Abstract

Class diagram and associated class descriptions & Sequence diagram depicting system functionality

Hari Vikas Janarthanan – [Janarthanan.h@husky.neu.edu](mailto:Janarthanan.h@husky.neu.edu)

Rohit Kumar – [kumar.ro@husky.neu.edu](mailto:kumar.ro@husky.neu.edu)

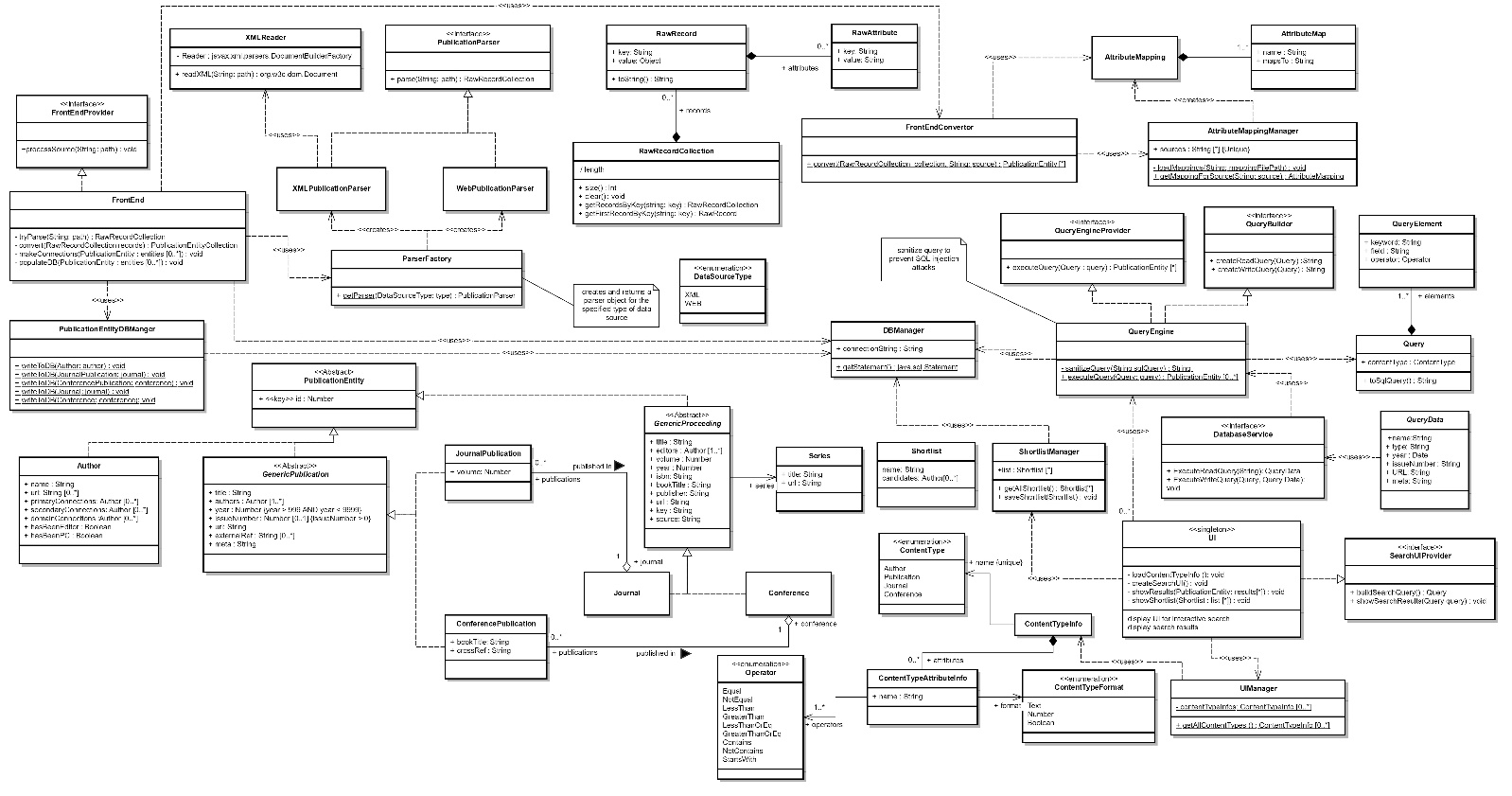
Yuzhi Ma – [ma.yuz@husky.neu.edu](mailto:ma.yuz@husky.neu.edu)

cs5500-spring17-team1

System UML Design

V1.2

# Class Diagram



[View Diagram Online](https://cs5500.ccs.neu.edu/confluence/x/FIDx)

