

Semitag – External Specifications Documents

Master M1 MOSIG, Grenoble Universities

TCHECHMEDJIEV Andon
TCHOUGOURIAN Tigran

05/04/2011



Supervisors :
Gaëlle CALVARY
François BERARD

The goal of this project is to put in practice the User Centered Design method for designing interactive computer applications.

Contents

1. Design	2
1.1. User Model	2
1.2. Environment Model	2
1.3. Platform Model	2
1.4. Task Model	2
2. Detailed scenarios.....	4
3. Interface descriptions	10
3.1. General description	10
3.2. The input form	11
3.3. Search bar algorithm.....	13
3.4. Results	14
3.4.1. The Summary view	14
3.4.2. The Details view	15

1. Design

1.1. User Model

The application is designed for novice users familiar with paper based forms.

1.2. Environment Model

The application is provided by the semitag web site. It has to be accessible from any computer which has an internet connection and a web browser (Internet Explorer 9.0/Firefox 4.0/Chrome 10.0 from Windows, Firefox 4.0 for Linux, and Safari 5.0 for Mac)

1.3. Platform Model

The application is meant to run on a web browser on a home computer. It uses the mouse and the keyboard to navigate and get input.

1.4. Task Model

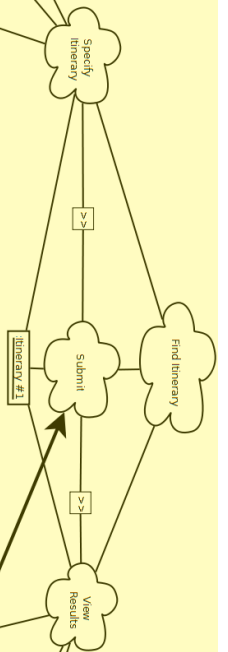
It was decided to use Concurrent Task Trees to represent the Task Model of the application. There have been, however, several modifications to the standard notation so as to make it more legible in this particular context.

The first modification is that instead of linking nodes of the tree directly to UML classes representing the data model of the application; it has been decided to link them to objects instead. That is boxes containing “:ClassName #N”, where N is the number of the instance in question, as specified by the UML notation. The purpose of this modification is to eliminate unnecessary links as several tasks refer to the same classes.

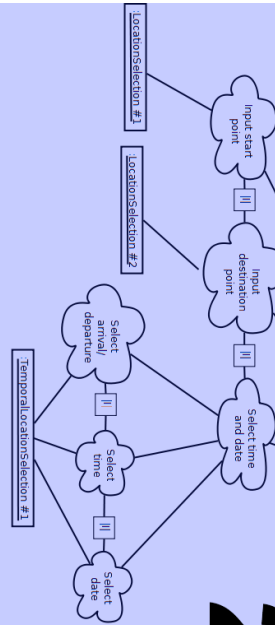
The second modification of the model was the use of an arrow to mark that the system has reverted in a previous situation. Namely, in this particular case to express the fact that when the itinerary is refined and the search button is pressed once more, the system will be in the same state in terms of the task at hand as when the itinerary is first searched for. The modification was mainly useful to avoid the recursive duplication of a whole chunk of the tree.

In order to make the reading of the diagram more convenient it has been split into four zones as shown on the full diagram below by the numbers and colour zones. The enlarged diagrams of each zone will follow the full diagram.

