Crimson Publishers
Wings to the Research

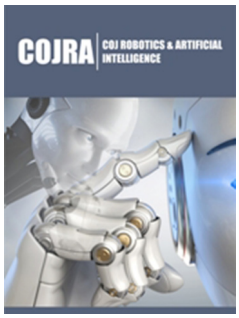
Mini Review

Development of a Software Complex for Studying the Life Cycle of a Stream of Internet Memes

Gurzhiy PA and Kozlova MG*

V I Vernadsky Crimean Federal University, Development of Computer Science, Russia

ISSN: 2832-4463



***Corresponding author:** Kozlova MG, VI Vernadsky Crimean Federal University, Development of Computer Science, Russia

Submission: March 29, 2022

Published: April 18, 2022

Volume 1 - Issue 5

How to cite this article: Gurzhiy PA and Kozlova MG*. Development of a Software Complex for Studying the Life Cycle of a Stream of Internet Memes. COJ Rob Artificial Intel. 1(5). COJRA. 000525. 2022. DOI: 10.31031/COJRA.2022.01.000525

Abstract

The aim of the work is to create a software package for collecting information and studying the process of distribution of Internet memes using modern development tools. The work consists of the basic provisions of the subject area, testing and application development for automated work with Internet memes. The result is a developed set of cross-platform applications (including web and server applications) designed for scientific, sociological research on Internet memes.

Keywords: Internet meme; Lifecycle; Software development; Web application; Process automation tools; Web parsing; API; JSON; Data stream processing; Image; HTTP request; C#; Python; JS; .NET Core

Introduction

Today, Internet memes are increasingly flooding the information and media environment. Thousands of memes appear every day, some become irrelevant, while others begin to gain popularity, thus forming a certain cycle of "life". According to research by sociologists, these changes may reflect a person's reaction to current events in the world, society or a narrow circle of communication. That is why experts in the field of sociology are extremely interested in studying such a reaction of society. The first version of the software package for working on the analysis of the life cycle of Internet memes was developed two years ago. From that moment on, we continued to work together with a team of sociologists. In the course of this process, feedback was received on the software package [1]. Taking into account all the positive and negative comments, as well as the structural change in the main task of the complex, it was decided to create a second version. The purpose of the work is to create a software package for analyzing the flow of Internet memes. This complex should include components for the automatic collection, structuring and analysis of Internet memes. The relevance of the work is justified by the presence of demand in this software by a group of sociologists to conduct research in the field of information dissemination in the media space through Internet memes.

Determination of requirements and tasks of the software package based on feedback

Previously, we developed a software package (TagRun), which provided part of the functionality necessary for the work of expert sociologists. TagRun coped with the tasks set at that time. However, during the research it became clear that the complex needs to be improved and new functionality added. The first problem faced by experts is the large amount of information for classification. Regular search for current Internet memes through search engines produces more than 100 image results for each query. For further work, each of them should be marked with at least several tags. If it takes 5 minutes to classify one image, then processing all the search query results will take more than 8 hours. The second problem that

became obvious while working with the complex is the updating of search queries [2]. In the previous implementation of the complex, experts were forced to independently add and remove search queries for which Internet memes were collected. To determine the current topic on which new Internet memes could appear, experts analyzed the weekly news summary. The work on collecting and grouping news headlines was quite resource-intensive, so it was decided to automate the process of analyzing news in order to highlight relevant topics. The third problem was the identification of duplicates among Internet memes. When collecting information, visually similar images were obtained from various sources, which are the same Internet memes. To combine several such research objects into one, manual control of an expert is necessary. The fourth problem is the limited functionality of the complex for filtering, grouping and exporting the collected information about Internet memes. At first, it was assumed that there would be enough expert requests for tags to select Internet memes. However, in the course of working with TagRun, it became obvious that in addition to filtering by tags, additional criteria were needed, such as: search dates, source resource, number of appearances, initial request. Also, to analyze the results outside the complex, a way to export the information received was needed. Thus, the main requirements and tasks that the new software package solves were highlighted:

a. the ability to add, delete and edit search queries,

b. automatic generation of search queries based on the analysis of news resources,

c. collecting information about images based on search queries,

d. automatic classification and grouping of collected information,

e. storage of various meta information about the image,

f. flexible configuration of filters to filter the general flow of information in order to identify Internet memes,

g. exporting information in a convenient format,

h. availability of interfaces for further expansion and/or modification of the complex.

Design of the complex and components

The structure of the complex, as in the first version, is modular, where each module solves its own task. Based on the requirements listed above, we will determine what each component serves for:

a. Database - for storing information,

b. File storage - for saving images,

c. Web applications - for working with the complex,

d. A program for collecting information on search queries,

e. Program for analyzing news headlines and generating queries,

f. Service for detecting duplicate images,

g. The server responsible for the communication of the other components.

