Bus on Demand

Filipe André Conceição Proença
Instituto Superior Técnico – Taguspark

Abstract
With the change in mobility patterns, operators of public transportation services need to upgrade their offer to meet these new requirements. It is in this context that is intended to study Bus on Demand services, services with flexible routes and schedules, where users define their origin/destination and a time window for the trip. This work aims to design and build a management solution for flexible services in real time and automatically, and that can create flexible travel solutions for their users. Another aim is to enable users of this service to make requests by various means of communication and receive feedback to the previously used mean of communication, warning them also when it is impossible to fulfill their requests.

Keywords: Bus on Demand, algorithm, itinerary, schedule, boarding place, landing place, time window, email, SMS.

1. Introduction
Nowadays, people have changed their mobility patterns from house – work – house movements to more irregular and complex ones. For this reason, public transport services operators have much more difficult in keeping up and satisfying their users’ necessities and requisites.

Other problem found by the public transport operators is that their fixed services don’t have the necessary demand, not only because of their itineraries and schedules but also because of the population density in their operation areas. In order to address their limitations, public service operators begin to diversify their offerings with more flexible services such as car-sharing [1] or bus on demand [2].

This work has as main objective to study bus on demand systems. Nowadays, bus on demand, also called dial-a-ride, systems use as principal mean to make transportation requests the telephone call, being necessary to have an operator in a call center to do the registration of the request in the system and to reply to the user if his request is fulfilled or not. The Internet has been another mean to make requests.

One of the problems detected on this study was that the existing systems do not use the paradigm of requests registration and itineraries and schedules calculation in real time, being necessary to make trips reservation requests with, at least, one day of advance. Other problem is due to the lack of
integration with the fixed service systems, which means that these two types of systems work in parallel and information is never crossed.

At last, but not least, there is the problem of the total lack of automation of the process, being necessary human hand to carry out all the stages of the process, either at the request registration as sometimes at the itineraries and schedules calculations.

2. State of the Art

2.1. Passengers’ Public Transportation Services

Passengers’ public transportation services are mostly done in networks with fixed itineraries, stops and schedules. This kind of transportation is usually denominated as regular service, since there is a set of vehicles engaged to a particular default route, with a time gap between the passages of those vehicles in some stopping points. However, this kind of service is beginning to be financially unprofitable in rural places or places with low population density, where the number of users is not enough to maintain running a fleet of vehicles with some periodicity during the day.

Bus on demand, also know as dial-a-ride, is inserted in Demand Responsive Transport context. Demand Responsive Transport is an advanced passengers’ public transportation, user oriented, characterized by having a fleet of vehicles of small/average dimension moving with flexible itineraries and schedules in a sharing mode of travelling from an origin to a destination and always focused on the needs of its passengers. Demand Responsive Transport systems are suitable for a public transport service in rural or low-demand areas, where regular service has difficulty responding appropriately.

Inside flexible services there are several kinds of services, like Paratransit\(^1\), that its objective is to transport persons with low mobility, and xTransit\(^2\) [4], whose main objective is to lead people form outskirts of urban areas and into those using vehicles with smaller dimensions than buses, dynamically shared with others, aided by advanced communications and offering mobility similar to the car.

Most of these services are implemented in areas with low demand and in systems that do not work in an integrated way with existing services. Usually, this kind of services work in a given area which obligates people to wait for another mean of transportation to go outside of the covered area, since there is no integration with fixed services, although there are solutions that contemplate this problem [3].

Bus on demand services can be grouped by four levels of service [4]:


\(^2\) [http://www.ecoplan.org/general/xtransit.htm](http://www.ecoplan.org/general/xtransit.htm)
• Level 1 – Fixed itineraries. The user can book the service which is based on fixed journeys with predefined stops.
• Level 2 – Fixed itineraries with deviations. Itineraries and timetables are only partially predefined. In particular itineraries can change on demand with route deviations at fixed points so that optional stops are added to the original journey.
• Level 3 – Itineraries with predefined points. Itineraries can be towards predefined points of public interest such as parking, rail stations (many to few) or they can join a larger area with free timetables towards predefined points (many to many).
• Level 4 – Free itineraries not including predefined points. Itineraries and points are free and the origin and destination of the passengers are not predefined; can be similar to the taxi service (door to door).