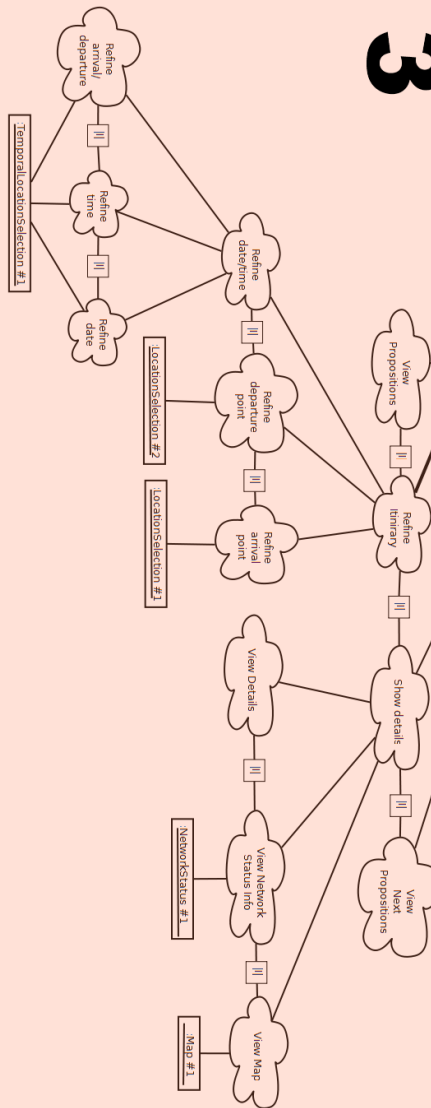
1



2



3



4

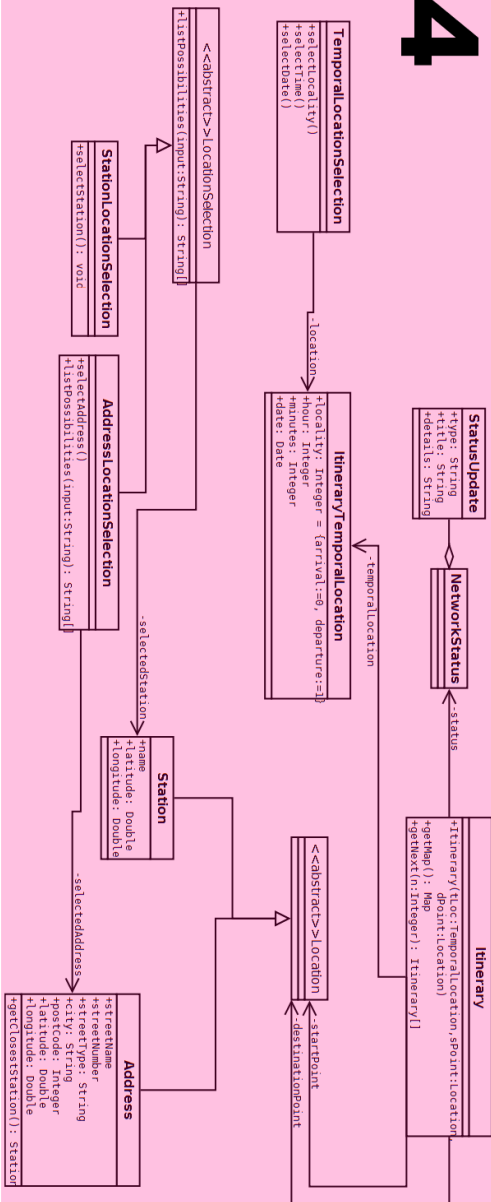


Diagram for Zone 1:

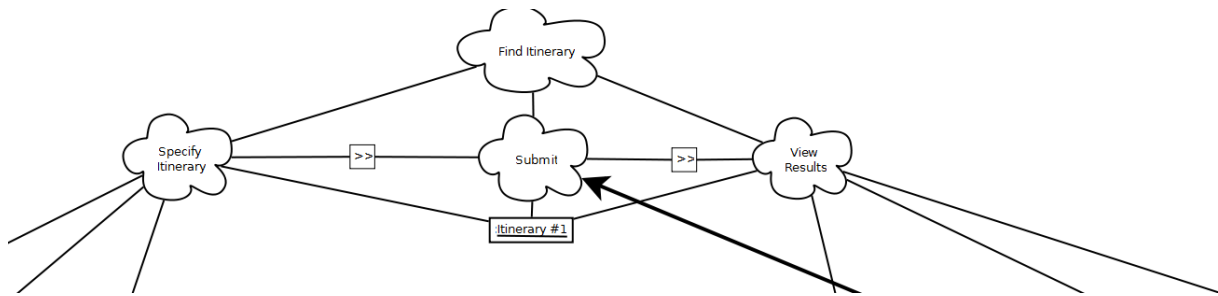


Diagram for Zone 2:

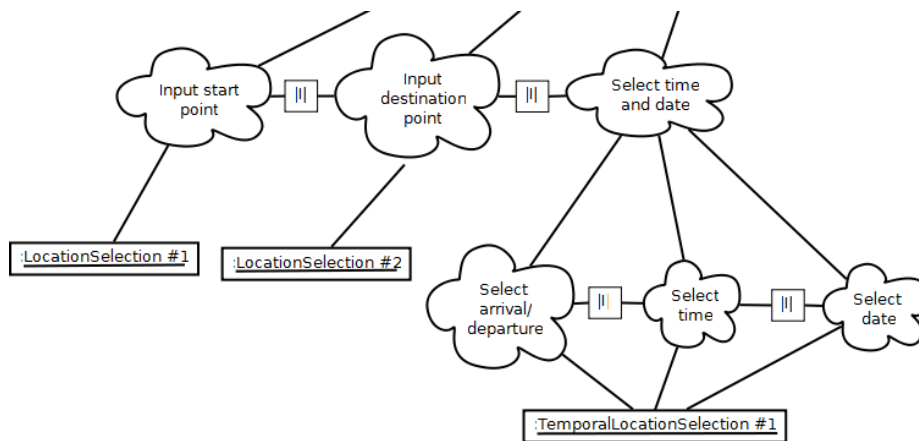


Diagram of Zone 3:

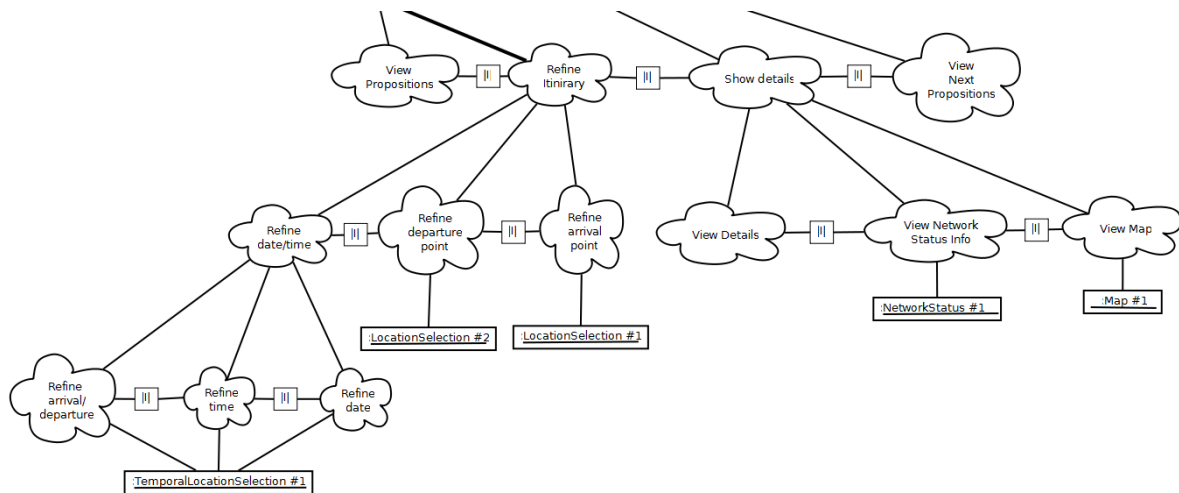
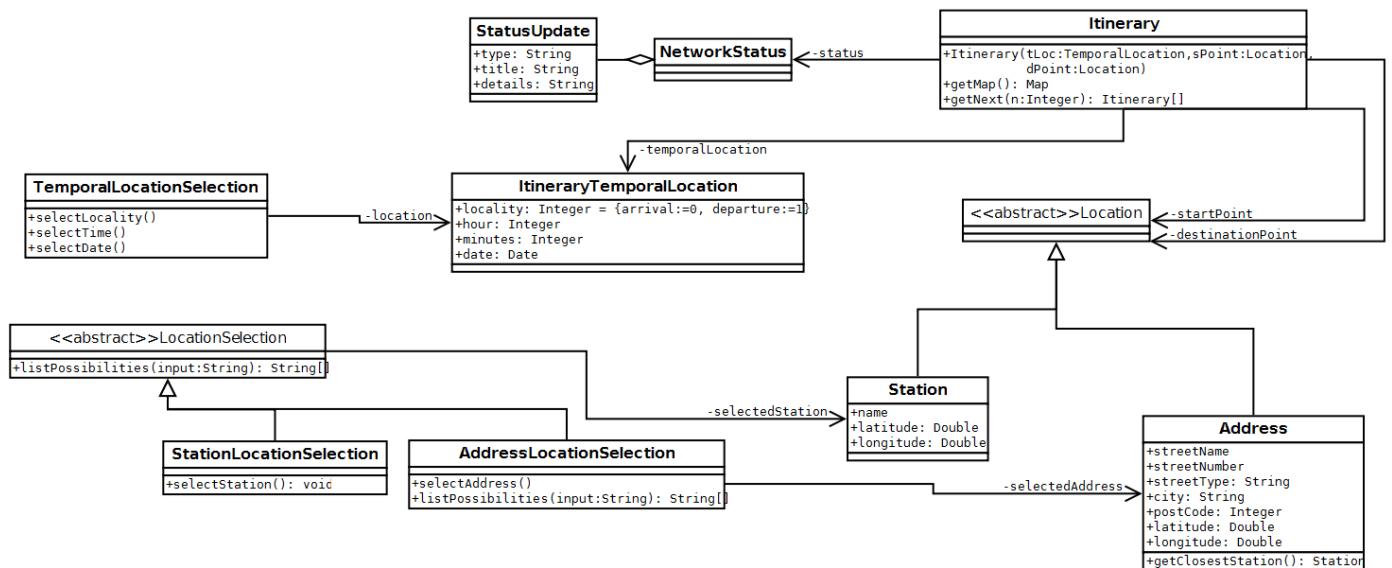


