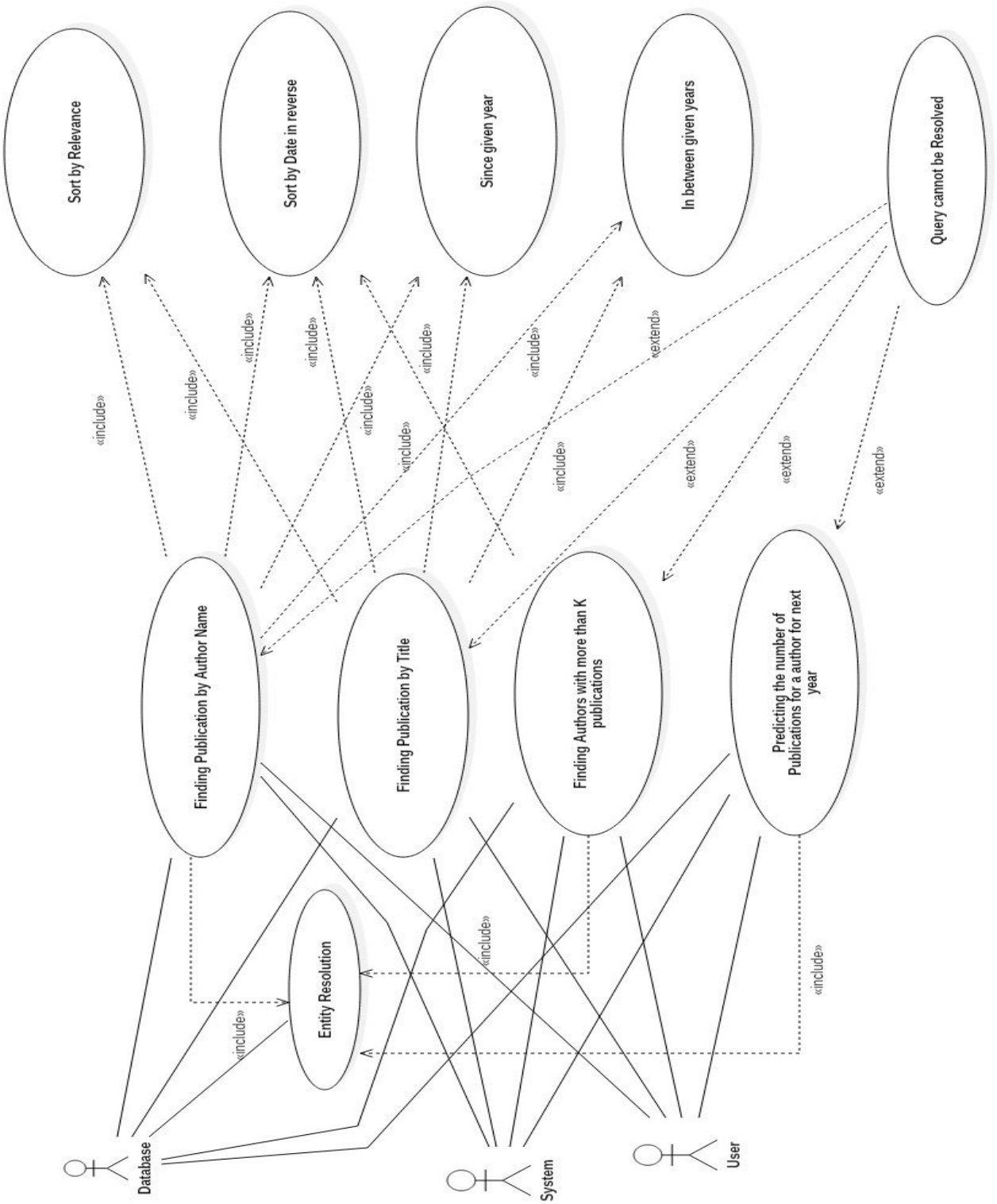
DBLP Project

-Saksham Suri 2015082

-Tushar Kataria 2015082

Use Case Diagram



Use Case Descriptions (Lab-7) DBLP

1) **Name:** Entity Resolution
   Participating actor: Database
   Entry condition: The database has been prepared and stored in a collection.
   Exit condition: Entity resolution has been done and clusters of different authors have been created having publications for that author.
   Event flow:
   1. The whole database is traversed.
   2. For every author name we check whether it is similar any existing author names using <www> tag.
   3. If yes then that object is added to the cluster of that author.
   4. If not a new item in the set is created for the new author and the specific object is added to the cluster for that author.
   Exceptions: None