2.2. Dial-a-Ride Problem
The dial-a-ride problem [3] focuses in calculate itineraries and schedules of vehicles for a certain number of users that make transportation requests from an origin to a destination. Usually, the same user makes two requests: a going and a return requests. In some cases, users may define a time window, which represents a time interval to satisfy the request that can not be exceeded. This time window indicates that the trip to satisfy the request can not start before the lower limit of that interval and can not end after the upper limit of that same interval.

The objective of the resolution of this problem is to calculate the itineraries minimal cost to satisfy the maximum requests, based on a set of constraints. In the resolution of this problem is necessary to consider two main objectives:

• Minimize costs of the total satisfaction of the requests;
• Maximize requests satisfaction with the available vehicles.

For both objectives there must be implicit the minimization of client dissatisfaction due to an excessive trip time (exceed the arrival time limit) or a deviation of service time (an excessive difference between the real begin/end time and the suggested one). [5]

The construction of dial-a-ride problem solutions always has three main decisions associated [6]:

• Determination of sets of users that will be served by the same vehicle;
• Sequentially put users on the itinerary of that vehicle;
• Schedule activities of loading, unloading and driving in each itinerary.

This problem solution consists in the definition of algorithms and heuristics to calculate routes and schedules. There are four models for this problem:

• Static single vehicle dial-a-ride problem;
• Static multi-vehicle dial-a-ride problem;
• Dynamic single vehicle dial-a-ride problem;
• Dynamic multi-vehicle dial-a-ride problem;

In the static dial-a-ride problem all the requests are known before the beginning of the processing and there will not be done any request in the future. There are two variants of the static dial-a-ride problem: the single and the multi-vehicle variants. The difference between them is that, in the first, there is only one vehicle that will try to fulfill all requests and, in the second there are several vehicles (two or more).

In the dynamic dial-a-ride problem the requests emerge dynamically over time, but no information is known about future requests. When a new request emerges for the time instant $t$, there is a planned solution till that time. Since that all the requests before $t$ are already processed and are not relevant, the problem is in the reoptimization of the portion of the solution from the instant $t$, where the new request will be added. Such as in the previous model, there are two variants of this model: the single and multi-vehicle variants.

2.3. Existing Solutions

Nowadays there are several solutions for bus on demand, both commercial as well as research. The solutions that can be seen in [7], [8] and [9] are good examples of commercial solutions. As research solutions examples there are the solutions in [3] and [10].

All the referred solutions calculate automatically itineraries and schedules for the vehicles fleet. With the exception of [8], all the solutions lack the integration with other solutions, being unique and closed solutions.

The reservation request of all solutions varies between phone call and web, and, sometimes, both ways. When the only used way to do a reservation request is the phone call, the process needs human intervention to insert the request in the system, which implies that the whole process is not automatic. But the use of web in [3] provides an automatic way of requests insertion.

Another disadvantage of these solutions, except [3], that is based in a service that varies between a dynamic and a fixed service, is the lack of integration with fixed services, which means that they are only used to manage dynamic services and do not interoperate with fixed services.

3. DEVELOPED SOLUTION

The solution developed for this problem has as main objective to automate the sending and reception of transportation requests between two places, an origin and a destination, and the dynamic calculation of itineraries and schedules.
The functional architecture of the developed solution can be seen on Figure 1. The solution is divided in two parts: the client solution and the central solution.

The client solution supports clients’ trips reservation requests and give a response to that requests, if the request was or was not reserved. This solution enables that the client can perform a cancelation request to a previous reservation request. To allow the automation of all the process, the solution permits the client to fulfill his requests using a cell phone or an e-mail provider.

To make a trip reservation, the client indicates the places of departure and arrival, as well as the time to start the trip and a maximum time to arrive at the destination, giving a time window in which the service may be done. The message with this data will be sent, via SMS or e-mail, to the client system.

After the arrival of the message at the client system, it will be processed and it will be verified if all the parameters are right, such as if the given time is not lower than the actual time (it can not be made any reservation before the actual time), and after that all the information is sent to the central system.

![Figure 1 – Functional architecture of the developed solution.](image)

The central system main objective is to verify if it is possible to reserve a trip to satisfy the request, and if it is possibly then calculate which itinerary should be done and the schedule to satisfy that same request.

When the request information is received by the central system it will be verified if the places that go along with the request exist in the system, and, if so, it will also verify if the client that is asking for a dynamic trip is registered in the system. The client verification is done to prevent ghost requests, requests that are made and no one appears at the departure point, and it is done by searching if the cell phone number, or email address, corresponds to any registered user.
If one of these verifications fails, the system answers to the client system transmitting that the place(s) or the client does not exist in the system and due to that the request can not be fulfilled. Otherwise the request is inserted in the system and waits for processing. At this time, the central system transmits to the client system that the request is inserted in the system and the registration number, so that the client may know what state it is his request.