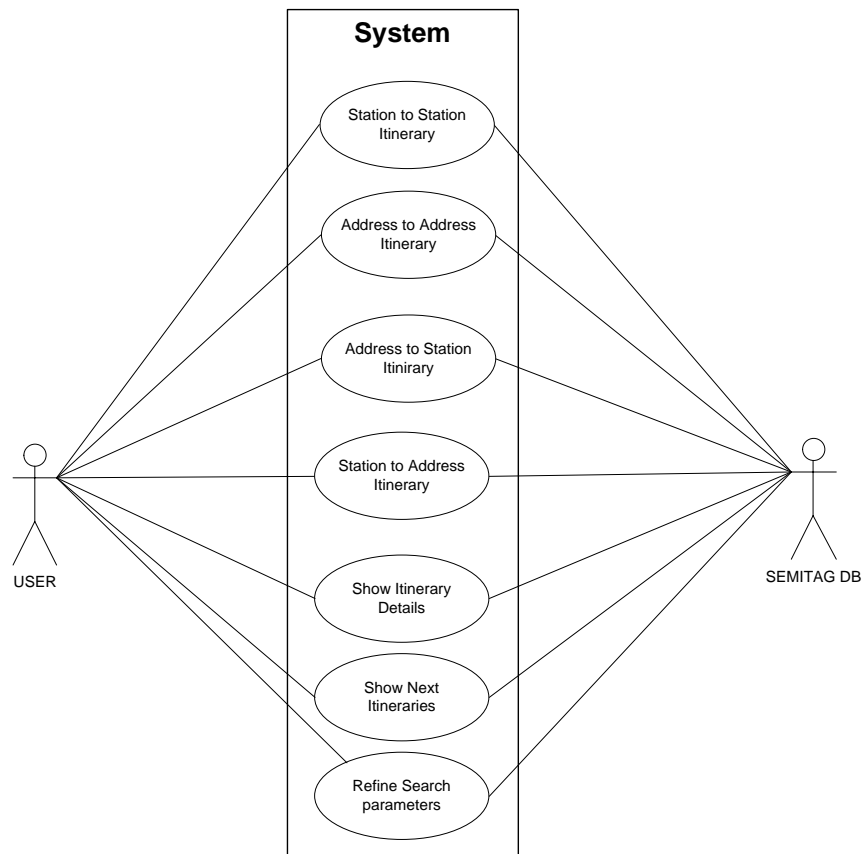
Diagram for zone 4:



For more convenience the diagram in its full resolution version is available at <http://theainur.free.fr/mosig/hci/TaskModelOrig.png>

2. Detailed scenarios

From the task model above and from the scenarios elicited during the analysis phase it is possible to identify the set of use cases for the application. Bellow can be seen the use cases identified for this project.



From the use cases above have been devised detailed interaction scenarios between the user and the system. Note that the use cases “Address to Station” and “Station to Address” are exactly the same except for the reverse order of the input of the departure and destination points; that’s why only one interaction scenarios presented for both of them.

Station to Station Itinerary	
Issues:	The user wants to know how to get from a station to another
Requirements:	<ul style="list-style-type: none"> ➤ An Internet Access ➤ Prior knowledge of the start and destination stops.
Actors:	<ul style="list-style-type: none"> ➤ The user ➤ The SEMITAG servers
Goals:	Find the optimal (in relation to travel time) itinerary between two stations at a given time
Assumptions:	The user has a least moderate knowledge of the SEMITAG network
Scenario steps:	<ul style="list-style-type: none"> ➤ The user starts typing the name of the start station ➤ The system incrementally shows the available stops matching the currently typed characters <ul style="list-style-type: none"> ▪ The user fully types the name OR Picks one of the propositions. ➤ The same process is repeated for the input of the destination station ➤ The user selects: whether to select a departure or arrival time, a date and a time. ➤ The user submits the form ➤ The system displays three successive possible itineraries, and inactivates the input form
Exception cases:	<ul style="list-style-type: none"> ➤ If one of the source or destination station does not exist. <ul style="list-style-type: none"> ➤ The system displays a list of the first few (limited to 7) stations closest to what was typed. ➤ If one of the fields is empty the systems points it out to the user ➤ If there are no available itineraries between the stations selected by the user, the system displays a message stating that no itineraries could be found with the selected parameter.

