

PlaceFinder

LT2319/Dialogue Systems 2 Report

Github code: https://github.com/jacobcoles/LT2319_final_project

1. Introduction

My system is an interface for Google Maps, which should (ideally) be able to do everything that Google Maps could possibly do through a verbal interface. My system will have somewhat reduced functionality compared to this as it is a small project. A key example of what the system should be able to do is help the user find places and navigate to them.

2. Summary and APIs Used

The system uses a set of associated APIs provided by Google.

The first is called The Geocoding API, which enables the user to input an address, city, suburb etc. and converts it into a set of coordinates to enable us to ping other APIs.

The second is The Nearby Search API which will return a list of locations that are close to the location selected by the user; this search can be narrowed based on other parameters and keywords. This will enable the user to ask questions like “Where is the nearest Italian restaurant?”, and it should be able to return the name and address.

The final API is The Directions API, which enables the user to select two places, and get the directions between them. The user can also choose between driving, cycling and public-transportation.

3. Data collection

Scenario

The user would like to find locations and navigate to them. The user should also be able to narrow the search based on other features such as star-rating or category of restaurant. An example of when a user would use this system is when they might be in town and want to know where, for example, a nearby chinese restaurant is, and then get navigation directions there.

Collected material:

Recorded dialogue 1:

S> What would you like to do?
U> I'd like to find somewhere nearby that's Italian
S> Like an Italian restaurant?
U> Yep, something with good ratings maybe
S> We're pretty close to Johnny's Pizzeria, does that sound good?
U> Sounds good, whereabouts is it?
S> Would you like me give you some directions?
U> Yep
S> Go straight 50m and turn left at the roundabout
U> Are we going there?
S> Yeah, we're walking and tell me when you've done that and i'll keep giving you directions ok

U> Yep, cool I'm at the roundabout
S> Nice, alright so you've turned left there
U> Sure
S> Alright now go 100m and take the second right
...

Distillation 1:

S> What would you like to do?
U> I'd like to find somewhere nearby that's Italian
S> So, are you looking for an Italian restaurant?
U> Yep, something with good ratings maybe
S> The nearest is Johnny's pizzeria.
U> Sounds good, whereabouts is it?
S> Would you like to navigate there?
U> Yep
S> Go straight 50m and turn left at the roundabout
U> Are we going there?
S> I am giving you directions to Johnny's Pizzeria.
U> Yep, cool I'm at the roundabout
S> Have you turned left at the roundabout?
U> Sure
S> Continue for 100m and take the second street on your right.
...

Tagged 1:

S> <greet>What would you like to do?</greet>
U> <ask>I'd like to find somewhere nearby that's Italian</ask>
S> <ask>So, are you looking for an Italian restaurant?</ask>
U> <confirm>Yep, </confirm> <ask>something with good ratings maybe</ask>
S> <answer>The nearest is Johnny's pizzeria. </answer>
U> <ask:instructions>Sounds good, whereabouts is it?</ask>
S> <ask>Would you like to navigate there?</ask>
U> <confirm>Yep</confirm>
S> <answer:instructions>Go straight 50m and turn left at the roundabout</answer>
U> <ask>Are we going there?</ask>
S> <answer>I am giving you directions to Johnny's Pizzeria. </answer>
U> <confirm:instructions>Yep, cool I'm at the roundabout</confirm>
S> <ask>Have you turned left at the roundabout?</ask>
U> <confirm>Sure</confirm>
S> <answer:instructions>Continue for 100m and take the second street on your right.</answer>
...

Recorded dialogue 2:

S> What would you like to do?
U> Are there restaurants near Vasa that are pretty cheap? Something that's...uhh, at least 4 stars rating?
S> Yep, there are a couple places around there, want to hear about them?
U> Sure
S> Theres Sally's Italian which is right on the corner there or Big Burgers which is just down Avenyn like a hundred meters.
U> Ok...I'm kinda keen for burgers, is it easy to get to from Kungssportsplatsen?
S> Do you know where the Poseidon statue is?
U> Yep
S> Yeah, so just go that direction for like 200m and it'll be on your right like half way to the statue.
U> Ok sick
S> Do you want directions now?
U> Na, I'll be going later but thanks anyway
S> Ok

Distillation 2:

S> What would you like to do?
U> Are there restaurants near Vasa that are pretty cheap? Something that's...uhh, at least 4 stars rating?