2) **Name:** Finding publications by author name
   Participating actor: Database, User, System.
   Entry condition: The database has been prepared and stored in a collection.
   Exit condition: All publications have for a given author name have been found.
   Event flow:
   1. Entity resolution is done on the whole database.
   2. For a given author name publications corresponding to the cluster of that author are separated and returned.
   Exceptions: The input is not as per the requirement, no author matching the input author name exists.

3) **Name:** Finding publications by title
   Participating actor: Database, User, System.
   Entry condition: The database has been prepared and stored in a collection.
   Exit condition: All publications corresponding to the given title have been found
   Event flow:
   1. For a given title, publications matching the given title are retrieved and returned.
   Exceptions: The input is not as per the requirement, no title matching the input title exists.

4) **Name:** Finding authors with more than K publications
   Participating actor: Database, User, System.
   Entry condition: The database has been prepared and stored in a collection.
   Exit condition: All authors having more than k publications have been found
   Event flow:
   1. For a given value of k author names of authors having more than k publications is returned.
   Exceptions: The input is not as per the requirement, no author has more than k publications.

5) **Name:** Predicting number of publications for an author for a specified year
   Participating actor: Database, User, System.
   Entry condition: The database has been prepared and stored in a collection.
   Exit condition: The number of publications for the next year have been predicted.
   Event flow:
   1. For a given author, using the data up to a particular year the number of publications by that author for the next year are predicted using prediction algorithms.

Exceptions: The input is not as per the requirement, no author has more than k publications.

6) **Name:** Query not found/Inappropriate query
Participating actor: Database,User
Entry condition: The database has been loaded onto the memory and user enters a query to be searched.
Exit condition: Search query returned no results.
Event flow:
For a given input, all records have been searched and no record matched the input query as either input was not appropriate or no matching result existed.