Address to Address Itinerary	
Issues:	The user wants to know how to get from an address to another address
Requirements:	An internet access.
Actors:	The user, The SEMITAG servers
Goals:	Find the optimal (in relation to travel time) itinerary between two addresses at a given time
Assumptions:	The user has no knowledge about the SEMITAG network
Scenario steps:	<ul style="list-style-type: none"> ➤ The user starts typing the name of the departure street/city ➤ The system incrementally shows the available streets/cities matching the currently typed characters ➤ The user fully types the name OR picks one of the propositions. ➤ The user starts typing the street number ➤ The system incrementally shows the available numbers matching the currently input characters ➤ The user either fully types the number OR selects one of the propositions. ➤ The same process is repeated for the selection of the destination address. ➤ The user selects: whether to select a departure or arrival time, a date and a time. ➤ The system displays eventual network status information and three successive possible itineraries ➤ The user submits the form ➤ The system displays three successive possible itineraries, and inactivates the input form
Exception cases:	<ul style="list-style-type: none"> ➤ If one of the street/city combination do not exist: <ul style="list-style-type: none"> ➤ The system displays a list of the first few (limited to 7) closest to what was typed. ➤ The closest stop to the selected addresses are used for the itineraries ➤ If no street number is selected, the start of the street is considered. ➤ If there are no available itineraries between the addresses selected by the user, the system displays a message stating that no itineraries could be found with the selected parameter.

Station to Address Itinerary	
Issues:	The user wants to know how to get from a station to an address
Requirements:	An internet access. Partial knowledge of the tag network
Actors:	The user, The SEMITAG servers
Goals:	Find the optimal (in relation to travel time) itinerary between two addresses at a given time
Assumptions:	The user has a partial knowledge about the SEMITAG network
Scenario steps:	<ul style="list-style-type: none"> ➤ The user starts typing the name of the start station ➤ The system incrementally shows the available stops matching the currently typed characters <ul style="list-style-type: none"> ▪ The user fully types the name OR Picks one of the propositions. ➤ The user starts typing the name of the destination street/city ➤ The system incrementally shows the available streets/cities matching the currently typed characters ➤ The user fully types the name OR picks one of the propositions. ➤ The user starts typing the street number ➤ The system incrementally shows the available numbers matching the currently input characters ➤ The user either fully types the number OR selects one of the propositions. ➤ The user submits the form ➤ The system displays three successive possible itineraries, and inactivates the input form
Exception cases:	<ul style="list-style-type: none"> ➤ If the departure station does not exist: <ul style="list-style-type: none"> ➤ The system displays a list of the first few (limited to 7) stations closest to what was typed. ➤ If the destination street/city combination does not exist: <ul style="list-style-type: none"> ➤ The system displays a list of the first few (limited to 7) closest to what was typed. ➤ The closest station to the selected arrival addresses is used for the itineraries. ➤ If no street number is selected, the start of the street is considered. ➤ If there are no available itineraries between the addresses selected by the user, the system displays a message stating that no itineraries could be found with the selected parameter.

Show Itinerary details	
Issues:	The user wants to see the details of an itinerary
Requirements:	An internet access. The result page of an itinerary search
Actors:	The user, The SEMITAG servers
Goals:	Show the user more information about the selected itinerary
Assumptions:	The user has already performed a search and wants more details about a given proposed itinerary
Scenario steps:	<ul style="list-style-type: none"> ➤ The user decides on an itinerary and selects “Details” ➤ The system expands the itinerary by showing: <ul style="list-style-type: none"> ▪ A detailed step-by-step view of the itinerary ▪ A map showing the itinerary in the city ▪ Eventual information about the status of the network (strike, weather, etc)
Exception cases:	

