of a service.

The "Break down the convoy into small cars" event is the beginning of a relocation process. Since the convoy considered as a single entity is re-labelled as platton (meaning a locomotive plus some cars to relocate), the first car now behaves as a locomotive and immediately requests a relocation plan to the fleet manager.

3.5 Relocation events

Relocation is probably one of the most challenging part in nowadays car sharing systems. It is critical for smooth operations and quality of service but is expensive so that it should be optimized. In this section we will agin not give any efficient algorithm but insist on the organization of the relocation by describing the atomic events contituting the relocation process. In fact, what we observe here from a more general point of view is a switch-over time for mode transition (relocate cars before or after shuttle services) as well as a standard relocation that occurs naturally during car sharing mode. It means the system is fundamentally the same during transient or recurrent periods; the only change is the intention of the system (ditribute the cars or build a shuttle).

The main fact behind the fig. 5 is the platooning technique. Cars are redistributed using a "locomotive" i.e. a car with a professional driver. Some cars can be towed away. This is what is described in the main loop of fig. 5. There are also special events that are necessary to built a shuttle at the right place so that one possible exit of this graph is the make up of a shuttle. We also consider the end of the service of a locomotive. The start of the locomotive service is the event "locomotive arrival at a station".

We see that the main loop amways go through the "request a plan" event. This occurs indeed after every simple operation (tow, detach, move). A relocation plan is at least the next simple relocation operation to do. But our intend is rather to use lists of operations so that locomotive have consistent plans without needing too many requests: in a discrete events model a plan request event can be to skip the request if the plan is fresh.

This request event is where global information is used. Since algorithms are undefined in our model, we do not assume a central server has all the information but it is *possible* in our model to implement such decision with full information.

3.6 Properties of the model

As we mentioned in the introduction, the main goal of the model is to ensure completeness of the system description. With a Process Flow Diagram, we simply need to check that all decisions or events generate all the desired events and that the diagram is closed. Then any scenario is generated by a path through the graph. Therefore, completeness is inherent to such a description. Adding new features means adding new events and links until the graph is again closed.

We, therefore, looked after other desirable properties. The main one is modularity. First, we can decompose the global system in subsystems in a rather natural way. Better, the model can combine many transportation sub-models. So, it can be extended to model a transportation system with more than the current two operating modes. Adding a new transporta-

tion mode is made simple by separating the sub-models and making them independent of each other. Furthermore, we can simulate multiple modes simultaneously and the customer will be free to choose between them.

The reason why we can operate such a separation lies in the structure of the system. The model's events occur only at stations (e.g. Client arrival, shuttle embarks a client...). The sub-models depend mainly on the station's state (queue of waiting clients, number of available vehicles ...) and that is crucial to combine them in a global diagram. A station is considered as a multimodal station where we can find more than one transportation service. This multimodality helps to mix different sub-models representing various transport modes.

Each station of the system is modeled as a queuing system where cars and shuttles are considered as *resources* and clients as the *processes* who can use an available resource (if there is) or wait for one to be freed[9]. The whole system is then seen as a queuing network where customers travel through the network and are served at the nodes. Using this *polling model* gives us access to many useful performance measures such as the average number in the queue, the average time spent in the queue and so on.

When processing an event, there can be a need to take some decisions in order to control the entire system. Decisions can be taken locally (e.g. user's decision to wait or not in the station) or globally (e.g. the plan of execution of a locomotive). Local decisions depend only on the entity's state and can differ from one entity to another. On the other side, global decisions need to have a global view of the system, and that's why they are given in a centralized way by the operator center. We will not go into details here, but with some care those global events do not disturb the separation of the modes and ensure the optimization at the system level.

4 CONCLUSION

The European CATS project needed a model to help evaluation by simulation. We built such a model but transition between two modes challenged us to produce a minimal model — in a sense that was not clear at the beginning — and we finally discovered a guiding separating principle, combining discrete events models and queuing network theory (polling systems). This approach seems fruitful. It is clearly not universal, but we believe it will help a lot in the modeling of more complex transportation systems, particularly in the current context of introduction of many new modes (e.g. Vélib, Autolib, collective taxis).

Current work is now to complete the implementation of such a model. For that we benefit of the experience of a first simulator for collective taxis [5]. Once the simulator is ready, we will perform numbers of simulations to evaluate the systems, its efficiency and its properties, and optimize decisions. We could even adapt some operating principles (e.g. instead of a single mode switching over time, modes could be mixed?) since we know the rules to follow to keep the good properties of our model. Our approach raised some interest in France, and further work could be extension of this simulator to other modes or to the transportation of goods.

Acknowledgment The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement 234341 (CATS project: `www.cats-project.org`).

REFERENCES

[1] Lioris, E., Cohen, G., and deLa Fortelle, A. September 20–21 2007 In International Colloquium on Taxis Lisbon, Portugal: . .

[2] Lioris, E., Cohen, G., and de laFortelle, A. 4–5 février 2009 In 12th IFAC Symposium on Control in Transportation Systems Versailles, France: . .

[3] Lioris, E., Cohen, G., and de laFortelle, A. 4–5 février 2009 In Congrès International ATEC-ITS Versailles, France: . .

[4] Lioris, J. E., Cohen, G., and deLa Fortelle, A. (2010) In 2010 Intelligent Vehicles Symposium : To appear.

[5] Lioris, J. E., Cohen, G., and deLa Fortelle, A. (2010) In SIMUL 2010, 2nd International Conference on Advances in System Simulations IARIA, International Academy, Research and Industry Association : Best paper award; to appear in IARIA journal.

[6] Cassandras Christos G., Lafortune, S. (2008) Introduction to discrete event systems, Springer, .

[7] Jerry Banks, John S. Carson II (Auteur), B. L. N. A. D. M. N. A. (2009) Discrete-Event System Simulation, Prentice Hall, .

[8] Lacy, L. W. Interchanging discrete event simulation process interaction models using the Web Ontology Language - owl PhD thesis College of Engineering and Computer Science at the University of Central Florida Orlando, Florida (2006).

[9] Law, A. M. (1999) Simulation Modeling and Analysis, McGraw-Hill Science, .