After receiving the confirmation or denial of the request registration at the central system, the client system will prepare a response message for the mean of communication by which the request was made, and send the right response. If the registration was succeeded then the message is written with a succeed response plus the registration number. If there is an error in one of the verifications, the response message has a failure message and the reason why it was not succeeded: client or place does not exist in the system.

In the mean time, the central system verifies the existence of requests registered in the system, which is made with the use of a scheduler in the service manager. This scheduler verifies if there is any request that has exceeded a threshold of time from the actual time till the requested initial time. If there are requests that fulfill this verification then it is verified the possibility of satisfaction of each request, and, if it is possible, calculate the itineraries and schedules to fulfill each request.

To verify the possibility to fulfill each request it was developed an algorithm to calculate itineraries and schedules. The first step made by this algorithm is to verify if there is some vehicle to satisfy each needed trip. If there is any vehicle then the requests can not be fulfilled and it is send back that information to the client system which will send a message to the user informing that his request can not be fulfilled. When there are vehicles available then is verified if the time window given in the request allows to fulfill it. If yes then is verified if there is a dynamic service scheduled, and if not then it will be created a new one that will have origin at the depot, will pass through the requested departure place and end at the arrival point. If it is not possible to fulfill the request with the given time window that information will be sent back to the client system, which will send a message to the client informing that is not possible to fulfill his request.

If there are scheduled dynamic services, it is not reasonable to create a new service without testing if it is possible to add/integrate the requested trip in one of those services. First, it will be verified if there is some trip in those services that begins at the requested origin place in the requested time window. If there is at least one trip that fulfills that requirement then is verified if there is a trip in the same service that ends at the requested destination place in the requested time window. If both requisites are satisfied and the number of free seats during all the trips from the origin to the destination are greater than zero, than the request can be satisfied without changing the vehicle itinerary.

If there is not some trip that ends at the requested destination place, it will try to insert a new trip during the service, after the beginning trip, changing the itinerary of the vehicle to pass through the destination place without exceed the maximum delay time of all trips after the inserted one and the number of free seats. If it is not possible to insert that trip, then is verified if there is some service with
trips that end at the requested destination place in the requested time window. If exists such trip, then it will try to insert a new trip in the same service that starts at the requested origin place before the found trip, that starts after the requested departure time and that the insertion of this trip does not exceed the maximum delay time of the next trips and the free seats between this new trip and the end trip. In both cases, the search of insertion of a new trip ends when the number of free seats does not permit its insertion or if it is found a point in the itinerary that permits its insertion.

If there is not any trip with its beginning at the requested origin place and/or with its ending at the requested destination place and/or it is not possible to insert a trip during those trips, then it will try to insert a new trip at the end of the existing dynamic services. The reason why this test is done is due to minimize the time difference between the schedule transmitted to the client and the real one. To insert a trip in the end of a service it is verified if the service does not end after the given time window, cause only those have the possibility to receive this new trip. If the given time window allows the insertion of the new trip at the end of a service then the new trip will be a trip with start at the requested origin and end at the requested destination. If the destination of the last trip of the service is the same of the requested origin, then only the new trip will be inserted at the end of the service, otherwise will be created a new trip that starts at the last destination of the service and ends at the requested origin.

If for some reason it is not possible to insert the trip at the end of the service, the last possibility is to traverse all the trips of all services till find any possible combination of trips that allows the insertion of the requested places in the itinerary.

At the end of each step, before inserting a request in a service is verified if it was not cancelled. If not then it is inserted in the service and the central system will transmit that information to the client system adding the information of the trip: origin place, foreseen boarding time, destination place, foreseen landing time and the vehicle information that will satisfy the request. With this information, the client system will create and send a message to the client informing that the request was satisfied accompanied with the trip information.

Besides booking a trip, the developed solution allows too cancel a booked request. A book cancelation is made the same way as a book request, it means, a book cancelation is made by SMS or email messages.

After receiving a book cancelation, the client system passes that information to the central system. At the central system it is verified if the client is registered, the same way as described earlier, and if the registration number given corresponds to any request made by that client. If there is any registration with the given number for that client, then that information is sent back to client system that will transmit to the client that the cancelation was not made due to the inexistence of any book with the registration number to that client.

