QueryHek and JSONParser Program Functional Requirements

Montana State University

Data Mining Lab

Table of Contents

Introduction

The QueryHek program will allow users to download events reported to HEK and it will generate a output file in XML format or JSON format. The XmlParser program will process XML format files(that is generated by QueryHek program), and it will generate a tab-delimited output file. The JsonParser program will process JSON format files(that is generated by QueryHek program), and it will generate a tab-delimited output file.

Context Diagram of QueryHek and JsonParser program

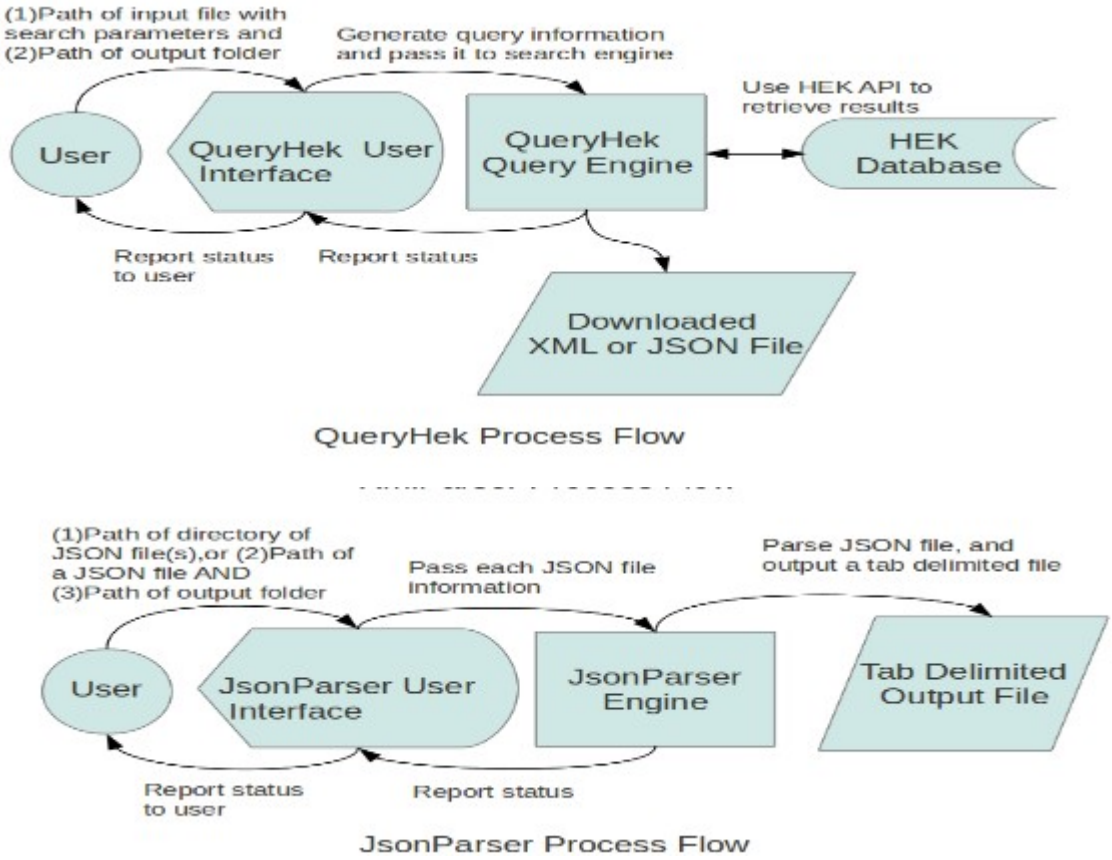

Fig1. Context diagram of the system

Requirements of QueryHek program

1. The input arguments to the program are an input text file and path to an output folder, where downloaded file will be saved.
2. Users can use this program to download only one event type (eg: sigmoids or filaments) at a time.
3. The program will generate one output file that aggregates results for the given search conditions

(provided records are found).

4. The program will use HEK API to download the data in XML format.
5. By default, HEK API returns all the required and optional parameters (attributes) for an event type (sorted by reverse chronological order, and it is based on event start time), and thus this program will return the results in the order HEK API returns the results.
6. For details about the required and optional parameters (attributes) for each event type, please use this link "http://www.lmsal.com/helio-informatics/hpkb/VOEvent_Spec.html ". **For few event types, this link does not provide all the parameters (attributes) returned by HEK API though (example: Sigmoid).**
7. Format of input file.
    1. It has six mandatory "Parameter-Value" pairs followed by optional "Parameter- Operator, Value" triplet.
    2. The first six parameters are Start Date, Start Time, End Date, End Time, Event Type, Spatial Region. Format for each is "Parameter-Value" and a newline at the end. Currently, we have the following Event Types, available from HEK, Sigmoid (sg), Filaments (fi), Flares (fl), ActiveRegion (ar), CoronalHole (ch), EmergingFlux (ef). For Spatial Region parameter currently helioprojective (from Earth's perspective, in arcseconds from disk center), Stonyhurst (longitude measured from central meridian and latitude in degrees) and Carrington (longitude and latitude in degrees) are supported. Let (x1,y1) and (x2,y2) be the lower-left and upper-right coordinates of a bounding box respectively. Format of Spatial Region value is "region, x1, x2, y1, y2" (note: Here region should be helioprojective or stonyhurst or carrington). An example of input (file) is given below.