Show next itineraries	
Issues:	The user wants to see the following itineraries (in terms of time)
Requirements:	An internet access. The result page of an itinerary search
Actors:	The user, The SEMITAG servers
Goals:	Show the user the three next possible itineraries
Assumptions:	The user has already performed a search and wants to see later itineraries
Scenario steps:	<ul style="list-style-type: none"> ➤ The user selects “Later connections” ➤ The system displays a list of three more itineraries
Exception cases:	

Refine search parameters	
Issues:	The user made a mistake in the input of the search parameters
Requirements:	An internet access. The result page of an itinerary search
Actors:	The user, The SEMITAG servers
Goals:	Allow the user to change any of the information input about the parameters of the itinerary search
Assumptions:	The user has already performed a search.
Scenario steps:	<ul style="list-style-type: none"> ➤ The user clicks on the title of the search area ➤ The system reactivates the search area so as to allow the modification. ➤ The user modifies one or more value as described in the respective itinerary search scenarios and then submits ➤ The system displays three successive possible itineraries, and inactivates the input form.
Exception cases:	The same exception cases as the respective search scenarios

3. Interface descriptions

Upon arriving on the semitag itinerary search page, users should be able to browse the search results in at most 2 clicks (submit actions).

3.1. General description

The interface is divided in two frames as shown on Figure 3-3. Frame A handles the input of the user, whereas Frame B handles the display of the results.

Frames can be toggled up or down (Figure 3-1 and Figure 3-2) so as to not pollute the display. When the user starts a new search, Frame B is completely hidden. Once the request is submitted (e.g. [3.2 The input form](#)), Frame A is closed (only the title of the frame is visible) whereas Frame B is visible and the results appear.

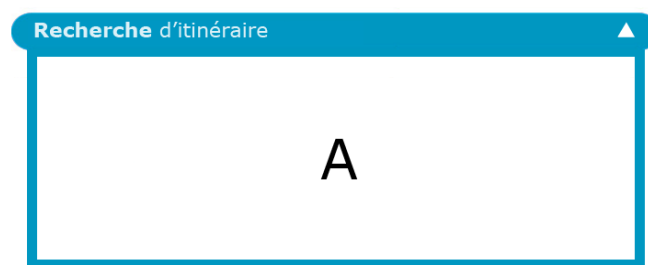


Figure 3-1: Open Frame

The user can always reopen Frame A to refine the search parameters. That means that frame A should remember the input values to help the user. Frame B has to be open during the refining phase, and reloaded with the new results after the user submits.

Recherche d'itinéraire

Figure 3-2: Close Frame

[3.2 The input form](#) instantiates Frame A, whereas [3.4 Results](#) instantiates frame B.

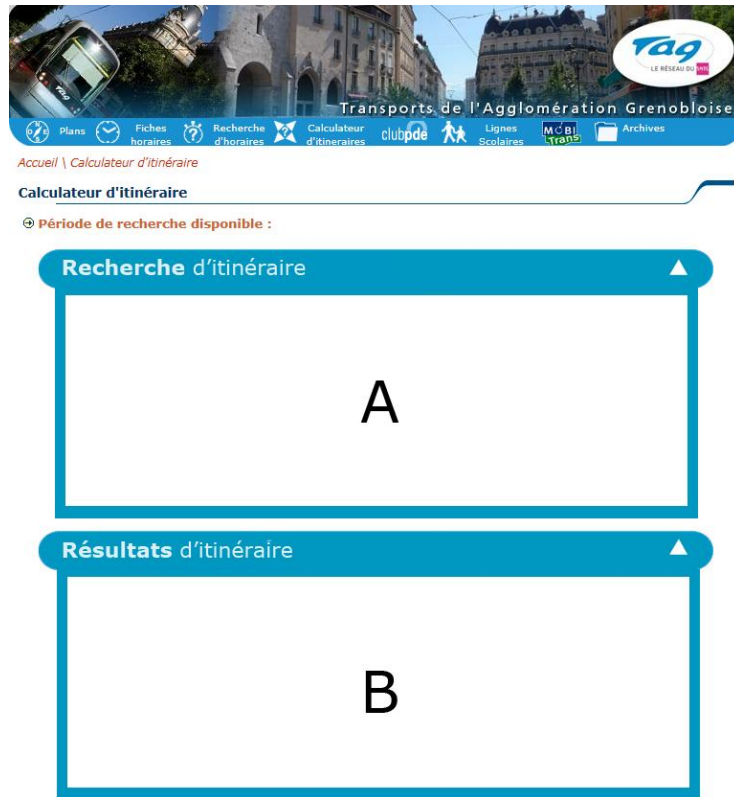


Figure 3-3: General view of the function

3.2. The input form

The input form is divided in 4 different information parts:

From:	<input type="text"/>
To:	<input type="text"/>
Date:	<input type="text" value="30"/> <input type="text" value="March 2011"/> <input type="button" value="Calendar"/> <input checked="" type="radio"/> Departure <input type="radio"/> Arrival At: <input type="text" value="10:34"/>
<input type="button" value="Search"/>	

Figure 3-4: Unique form to submit before to find an answer

- The “From” input field: indicates the source of the itinerary. The user sets the departure point (e.g. [Search Bar algorithm](#)).
- The “To” input field: indicates the destination of the itinerary. The user sets the arrival point (e.g. [Search Bar algorithm](#)).
- Date of search: The user has two ways to set the date: With the drop down list or with a calendar script.
 - The drop list should respect the number of days for each month.

- The calendar and the drop down lists are by default set to the current day.
- Type of search: By default “Departure” is selected and the “At” field is set to the current time + 5 min. The “Arrival” option represents the time that the user wants to arrive at the destination point. If the user clicks with the mouse to “Arrival”, it will automatically erase the content of the “At” field and set the cursor on that field. If the user select back the “Departure”, then the cursor is moved to the “At” field.
 - The “At” field should be smart enough to let the user only input numerical values and the “:” character.

Example:

Digits	Allowed	Transformed
1	1	01:00
2	01	01:00
2	24	00:00
2 with colon	01:	01:00
2 with colon	1:1	01:10
2 with colon	:10	00:10
3	100	10:00
3	230	23:00
3 with colon	23:0	23:00
3 with colon	1:10	01:10
4	1000	10:00
4 with colon	10:00	10:00

Figure 3-5: Input allowed and transformation

Digits	Not allowed
2	25
2	99
2 with colon	25:
2 with colon	0:6
2 with colon	:60
3	250
3	106
3 with colon	23:6
3 with colon	1:60
4	2460
4 with colon	10:60

Figure 3-6: Corrupted input

The “search” button will submit the form if all the values are filled. If there is a missing field or wrong data, a red rectangle has to surround the incorrect field and the user has to set the appropriate new values. If for any reason the frame was closes, it has to be opened again so that the user sees the errors.

The following information is missing

To:

Figure 3-7: A missing field

In the case of the “From” and “To” fields, the red rectangle is appears if no matches were found. If there are multiple matches, then a drop down list will be displayed so that the user can disambiguate the choice.


From: 

Figure 3-8: Possible Matches

3.3. Search bar algorithm

The search interface uses a form to get information about the source and destination address/station. The input fields “From” and “To” are not mere input field as they use an algorithm that finds for the user the exact name of the address or station that they are searching for. The algorithm automatically displays under the input field the best matches. It is an incremental search; therefore, in order to avoid making requests to the database too often, the incremental completion should start only after 3 characters have been typed. At most 7 values are returned. The algorithm should always favour addresses as the first choice.

To each matching row is associated an icon on the right. A house represents an address, a bus represents a bus station, and a tramway represents a tramway station.

From:

Place **Victor** Hugo

Station **Victor** Hugo






Figure 3-9: A user tries to find a station or address with "Victor" inside

To help the user, system should display the user input in bold in the propositions. The matching is only performed on full words, but not on their individual composing letters.

Example:

A user searches for “André”. He starts to write “Andr”. The algorithm shouldn’t return **Alexandra**

Signature of the function:

Precondition: String of at least length 3

Find (String address): List<Complete address, Image>

Postcondition: a list of 7 items at most; an item is composed of the full name of street/station and of the corresponding image (bus, house, tramway)

3.4. Results

Once the input form is completed and submitted, Frame A is toggled up and frame B is toggled down. The results should be presented in the current form (e.g. Figure 3-7). At most 3 results with different paths. They should be sorted by departure time, and the first one should be selected.

At:	Station/Stop:	More:	Date:	Duration:	Summary:
<input checked="" type="radio"/> departure 10:50 <input type="radio"/> arrival 11:11	Place Victor Hugo Station Chavant	Details	30 March 2011	22min	→
<input type="radio"/> departure 10:53 <input type="radio"/> arrival 11:12	Place Victor Hugo Station Chavant	Details	30 March 2011	19min	→
<input type="radio"/> departure 10:57 <input type="radio"/> arrival 11:18	Place Victor Hugo Station Chavant	Details	30 March 2011	21min	→

[Later connections](#)

Figure 3-10: The results

3.4.1. The Summary view

The “Summary” represents a fast way for the user to see what he/she should do during the trip.