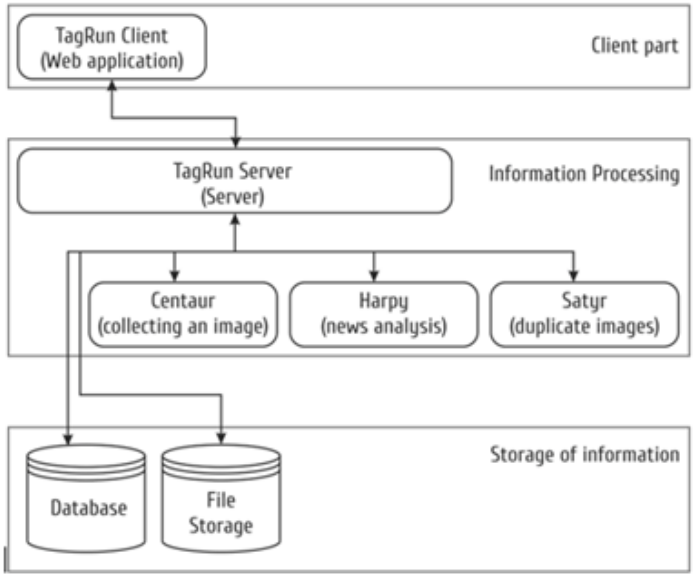


Figure 1: Software structure.

To store information about images, monitoring requests, monitoring results, as well as tags and timestamps, a database is needed. Since there are a significant number of relationships between storage objects, it makes sense to use a relational database. However, the database provides storage only of information about objects and their relationships but does not provide storage of image files. Therefore, a storage service is needed for uploaded files [3]. To determine the similarity of images (finding duplicates), it is necessary to hash these images in a special way. The image hashing service takes over this task. In this case, hashing is understood as

an algorithm for creating hashes with the feature that the hashes of images with minimal differences should be as close as possible in a certain metric. To fill the database with up-to-date information, the Centaur automatic collector is used. The decision to make the logic of assembling information on search queries and reverse image search into a separate program is due to the fact that the collection algorithm changes depending on the resources from which the collection takes place. Since the format of these resources is not constant, the program may need frequent changes (Figure 1). The database is the most important component of the complex, as it stores all the results of the complex. Since the first version, the database has undergone some structural changes to meet the new requirements (Figure 2). The following entities are defined in the database:

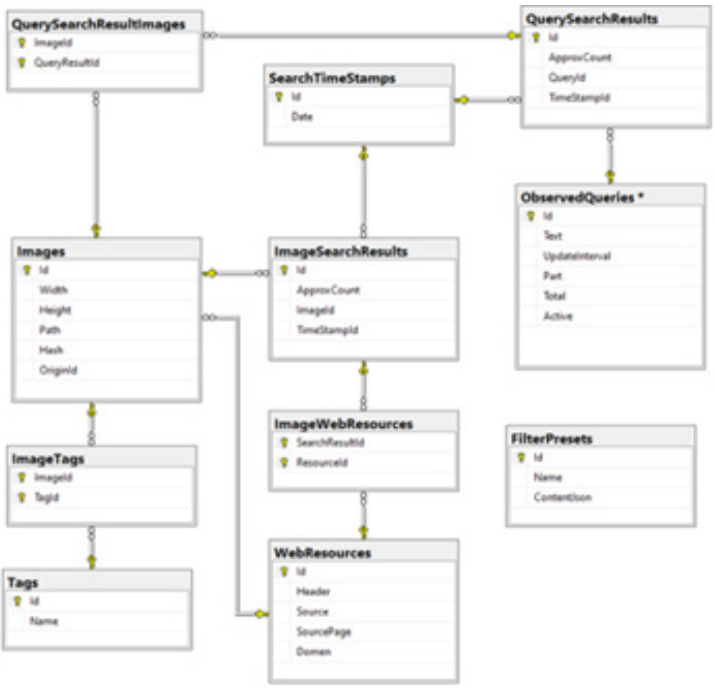


Figure 2: Database structure.

a. Observed Queries (Monitoring requests)

b. Search Timestamps (Search Dates)

c. Images (Images)

d. Tags (Tags)

e. Web Resources (Links to web resources)

f. Query Search Results (Search results for the query)

g. Image Search Results (Reverse image search results)

h. Filter Presets (Filter templates).

Development of software package components

The main component of the complex is TagRun Server, written in C# for the platform.Net Core. It represents the connecting layer of the classical three-level architecture. The main frameworks used for its development are ASP.NET and Entity Framework. The first one organizes a web server for the application and makes it possible to easily and quickly create handlers for requests. Based on ASP.NET The REST API is implemented, which is used by most of the other components for interaction. The logic of working with each of the entities is implemented in a separate controller, and data is exchanged in JSON format [4].