    Start Date : 2011-01-01

    Start Time : 00-00-00

    End Date : 2011-01-30

    End Time : 23-59-59

    Event Type : sg

    Spatial Region : helioprojective, -5000,5000,-5000,5000

    (Note: The parameter values and operators of input file will be automatically URL encoded (with encoding scheme "UTF-8") by the program).

    3. Optional filter conditions, can be used to further filter search results. Format for each is "Parameter – Operator, Value"and a newline at the end. At present the following operators are supported by HEK API, =, !=, >, <, >=, <= and **like**. For more information about filter parameters, please use this link, "http://vso.stanford.edu/hekwiki/ApplicationProgrammingInterface?action=print#head-4505d658c207b707054a1c76a53a9d8ca3778d5d". Also for a complete list of all parameter, descriptions and the corresponding event types, please use this following link, "http://www.lmsal.com/hek/VOEvent_Spec.html". An example of input (file) is given below,

    Start Date : 2011-01-01

Start Time : 00-00-00

End Date : 2011-01-30

End Time : 23-59-59

Event Type : sg

Spatial Region : helioprojective, -5000,5000,-5000,5000

FRM_Contact :=, Nour.Eddine.Raouafi@jhuapl.edu;manolis.georgoulis@academyofathens.gr

FRM_Name : =, Sigmoid Sniffer

OBS_ChannelID : =, 94_THIN

Note : Start Date and Start Time specifies beginning of event start time, and End Date and End Time specifies end of event start time. So events in HEK will be retrieved based on event_starttime >= "Start Date" + "Start Time" and event_starttime < "End Date" + "End Time". Also, for the returned events, event_endtime can be outside this time window (beyond "End Date" + "End Time"). See Fig2, for events that will be retrieved for the given query window.

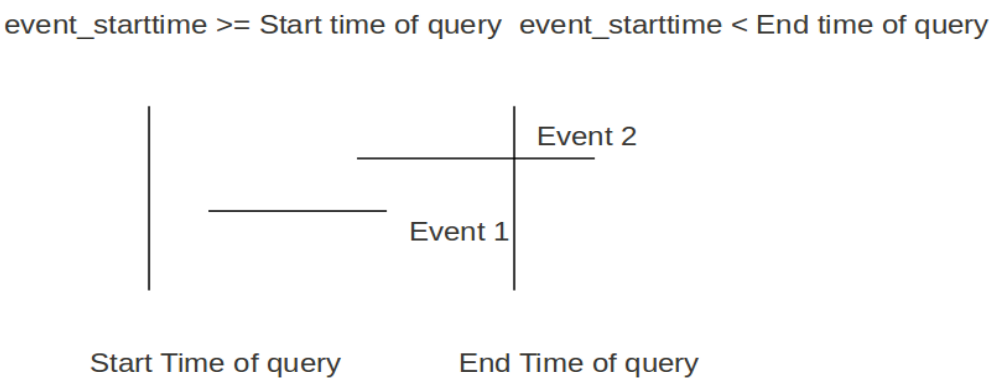

Fig2: Query time window and events that will be retrieved

Requirements of JsonParser program

1. The input arguments to the program are a JSON formatted input file or a directory that will be recursively searched for JSON formatted files or a directory that will be searched for JSON formatted files and path to an output folder, where a tab-delimited file(s) for each JSON file will be saved.
2. The program will parse, JSON formatted file(s) (file(s) generated by QueryHek program)and generate a tab-delimited output file(s).
3. The tab-delimited output file, will have a header row that gives the column names and the remaining rows will have values for each of the column names.

4. The order of columns generated will be in the order that is found in the JSON formatted file.
5. Columns that have same names under a parameter will be tagged with identifiers (count) at the end, in the header row of the generated tab-delimited output file. For example, in JSON formatted file, under “refs” parameter we will have repetitive structure representing data. An example is "refs":[{"ref_name": "Module Image","ref_type": "image","ref_url": "https://www.lmsal.com/eds-aux/podimages/2011/12/14/pod_autosubmission_eds_2011-12-14T16%3A12%3A38.908/AIA_20111011_0432.png"},{"ref_name": "FRM_URL","ref_type": "unknown","ref_url": "http://solar.physics.montana.edu/sol_phys/fft/"}]}]. In the generated output file, corresponding values in header row will be of the form ref_name_1, ref_type_1, ref_url_1, ref_name_2, ref_type_1, and ref_url_1.
6. If any record in the file has columns not equal to the first record it will be treated as a bad record and written to a separate file.

Requirements of ExtractAttributes program

1. A ExtractAttributes takes a path to tab delimited input file (including file name), path to input configuration file (including file name) and path to output folder and generates a tab delimited file with columns specified in the configuration file.
2. Config file should be a text file with column names to be extracted in each line.
3. Output is a tab delimited file in the output folder by name "Extracted-Attributes.txt".
4. If column information in config file is not found in input file, the program will stop.

Technical Details

These programs will be available in JAVA.