S> Yes, there are two places matching your description within 150m of Vasaplatsen. Do you want to hear about them?
 U> Sure
 S> You have two options. Sally's Italian lies beside Vasaplatsen, whereas Big Burgers is 100m away.
 U> Ok...I'm kinda keen for burgers, is it easy to get to from Kungsportsplatsen?
 S> It is near Kungsportsplatsen. Do you want a better description relative to the Poseidon statue?
 U> Yep
 S> If you travel 200m in the direction of the Poseidon statue, it will be on your right.
 U> Ok sick
 S> Would you like more detailed navigation directions now?
 U> Na, I'll be going later but thanks anyway
 S> Ok

Tagged 2:

S> <greet>What would you like to do?</greet>
 U> <ask>Are there restaurants near Vasa that are pretty cheap?</ask> <ask>Something that's...uhh, at least 4 stars rating?</ask>
 S> <feedback>Yes,</feedback> <answer>there are two places matching your description within 150m of Vasaplatsen.</answer> Do you want to hear about them?
 U> <confirm>Sure</confirm>
 S> <answer>You have two options. Sally's Italian lies beside Vasaplatsen, whereas Big Burgers is 100m away.</answer>
 U> <confirm>Ok...</confirm><ask>I'm kinda keen for burgers, is it easy to get to from Kungsportsplatsen?</ask>
 S> <feedback>It is near Kungsportsplatsen.</feedback> <ask>Do you want a better description relative to the Poseidon statue?</ask>
 U> <confirm:instructions>Yep</confirm>
 S> <answer:instructions>If you travel 200m in the direction of the Poseidon statue, it will be on your right.</answer:instructions>
 U> <confirm:instructions>Ok sick</confirm:instructions>
 S> <ask>Would you like more detailed navigation directions now?</ask>
 U> <answer>Na, I'll be going later but thanks anyway</answer>
 S> <confirm>Ok</confirm>

4. Implementation

The system I eventually implemented had two primary functions; to find locations nearby the user, and to find navigation directions between two places. These two functions are separate but there are overlaps in the predicates they use.

Ontology:

The ontology has the sorts:

```
<sort name="city" dynamic="true"/>
<sort name="location_type_sort" dynamic="true"/>
<sort name="location_sub_type_sort" dynamic="true"/>
<sort name="transport_mode_sort" dynamic="true"/>
```

The first sort of 'city' is used for the user location and navigation locations (to/from) and is implemented as the following predicates:

```
<predicate name="dep_city_pred" sort="city"/>
<predicate name="dest_city_pred" sort="city"/>
```

This simplifies the system as we are able to minimise the number of possible locations we need to hard code, though at this point the system won't be able to implement complex locations, like addresses yet.

We also have the following predicates which allow the user to specify options of their search/information request:

```
<predicate name="location_type" sort="location_type_sort"/>
<predicate name="location_sub_type" sort="location_sub_type_sort"/>
<predicate name="transport_mode" sort="transport_mode_sort"/>
```

There are individuals associated with each of these predicates and sorts.

The following predicates correspond to the information we will be getting from our service interface. They are of type string, and are the information the user is requesting. 'get_locations' is the answer to the nearby location the user is asking about, and the 'get_next_direction' is the next direction in an instructional dialogue (there is only one because the system retrieves the information 'recursively' on the same variable).

```
<predicate name="get_locations" sort="string"/>
<predicate name="get_next_direction" sort="string"/>
```

The following actions are used for the instructional dialogue. The user will need to select between either getting the next direction or going back to the menu, to allow them to be able to keep requesting information (recursively) from the service interface.

```
<action name="get_directions"/>
<action name="back_to_menu"/>
```

Grammar:

The 'get_locations' predicate is implemented as a question with one-shot options. Here is a subset of this (many of the items were excluded here):

```
<question speaker="user" predicate="get_locations" type="wh_question">
  <one-of>
    <item>what is nearby</item>
    <item>what is around</item>
    <item>nearby</item>
    <item>what is nearby</item>
    <item>what is near <slot predicate="dep_city_pred"/></item>
    <item>is there a <slot predicate="location_type"/> near <slot
predicate="dep_city_pred"/></item>
    <item>is there a <slot predicate="location_sub_type"/><slot
predicate="location_type"/> near <slot predicate="dep_city_pred"/></item>
    <item>where is a closeby <slot predicate="location_sub_type"/> <slot
predicate="location_type"/></item>
  </one-of>
</question>
```