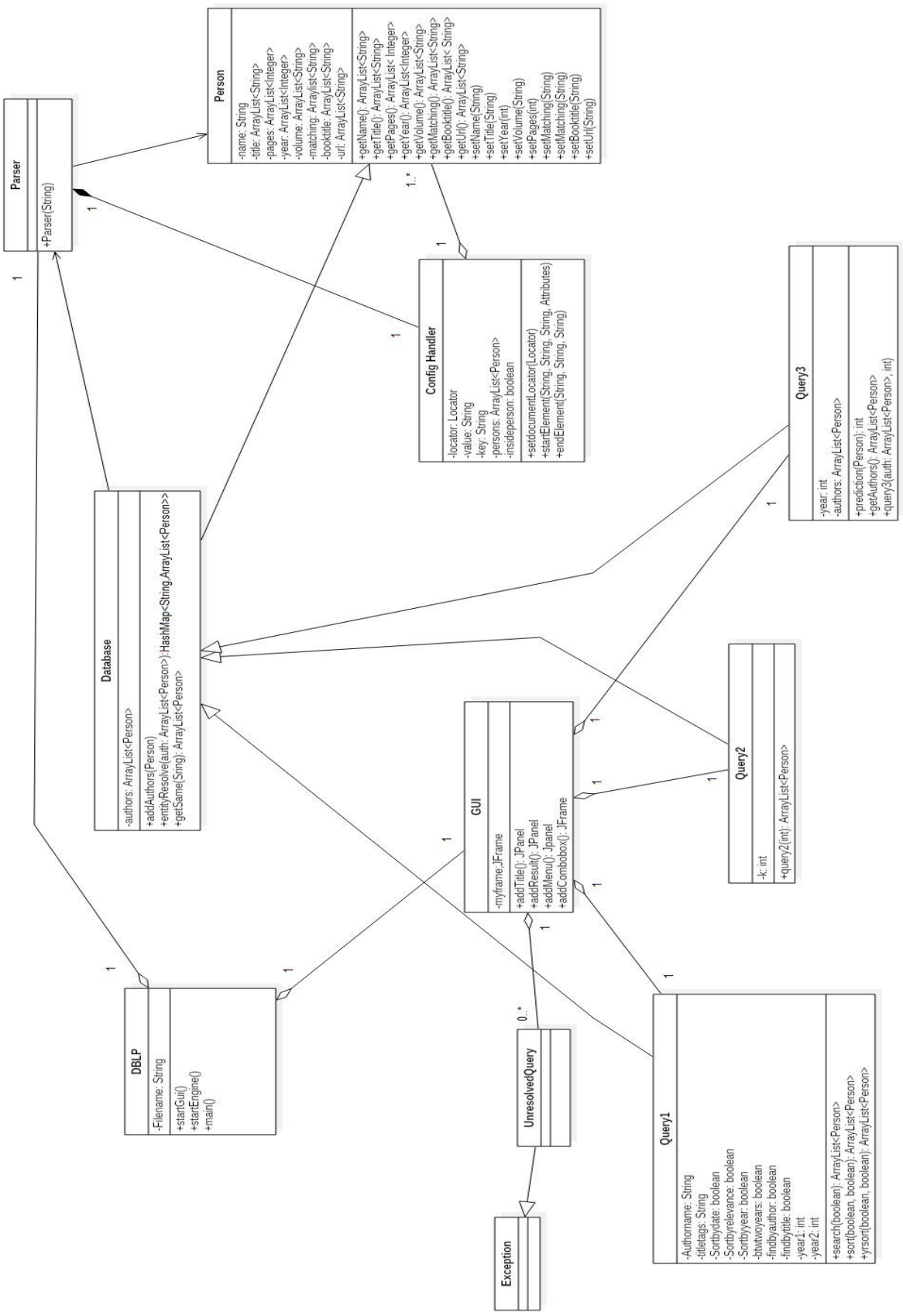
7) **Name:** Sort by relevance
Participating actor: Database
Entry condition: The database has been loaded onto the memory and user enters a query to be searched.
Exit condition: Search query returned no exception and the returned results have been sorted by relevance.
Event flow:
For a given input, all records have been searched and the matching results are sorted by their relevance in terms of matching number of words.

8) **Name:** Sort by date
Participating actor: Database
Entry condition: The database has been loaded onto the memory and user enters a query to be searched.
Exit condition: Search query returned no exception and the returned results have been sorted by date in reverse.
Event flow:
For the given input the result is obtained by sorting the input by date in reverse order.

9) **Name:** Return results since a particular year
Participating actor: Database
Entry condition: The database has been loaded onto the memory and user enters a query to be searched.
Exit condition: Search query returned no exception and the returned results have been filtered since after the given year.
Event flow:
For a given input, all records have been searched and the matching results have been filtered out based on given year i.e., only results having year greater than input year are returned.

10) **Name:** Return results between 2 years
Participating actor: Database
Entry condition: The database has been loaded onto the memory and user enters a query to be searched.
Exit condition: Search query returned no exception and the returned results have been filtered to be between 2 particular years.
Event flow:
The input is filtered to only obtain the results which have date in between the given years.

Class Diagram



Class Diagram Description

1)DBLP:

Variables:

a)Filename-Name of the database file(In our case dblp.xml)

Methods:

a)startGui-It initializes the GUI and makes it ready for user input for the type of query.

b)startEngine-It starts the database parsing and reading to get the database in the desired form.

c)main-The main method from where the query engine is started and the initializer functions are called.

2) GUI:

Variables:

a)myframe-It is the main frame which displays the content on the GUI.