If it passes all validations then it is verified in which state the request is. If the request is in “Registered” or “Processing” state then it is only needed to change its state to “Cancelled”. If its state
is “Booked” then it is needed to remove the trips that will satisfy that request from the service. If that trips are done to satisfy other requests, then it is only needed to update vehicle’s free seats and change the request state to “Cancelled”. If there is none request that uses those trips, then they are removed.

Because it is intended to minimize schedule changes, and because there is no interest that a vehicle arrives or departs from a stop earlier than scheduled, when trips are removed from a service the programmed schedule will remain with the difference that the vehicle will stay stopped at the stop before the removed stop waiting till the necessary time that allows it to arrive at the next stop at the programmed schedule.

The removal of a stop from a service consists in removing the trips to and from that stop and insert a new one that will leave from the previous stop of the removed one and will arrive at the next stop. After removing the intended trips, the number of free seats will be updated and then the request is marked as “Cancelled”.

To implement this solution it is necessary to have a SMS Gateway and an email server, which are responsible to receive and send SMS and email messages to and from the system. These two parts of the system are the link between client users and the solution. Clients will send a message through a cell phone, or some device that allows sending SMS messages, or through an email account. If the message is sent by SMS, then the SMS Gateway will send the message to the client solution via HTTP. Otherwise, if the client sends an email message it is necessary that the client solution gets the message from the email server, that should be done by POP3, so all the email messages are read and deleted not being read more than once time.
Because it is interesting to have systems’ portability and interoperability, the communication between the client and the central system is done by web services. The use of web services allows that any of the systems may be changed by another solution that provides the same services, and there is none language orientated limitation.

4. Tests

The developed solution tests try to see if its implementation works as expected. For that were done tests to verify if the book message sent by a client, by SMS or by email, arrives at the client system and, then, if all the data is passed to the central system.

To do that, a SMS message and an email message were sent to book two trips, with a time interval that allows their satisfaction in the same service. Both the requests were registered in the system, although the response to the email request took more time to arrive, which is explained with the need of the client system to go read the email messages to the server, which is done with some time interval. After some time, both requests were booked and that information arrived to the senders.

To test book cancelation, two cancelation messages were sent, with a time interval between them, to cancel the booked trips above. As expected, both trips were cancelled and the messages of their cancelation sent back to the senders. Besides the trips cancelation, once there were no more trips in the service, the service was also removed.

To test the schedules and itineraries calculation algorithm performance it were inserted 6169 requests in the central system, to be processed between the 22:30 and the 1:30 schedule, and provided 100 on demand vehicles to satisfy them. From the 6169 requests only 1079 were served and the calculation took 56 hours and 18 minutes to end. After this test, another one was made but this time only the 1079 served requests were inserted in the system. This time the calculation only took 2 hours and 52 minutes and, as expected, all the requests were satisfied.

5. CONCLUSIONS

Even if the objective of this work was not the creation of a schedules and itineraries calculation algorithm, since that only the algorithm is reason for a thesis, the creation and implementation of such algorithm was needed to test the solution.

Although the created and implemented algorithm lacks some validations, such as the minimization of the travel distances, due to processing time minimization, as can be seen in the tests section, the processing of a large amount of requests starts to be prohibitive and turns to be almost impossible to respond in time. This is one of the major issues that are needed to overcome and, as can be seen in [11], these issues have been studied and problems with more than 100 requests can be difficult to solve optimally.
On the other hand, the results of this test may be justified with the tight time windows that accompanied the inserted requests, which can cause lack of time in the request satisfaction. Once all the requests have to go through all the algorithm steps before been marked as not satisfied, becomes understandable that after a large number of booked requests the number of comparisons is so large that obligates a larger processing time. The small number of served requests can also be explained by the small number of vehicles available to satisfy them, a superior number of vehicles leads to more satisfied requests.

Finally, with this work has been shown that it is possible to automate all the process of booking services, from the registration till the calculation of schedules and itineraries and response to clients. With the use of new technologies it is possible to book a trip in any place and automate all the process with lesser human intervention, avoiding the need of an operator at a central receiving calls and inserting requests in the system avoiding, also, mistakes and failures and speeding up all the process.

References


[8] SAPRASOL SA Practical Solutions. MJ SCHEDULING_ROUTING SYSTEM.

[9] GIRO Inc. GIRO-ACCES