Whereas the 'get_directions' query is an action, so that we are able to (in the domain file) define a way of recursively getting information from the service interface. Here we first get the user request (using the following action), and will later use a question (recursively) to ask the user if they will want the next direction, which will actually return the information to the user.

```
<action name="get_directions">
```

```

<one-of>
  <item>find navigation directions</item>
  <item>directions</item>
  <item>tell me how to get from <slot predicate="dep_city_pred"/> to <slot
predicate="dest_city_pred"/> by <slot predicate="transport_mode"/></item>
  <item>directions from <slot predicate="dep_city_pred"/> to <slot
predicate="dest_city_pred"/> by <slot predicate="transport_mode"/></item>
  <item>can you give me <slot predicate="transport_mode"/> directions from <slot
predicate="dep_city_pred"/> to <slot predicate="dest_city_pred"/></item>
</one-of>
</action>

```

We implement 'get_next_directions' predicate which is what actually retrieves/displays the information in the instructional dialogue. We also have a way of cancelling/stopping the instructional dialogue with 'back_to_menu':

```

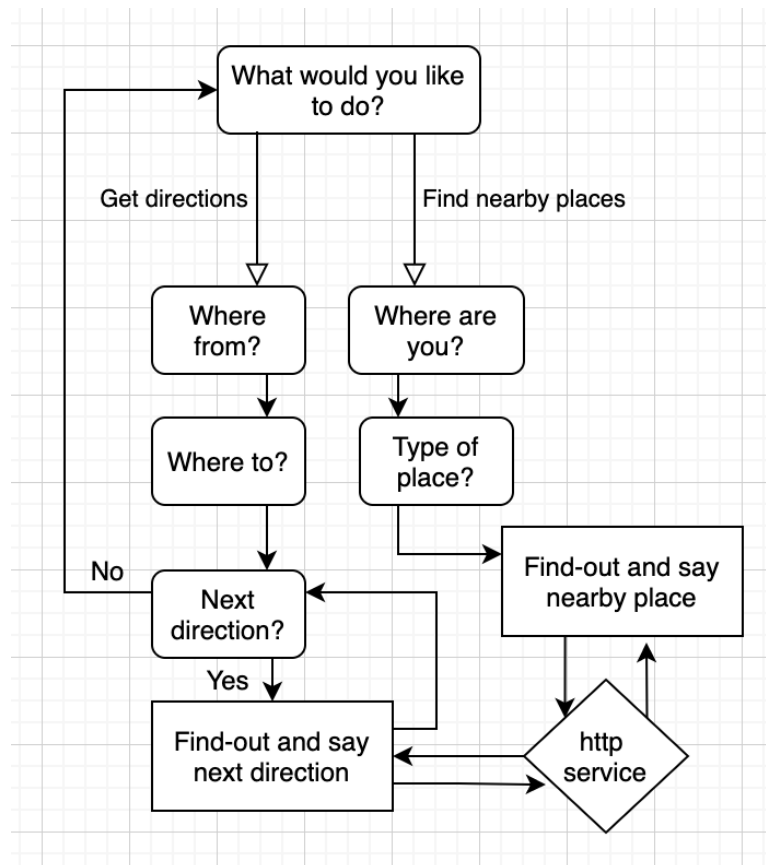
<question speaker="user" predicate="get_next_direction">
  <one-of>
    <item>yes</item>
    <item>continue</item>
    <item>next</item>
  </one-of>
</question>
<action name="back_to_menu">
  <one-of>
    <item>not</item>
    <item>no</item>
    <item>no thanks</item>
  </one-of>
</action>

```

There are also other actions, questions, answers and individuals in this file.

Domain:

The domain structures the logic of our dialogue. On the top level we ask what the user wants to do out of the two main options. We then get the information required from the user (for example the location where the user is etc.) and fulfil the request by using goals of 'resolve' type and the <invoke_service_query> tag. The general structure is as follows:



The 'get_directions' action works with the 'get_next_direction' resolve goal as a 'loop' to repeatedly get information (in the instructional dialogue).

Service Interface:

In the service interface, we have queries for 'get_locations' and 'get_next_direction'. These each have their own endpoint on our http-server. We send the user options to the service using a POST request at the corresponding endpoint, and get back a string, which is the answer for the user.

The http-server has implementations for interacting with the APIs which were described earlier and enable the system to get its information.