# Interface Definitions

|  |  |  |  |
| --- | --- | --- | --- |
| **FrontEndProvider** | Specifies behavior for analyzing, interpreting and storing data from a data source   |  |  | | --- | --- | | processSource(String: path) : void | Takes the path to the data source as input. Parses, converts and stores the data in the database | |
| **PublicationParser** | Specifies behavior for parsing a data source   |  |  | | --- | --- | | parse(String: path) : RawRecordCollection | Takes the path to the data source as input. Parses the data source and returns the representation as a RawRecordCollection | |
| **QueryEngineProvider** | Specifies behavior for retrieving search results from the database for a given query   |  |  | | --- | --- | | executeQuery(Query: query) : PublicationEntity [\*] | Retrieve search results from the database and return it as a collection of Publication entity objects | |
| **QueryBuilder** | Creates a query based on the search criteria formulated from the frontEnd.   |  |  | | --- | --- | | createReadQuery(String[] : queryparams) : String  createWriteQuery(String[] : queryparams) : String | Create an SQL query to read data from databse and return in the string format  Create an SQL query to write data and return in the string format | |
| **SearchUIProvider** | Specifies behavior for creating search query representation and communicating with the QueryEngine   |  |  | | --- | --- | | buildSearchQuery() : Query | Build Query object that represents the search query specified by the user in the UI window | | showSearchResults(PublicationEntity: entities[\*]) : void | Displays the collection of PublicationEnitity objects to the user | |

# Data Model

|  |  |
| --- | --- |
| **PublicationEntity**  **<<abstract>>** | Models the parent of all data model types related to publication |
| **Author** | Models the attributes of an author of a publication |
| **GenericPublication**  **<<abstract>>** | Models the common attributes of a publication |
| **JournalPublication** | Models the publication in a Journal |
| **ConferencePublication** | Models the publication in a Conference |
| **GenericProceeding**  **<<abstract>>** | Models the common attributes of a Journal and Conference |
| **Journal** | Models a Journal. A Journal has a collection of JournalPublication |
| **Conference** | Models a Conference. A Conference has a collection of ConferencePublication |
| **Series** | Models the attributes of a Series associated with a Conference or Journal |
| **ContentType** | Models the types of distinct Publication related Content |
| **ContentTypeInfo** | Models the attributes of a content type. |
| **ContentTypeAttributeInfo** | Models the properties of an attribute of a content type |
| **Operator**  **<<enumeration>>** | Models the various operations that can be performed on a content type attribute |
| **ContentTypeFormat** | Models the format of data that the attribute holds |
| **ContentTypeInfo** | Models the attributes of a content type |

# Font-end related Class Definitions

|  |  |
| --- | --- |
| **FrontEnd** | Models the functionality of the front-end component that analyzes the data source and converts it to a representation suitable for querying and stores the representation in the database |
| **RawRecord** | Models a key value pair that represents a single information in original data source |
| **RawAttribute** | Models a key value pair that represents meta data about a RawRecord object |
| **RawRecordCollection** | Defines a collection of RawAttribute |
| **AttributeMap** | Models a mapping between a data source attribute and a PublicationEntity attribute |
| **AttributeMapping** | Collection of AttributeMap for a data source |
| **AttributeMappingManager** | Manages mappings for all data sources |
| **FrontEndConvertor** | Models conversion of RawRecord object to a PublicationEntity object |
| **XMLPublicationParser** | Models parsing of a data source in XML format |
| **WebPublicationParser** | Models parsing of a data source in a web / HTML format |
| **XMLReader** | Models functionality for handling XML files |
| **ParserFactory** | Creates an appropriate parser object for the given data source type |
| **PublicationEntityDBManager** | Models functionality to store a PublicationEntity object to the corresponding table in the database |
| **DataSourceType**  **<<enumeration>>** | Modes the types of data source formats |

# Query Engine related Class Definitions

|  |  |
| --- | --- |
| **QueryEngine** | This is the main driver class for QueryEngine. It receives the query parameters in the form of query elements and performs necessary steps to fetch the result from the database |
| **Query** | Query object that contains the details about the query params. It is used by query builder to form a query |
| **QueryElement** | This is individual elements containing information about a filter or search text entered by user |
| **DatabaseService** | Used as bridge between the database and the Query engine. This class makes all the database search calls and sends the results back via the query engine. |
| **QueryData** | The Database results are formatted in the form of queryData object. This acts the object trealtion between database entities and the QueryEngine data object |
| **QueryBuilder** | Formulates the query based on the query object passed from the User-Interface. Validation and creation of different kinds of queries is performed by this class |

# UI related Class Definitions

|  |  |
| --- | --- |
| **UI** | Models the functionality that enables the user to search for publications |
| **UIManager** | Provides information on data models to generate UI elements correspondingly |
| **ShortList** | Models a list of potential candidates |
| **ShortListManager** | Models functionality to manage short lists |

# Database related Class Definitions

|  |  |
| --- | --- |
| **DBManager** | Handles connection to Database and provides a gateway for other components to access the database |

# Sequence Diagram