The second equally important framework allows you to use entities from the database in language objects that are convenient for code development. That is, it provides an opportunity to work with table elements, as with ordinary classes, avoiding writing SQL queries. At the same time, not only monosyllabic commands for adding or receiving data are supported, but also multilevel filters with combining elements (Figure 3). The TagRun Client application is a JavaScript web application. The use of modern web development technologies allows you to create a single client application for any platforms that support a web browser. The client developed by us interacts with the server via REST. The application consists of two screens: working with queries and working with filters.

The client's functionality includes:

1. adding, modifying, and deleting monitoring requests,

2. overview of the collected materials,

3. setting up filters of the received images,

4. exporting Filtered Images.

The main library for creating a web application in our project is React JS. This is a popular choice among developers to create web applications. React allows you to create interface components that are automatically updated when the application status changes

(switching to another screen, updating content, etc.). Thus, the developer can create a responsive and fast user interface that is convenient to work with. Also, due to its modular structure of components, applications created using this library are easy to expand and maintain. Centaur collects images. This application is also written in C#. Every day it receives information about monitoring requests from the server. For each of the queries, a search is performed in the Internet search engine (Google, Yandex), after which it collects and saves the results back to the server. This process is essentially web scraping: collecting information from web pages in order to convert it into a convenient format for work (Figure 4).

```
// POST: api/Image
[HttpPost]
0 references
public async Task<ActionResult<ImageJson>> UploadImage([FromForm]ImageUploadForm form)
{
    (var image, var isDup) = await imageValidator.FindDuplicate(form);
    if (isDup)
    {
        return new ImageJson(image) { IsNew = false };
    }
    try
    {
        image = await imageSaver.Save(image, form);
    }
    catch (Exception e)
    {
        logger.LogError(e, "Failed to save image");
        throw;
    }
    await context.AddImage(image);
    return new ImageJson(image) { IsNew = true };
}
```

Figure 3: Image loading processing using ASP.NET.

```
1 reference
private Task<IEnumerable<string>> ParseYandexTagsPage(string json)
{
    var jobs = JObject.Parse(json);
    var block = jobs["blocks"]
        .Where(c => c["name"]["block"].Value <string>() == "content_type_search-by-image")
        .FirstOrDefault();
    if (block == null)
    {
        return Task.FromResult(new List<string>() as IEnumerable<string>);
    }
    var html = block["html"].Value<string>();
    var hdoc = new HtmlDocument();
    hdoc.LoadHtml(html);
    var divWithTags = hdoc.DocumentNode
        .SelectNodes("//div[@class='Tags-Wrapper Tags-Wrapper_clip Tags-Wrapper_flow_wrap']")
        .Skip(1).First();

    var res = new List<string>();
    if (divWithTags != null)
    {
        foreach (var child in divWithTags.ChildNodes.Where(c => c.Name == "a"))
        {
            res.Add(child.InnerText);
        }
    }
    return Task.FromResult(res as IEnumerable<string>);
}
```

Figure 4: Getting tags for an image.

The role of calculating duplicates and similar images is assumed by Satyr. This is an application written in Python to determine the similarity of images. To do this, each image goes through a hashing algorithm. We use some kind of perceptual hashing. Its main difference from other algorithms is that the data is hashed as an image, not arbitrary information. In short, the algorithm we use can be described as follows: the image is compressed to a given size, then reduced to a certain shade of gray, also depending on the parameter (for example, with a minimum value, there can only be black or white), after which each of the pixels of the image is written out as a numeric value. To compare images with each other, it is enough to calculate the difference between the corresponding hashes, if it is less than a certain value, then the images are very likely similar. Harpy collects information from news resources. This web scraper is also written in Python. After collecting information about the news, an onto-semantic analysis takes place in order to highlight common topics among the received data. Based on the selected groups of words, search queries are generated, which are then saved to the server.

Conclusion

In the course of this work, a software package of the second version was developed, which implemented both the old and new functionality for automatically generating search queries, collecting images for these queries, tagging, filtering, grouping and exporting them. In the process of writing this project, software libraries and technologies for creating application complexes were studied. The architecture of a software product consisting of several applications was successfully designed. To create all the applications listed above, the programming languages C#, JavaScript and Python were used. In the course of writing the code, current approaches and best practices in creating fast and reliable applications were studied. In the future, it is possible to finalize the complex: analysis of the results obtained, visualization of the collected information, forecasting the further spread of Internet memes.

References

1. Martin RC (2017) Clean architecture: A craftsman's guide to software structure and design. Prentice Hall, New Jersey, USA.
2. Spinellis D, Gousios G (2009) Beautiful architecture: Leading thinkers reveal the hidden beauty in software design. O Reilly Media, Massachusetts, USA.
3. Troelsen A, Japikse P (2017) Pro C# 7: With net and net core. Apress, New York, USA.
4. Kozlova MG, Lukianenko VA, Germanchuk MS (2021) Development of the toolkit to process the internet memes meant for the modeling, analysis, monitoring and management of social processes. In: Abbasov IB(Eds.), Recognition and Perception of Images, Fundamentals and Applications, Chapter 6, Wiley-Scrivener, Texas, USA, pp. 189-219.