5. Sample dialogues

```
S> Do you want to find navigation directions or know about what is nearby?
U> what is a nearby italian restaurant
S> From which city?
U> copenhagen
S> The closest place matching the description is Ristorante Italiano at Fiolstræde 2, København K.
```

```
S> Do you want to find navigation directions or know about what is nearby?
U> what is nearby
S> From which city?
U> stockholm
S> What are you looking for?
U> a chinese restaurant
S> The closest place matching the description is Eat East at Fleminggatan 11, Stockholm.
```

```
S> Do you want to find navigation directions or know about what is nearby?
U> where is the closest pharmacy
S> From which city?
U> gothenburg
S> The closest place matching the description is Apoteket Centralen at 2, Göteborg.
U>
S> Returning to main menu. Do you want to find navigation directions or know about what is nearby?
```

```
S> Do you want to find navigation directions or know about what is nearby?
U> give me directions
S> To which city?
U> stockholm
S> From which city?
U> gothenburg
S> Do you want to know the next direction or not?
U> yes
S> Head west on Burggrevegatan toward Drottningtorget.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
U> yes
S> Continue onto Drottningtorget.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
U> please
S> Continue onto Drottningtorget.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
```

```
U> how do i get from stockholm to gothenburg by bicycling
S> Do you want to know the next direction or not?
U> yes
S> The next direction is: Head north on Gustav Adolfs torg toward Regeringsgatan.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
U> please
S> The next direction is: Turn right to stay on Gustav Adolfs torg.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
U> yes
S> The next direction is: Turn right onto Gustav Adolfs torg/Strömgatan.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
U> yes
S> The next direction is: Turn left onto Norrbro.
U>
S> Returning to find navigation directions. Do you want to know the next direction or not?
U> █
```

6. Discussion

On a high level, the system achieved the primary goals, which were to have a system that could be queried to find places and give navigation directions. There are however several compromises that had to be made due to the limitations of TDM as well as our time limitations.

One of the key ideas of this project was to build an instructional dialogue, which fit extremely well with the idea of getting directions in a mapping app. This came with some technical limitations. The main way for instructional dialogue to work with TDM (using the DDD) is by using the `<get_done>` tag. This works with a list of actions, one at a time. To use these, you need a preset number of steps which are hard-coded in the DDD, however this system would not work with such a design. The intention was to retrieve a (variable number of) steps/instructions from the backend service. When an implementation with the `<get_done>` tag and the `<if>` tag was used to design something that was 'loop-like', it didn't seem to work as intended, just looping eternally with no output, and a number of variations in this design (with and without the `<if>` tag) resulted in no functioning system. Perhaps there exists a functioning implementation similar to this, but in any case the lack of documentation and examples made it very difficult to find a solution.

The eventual working solution was a workaround using `<findout>` alternatives and resolving predicates. When the user asks for directions, the system goes to an intermediate `<goal>` which asks if the user wants to continue being instructed for directions, or to cancel. If the user selects to continue, it goes to a 'resolve' type `<goal>` which invokes the `<invoke_service_query>` to get and return the next direction (to tell the user). Then in the `<postplan>`, the predicate (which was just given to the user as an answer) is deleted by `<forget>`. It then returns to the previous step of asking the user if they want to hear the next direction again. If the user selects to cancel, it deletes all predicates with `<forget_all>` and returns to the main menu, otherwise if the user confirms, the loop repeats.

While this worked, it is a 'clunky' solution, and the system repeatedly asks the question if they would like to get the next direction repeatedly, quite unidiomatically.

Additionally, there was some error/fault in the system, that the service interface would store an old value of a predicate for a whole minute (this was tested), before it would retrieve a new value. Even if the value was 'deleted' (using the `<forget>` tag) in the domain, when the service query was invoked, the old value would be returned, and the http-service wouldn't even receive a request (again, unless an entire minute has elapsed). This is a significant limitation which wouldn't let this system work in practice.

When it came to the finding a nearby location, the system worked quite well. It was limited to the few locations that had been specifically coded in the system, so this is a limitation, and inputting addresses or other location types are not possible, but this could be implemented in the future if desired.

7. Future work

Finding a way to improve the instructional dialogue in a way that seems more idiomatic/natural would be ideal. Hopefully future improvements in TDM will allow this.

The system could be made more complete if it were possible to use the results from the nearby search as input for getting directions. That way the user could find where a place is and then navigate to it.

Using the current location of the user (retrieved from the system using some API) so that the user will not need to have to put in the effort of inputting their location, or so they can find places and navigate if they don't know where they are.

It would be useful to have address input, as well as other types of places, so that the system can navigate to anywhere that has an address.

8. Conclusion

Overall the system achieves the goals set out. There are many improvements to be made which have been identified.