A pedestrian is shown if one of the following actions has to be done by the user.

- The user has to go from one address to another one. If multiple address changes are performed, then the image is shown only once.
- The user has to go from one address to a station (bus or tramway).
- The user has to go from one station (bus or tramway) to another station (bus or tramway).
- The user has to go from one station (bus or tramway) to an address.



A bus with the identifier of the line is shown if one of the following actions has to be done by the user.

- The user has to take the bus



A tramway with the identifier of the line is shown if one of the following actions has to be done by the user.

- The user has to take the tramway



The arrow represents the next step to be performed.

In the bottom of Figure 3-10, the user can see later connections. It just recalculates the itinerary with a new departure time or arrival time (if possible). Otherwise later connections

are hidden. In this subset of the application the feature allowing users to see previous or earlier connection has not been implemented.

3.4.2. The Detailed view

Each row can be expanded to see in detail the steps described in the summary. For that the user has to click on the eye or “Details”. The detailed view is also displayed in frame B, but is toggled down just under the selected line. If a user clicks again on the eye or on “Details” the detailed view is toggled up and the screen reverts back to Figure 3-10.

At:	Station/Stop:	More:	Date:	Duration:	Summary:	
<input checked="" type="radio"/>	departure 10:50 arrival 11:11	Place Victor Hugo Station Chavant	Details	30 March 2011	22min	→
At:	Travel plan:		Map:			
	departure 10:50 arrival 10:51	Walk from Place Victor Hugo to the Station Victor Hugo Allow about 1min				
	departure 11:00 arrival 11:11	Take the Bus 13 at Station Victor Hugo to 3 Dauphins Arrival Chavant				
Strike : A strike is organized by CGT. It starts near to your departure station.						
<input type="radio"/>	departure 10:53 arrival 11:12	Place Victor Hugo Station Chavant	Details	30 March 2011	19min	→
<input type="radio"/>	departure 10:57 arrival 11:18	Place Victor Hugo Station Chavant	Details	30 March 2011	21min	→

Later connections

Figure 3-11: Detailed view toggled down

As shown on the image bellow, a template is used to describe the “Travel plan”.

If a user has to walk from one address to another one, this is the template to use.



Walk from [address A] to the [address B]
 Allow about [time] min

If a user has to walk from one address to a bus/tramway station, this is the template to use.



Walk from [address A] to the Station [station name]
 Allow about [time] min

If a user has to walk from one bus/tramway station to an address, this is the template to use.



Walk from [Bus|Tramway] Station [number of the bus] to [address]
 Allow about [time] min

If a user has to take a bus/tramway, this is the template to use.





Take the [Bus|Tramway] at Station [station name] to [destination of the transport]
 Arrival [Bus|Tramway] [station name]
 Allow about [time] min

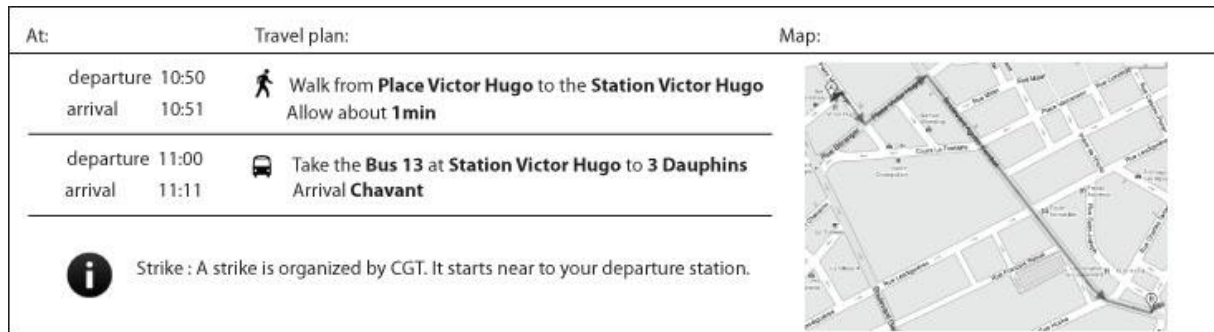


Figure 3-12: Detailed view of one path

The information area is visible if there is any important news about the traffic. The information is set by the administrator of the web site.

The map is a global view of the itinerary that the user has to take.