Report of Dynamic Content Management Project
Student : Mengzi ZHAO

I – Introduction

The goal of this project is to realize a query engine to execute complex queries to obtain results. In this part, I will present the structure of my project (document and code) to make you understand clearer how I realize this project.

We need to choose an API to realize the project, I continue to use the musicbrainz version 1, because this is a highly use API and as the professor gave an example to use this API, so I choose this API and it will be easier to use it.

I realized 3 web services :

getArtistInfoByName which permits to obtain the id of artist, begin date and end date by entering the input of artist name

getArtistInfoByName which permits to obtain id of albums and name of albums of the artist by entering the id of artist, this input can be obtained by the previous query or given by user when it executes itself

getSongByAlbumId which permits to obtain the name of song and its duration by using the id of album that we can obtain from the previous query or we can give it an input when it executes itself

So I have 3 directories to contain result documents for these 3 web services :

mb_getSongByAlbumId, **mb_getArtistInfoByName**, **mb_getAlbumByArtistId**

every directory contains 2 directory call_result and transf_result, call_result contains the documents that I obtain from the web service and transf_result contains the result obtained from the web service after transformation. There is also a directory ws-definitions which contains the documents xml (web service description) and xsl (transformation function).

For the source code, except code document offered by professor, I add some classes :

QueryEngine : contains the code to realize the 3 web services, output the result and generate the corresponding file

Album, **Person**, **Database** : generate the result and a table to realize the join of result of single queries in the complex query

II – Realization of project

At first, I need to go to the website of musicbrainz to see the structure of information to know which parameters that I can use in the query and to construct the xml and xsl document according to the structure of data.

I take the query given in the document and change some parameters after checking on the musicbrainz website.

```
String arg = " P(?title,?duration) <- getArtistInfoByName(?name=WANG, ?id, ?b, ?e)#
getAlbumByArtistId(?id, ?aid, ?albumName)#getSongByAlbumId(?aid, ?title, ?duration)";
```

My goal is to generate this complex query who contains 3 single query. For every web service, we need to create a xml and xsl following the principle in the Project.pdf and the structure of data on the musicbrainz website.

After having these xml and xsl documents, I create 3 functions for these 3 web services **ws_getArtistInfoByName**, **ws_getAlbumByArtistId**, **ws_getSongByAlbumId** for the 3 single query in the complex query. This 3 single query can be executed independently or execute 2 together or 3 together.

Before doing call by web service, I create a function checkQuery who permits to check the form of the query and in the other function like getOutput etc there are also some codes who permit to check the syntax of the query to avoid if there are some outputs which do not exist in the query or the web service written in the query does not exist etc, because when the query is not in a good form, this will influence the function of the code.

At first I create the code to realize the simplest part : generate these 3 single query independently, in this case, we need to think about the input for every single query, because if there is no input and there is no result. The function **getInputValue(String parsedQueryPart)** is to return the value of input of the query, so we can use this function to obtain the input and then use it to call the result by using the web service. I create the function **getHead(String parsedQueryPart)** which returns the head name of the single query, the parameter of this function is the single query in the format string, this is to help to choose the correspondent web service for the single query. In this simplest case, for the 3 web service functions, we need just to use the **WebServiceDescription.loadDescription** to load the web service description and **getCallResult** to obtain the result and **getTransformationResult** to transform the result obtained from the previous step.

The case to generate 2 queries together and the case to generate 3 queries together are the same principle.
Let us look at the query :

```
String arg = " P(?title,?duration) <- getArtistInfoByName(?name=WANG, ?id, ?b, ?e)#
getAlbumByArtistId(?id, ?aid, ?albumName)#getSongByAlbumId(?aid, ?title, ?duration)";
```

When we have a complex query, we need to split it in several single queries, so I create a function **parseQuery** to split the query by "<-"at first then by"#"to get the single queries. After splitting, we can obtain 3 single queries.
We need to use the id obtained by the first single query then put it in the second single query, and take the aid obtained from the second query as the input of the third query. In this case, we need to stock results of every single query that we can use for the next single query, so I create the **class Person** and I create a list containing the object of type Person to stock results of the first single query, and when the second single query is executed, we can take input values from the list and stock results of the second query in a list of objects of type Album and I use a list «resultList » to combine the result of the 2 queries, for the third single query, we can use the same principle.

III - Experiments

I tested with different type of queries, for example, I execute every one single query or any 2 single queries or 3 queries together. I will show you the result that I obtained by executing the 3 single queries together.

```
String arg = " P(?title,?duration) <- getArtistInfoByName(?name=WANG, ?id, ?b, ?e)#
getAlbumByArtistId(?id, ?aid, ?albumName)#getSongByAlbumId(?aid, ?title, ?duration)";
```

Then we use the function parseQuery to split the complex query, we can obtain :

getArtistInfoByName(?name=WANG, ?id, ?b, ?e)

getAlbumByArtistId(?id, ?aid, ?albumName)

getSongByAlbumId(?aid, ?title, ?duration)

Then we use the 3 web services to execute this 3 single queries, at first, we can obtain a table combining the results of these 3 single queries :

```
[王嘉尔, 993b5a29-a8f1-49d7-9ef3-e05233041454, 1994-03-28, , 3ad990f5-c054-4f82-b9b6-8f57d6fcb76b, Papillon, Papillon, 195000]
[王嘉尔, 993b5a29-a8f1-49d7-9ef3-e05233041454, 1994-03-28, , 71c6a43a-aaf8-4682-8144-e4b6315d1205, Generation 2, Generation 2, 171000]
[王嘉尔, 993b5a29-a8f1-49d7-9ef3-e05233041454, 1994-03-28, , 3ad990f5-c054-4f82-b9b6-8f57d6fcb76b, Papillon, Papillon, 195000]
[王嘉尔, 993b5a29-a8f1-49d7-9ef3-e05233041454, 1994-03-28, , 71c6a43a-aaf8-4682-8144-e4b6315d1205, Generation 2, Generation 2, 171000]
[王嘉尔, 993b5a29-a8f1-49d7-9ef3-e05233041454, 1994-03-28, , 3426a58d-adda-44b7-8886-09db1100035b, OKAY, OKAY, 192000]
[王嘉尔, 993b5a29-a8f1-49d7-9ef3-e05233041454, 1994-03-28, , 3426a58d-adda-44b7-8886-09db1100035b, OKAY, OKAY, 192000]
[Kerry JK, 6db972b2-ad5f-423a-b8ac-8fcef27a5d21, 1977-02-01, , 83cd5d20-d07d-4bfe-b80d-9bd45a82add4, I Am, I Am, 239000]
[Kerry JK, 6db972b2-ad5f-423a-b8ac-8fcef27a5d21, 1977-02-01, , 83cd5d20-d07d-4bfe-b80d-9bd45a82add4, I Am, The Possible Police, 23100
```

As the final output that I need is just the title and the duration according to P(?title,?duration), so I do the projection and then I obtained :

```
Appendice, 100000
All Alone, 217573
Bernadette, 230000
Love Man, 218840
The Devil in Me, 226000
Hope for the Future (Where Did You Go?), 230893
All Alone, 217573
Looking So Hard for Love, 179000
```

During this project, I meet some difficulties, the most difficult part of this project is to combine the result of single queries, because every time, the parameters of output are different, so it is a little difficult to put them together. And at the beginning, when I try to obtain the results, sometimes there are errors to download the data, I think this is because every loop executes too quick, so I make 500ms sleep time for every loop, and now this problem doesn't appear.

The shortcoming of my code is that I used too many variables to do the combinations of result, and every time the code need to use time to do another time for the combinations of results in another table.