Methods:

a)addTitle-Adds the title to the frame currently being viewed

b)addResult-Displays the result on the GUI

c)addMenu-Displays the menu to the user for choosing and entering the data for queries

d)addCombobox-Adds the Combo Box for different drop down lists to choose between different options.

3)UresolvedQuery: This is class for user defined exceptions. It is used to throw exception when either the input is not correct or no result is found.

4)Exception: It is the inbuilt exception class available in java.lang library.

5)Query1:

Variables:

a)Authorname-Stores the name of author

b)titletag-Stores the title

c)Sortbydate,Sortbyrelevance,Sortbyyear,Sortbtwtwoyears,findbyauthor,findbytitle-They are all Boolean variables which are true according to the type of query and according to how it is to be sorted.

d)year1,year2-Store the lower and upper value of years between whom the query result is to be shown.

Methods:

a)search-It searches the database for matches according to the title or author whichever is required.

b)sort-It sorts the result obtained from search according to the type of sorting required(according to date,relevance)

c)yrsort-It filters the result obtained from search to either get the results corresponding to a range of years or publications after a given year.

6)Query2:

Variables:

a)k-It stores the threshold value above which if an author has publications his name is displayed.

Methods:

a)query2-It obtains an ArrayList of authors having more than k publications.

7)Query3:

Variables:

a)year-It stores the year upto which the data can be used

b)authors-It is an ArrayList of authors for whom the predictions are to be made

Methods:

a)predictions-It predicts the number of publications an author will do using the data upto 1 year before.

b)getAuthors-It returns an ArrayList of all authors for whom prediction is being done

c)query3-It is basically calls the prediction function for each author under consideration.

8)Person:

Attributes:

a)name:Name of the author

b)Title:ArrayList of titles of publications by that Author

c)pages:ArrayList of no of pages in publications of that author

d)Year:ArrayList of year of publishings by that author

e)Volume:ArrayList of volume of publications by that author

f)matching:Contains synonyms/homonyms of person name

g)booktitle:Contains booktitles of publication by that author

h)url: Contains url of publications by that author

Methods:

a)getName():gets the name of person

b)getTitle():gets the ArrayList of titles of publications by that person

c)getPages():gets the ArrayList of pages of publications by that person

d)getYear():gets the ArrayList of years of publications by that person

e)getVolume():gets the ArrayList of volumes of publications by that person

f)getMatching:It returns an ArrayList of String containing the different names used by the that particular author whose person object it is.

g)getBookTitle():gets the ArrayList of booktitles of publications by that person

h)geturl():gets the ArrayList of urls of publications by that author

i) setName(String):sets the name of person

j) setTitle():adds the title of publication to that persons object

k) setPages():adds the no. pages of publication to that persons object

l) setYear():adds the year of publication to that persons object

m) setVolume():adds the volume of publication to that persons object

n) setMatching(ArrayList<String>):Performs entity resolution and creates an ArrayLIst of String containing different names of the author of that particular publication.

o) setBookTitle():adds the booktitle to that persons object

p) seturl():adds the url to that persons object

9)Parser

Method:

a)Parser(String):Parses the File name provided.

10)ConfigHandler

Attributes:

a)locator:Used to locate by parser

b)value:Value of key value pair

c)key:Key of key value pair

d)persons:Array List of Persons

e)insideperson:Denotes whether co-authors are present or not

Methods:

a)SetdocumentLocator(Locator):Sets the Locator at a particular positon in the dblp database

b)startElement(String,String,String,Attributes):Used by parser to determine start tag

c)endElement(String,String,String):Used by parser to determine end tag

11)Database

Attributes:

a)Authors:ArrayList of all Authors

Methods:

a)addAuthors(Person):Adds a new author if he/she does not already exist in the authors ArrayList

b)entityResolve(ArrayList<Person>):Creates clusters and groups the same author having a little different names together.

c)getSame(String):Return an ArrayList containing all authors having slightly different names but of the same author.