# Design and Implementation of a High-Performance Smart Collector for Web Data Acquisition

**Kamel H. Rahouma, Farag M. Afify and Hesham F. Hamed**

Electrical Engineering Department, Faculty of Engineering, Minia University, Minia, Egypt.

kamel_rahouma@yahoo.com

## Abstract

Indexing or semantic processing of the Internet huge data is tiring and energy consuming. Data collectors are used to transform these unstructured into structured forms that can be stored and analyzed in a central local database or spreadsheet. Traditional data collectors are tiresome and time consuming in their execution where users need to filter and manually delete lots of information. Smart collector software is the easiest collection technique since all the other techniques except traditional copy and paste require some form of technical experience.

This paper aims to design and implement a system of an automated web data collector which can collect the needed data from deep web quickly and accurately for web data acquisition. An overview of web data acquisition and its challenges is given. A literature review is introduced and then the proposed data collector is explained. This includes the systems software components and their algorithms. An application example is introduced and the results are discussed compared with the results of available collectors (e.g., Xenu, Sitemap Generator and Screaming Frog). The behavior of the proposed collector is faster, more accurate and able to deal with the complex sites than the other ones.

**Keywords:** *Data Collection, Web Data Acquisition, Smart Collector, Information Extraction.*

## 1. Introduction

The internet contains huge types of data of different resources. Some of this information is related to social networks, or to financial, industrial and agricultural transactions, scientific research and etc. Most of the people access information through internet for different purposes and data is available in various formats and through different access interfaces. Users need to collect and analyze data from multiple places and websites. This is the essential part of any research in the field(s) of information. The different websites which belong to the specific category displays information in different formats. However, indexing or semantic processing of the data through websites could be very tiring and energy consuming. The only option is to manually copy and paste the data shown by the website to a local file in your computer. This is a very tiring job which wastes a lot of time and money. [1]

Data collectors are the tool (software) which aims to solve this problem. A lot of techniques have been proposed to help people to collect data from multiple websites to a single spreadsheet or database. Smart collectors are used to transform unstructured data on the web into structured forms that can be stored and analyzed in a central local database or spreadsheet. Figure(1) shows the data collection process which makes it easy to analyze or even visualize the data.

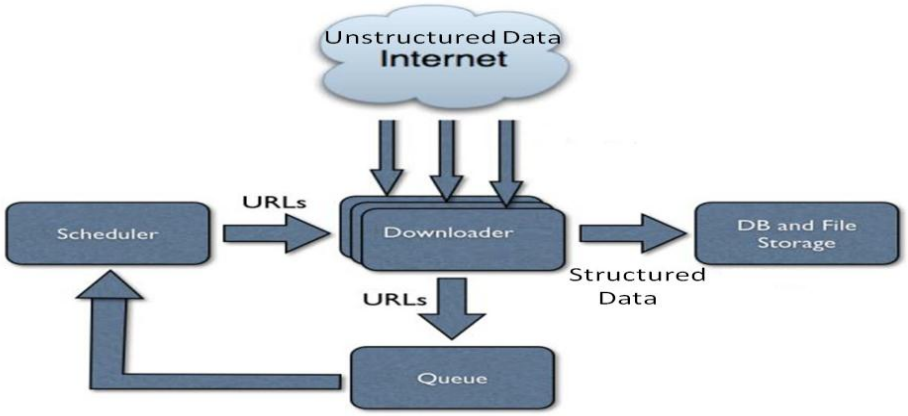


**Figure (1): Data collection from internet**

Data collection in a handcrafted way (i.e., search, copy and paste data in a spreadsheet to later processing) is truly inefficient and a waste of time and money. This is a tedious, annoying and tiresome process. Therefore, it makes much more sense to automate this process.

Data collector tool is a technology solution to collect information from web sites, in a quick, efficient and automated manner, offering data in a more structured and easier format to use, either for business-to-business or business-to-consumer processes. Data collector tools are also similar to Web Data Extractors, Data Harvesters, Crawling Tools or Web Content Mining Tools.

The worldwide web (or the Internet) is a wide range of billions of web pages that contain large bytes of information or data organized in a number of N servers. It is really difficult to locate deep web databases, because they are not registered with any search engines. Generally, they are limitedly distributed and constantly changing.

Over the past several years there has been a tremendous increase in the amount of data that produced by billions of different operations. Escalating usage of streaming multimedia and other Internet based applications has contributed to this surge in data usage. Another facet of the increase of information is the appearance and expansion of Big Data, which indicates that data sets are organized in many orders of magnitude larger than the standard files transmitted via the Internet. Big Data can range in size from hundreds of gigabytes to petabytes [2].

Within the past decade, everything from banking transactions to medical history has migrated to digital storage. This change from physical documents to digital files has necessitated the creation of large data sets and consequently dealing with large amounts of data. There is no sign that the continued increase in the amount of data being stored and submitted by users is slowing down. Every year companies, researchers and internet users are producing more and more data through various processes. With the growth of internet based applications, the amount of data being stored in distributed systems around the world is growing rapidly. There is a need for an efficient technique that can collect and store large amounts of data in different fields quickly and easily without impacting other users or applications [3].

The basic motives behind web crawler is to retrieve web pages and add them to a local repository, collect, store the needed data from them and update them every given period. Basically, it starts from a parent site and then uses hyperlinks and sub-links contained in it to reach other pages. It records every hypertext link in every page to index crawling. It keeps on repeating until it reaches a predefined value.

Geographically dispersed researchers eagerly await access to the newest datasets as they become available. The task of providing and maintaining fast and efficient data access to these users is a major undertaking. Also, operations such as: data collect, data transfer, data store, data access, and processing are critical for efficient resource allocation. This requires the gathering of metadata which describes the data from geographically distributed sources and the processing of such information to extract the relevant information [4].

Today's Internet can be considered as a vast pool of information from which any data can be found. But still there is so many data which cannot be retrieved by any search engine. Such data is found in deep web pages.

The deep web refers to the contents that lie behind HTML forms and cannot be identified by normal crawlers. This data is buried deep down the dynamic web pages and hence cannot be found by any search engines. They include dynamic web pages, blocked sites, unlinked sites, private site content and limited-access networks. The information available in deep web is approximately 400-550 times larger than the surface web. More than 200,000 deep websites currently exist [5].

Nowadays collectors can retrieve data from websites which are indexed, i.e., the webpages that can be reached by following hypertext links, ignoring webpages that need authentication or registration. So, they ignore a large amount of high quality data hidden behind search forms, in large searchable electronic databases. Deep web contains huge amounts of valuable information which are highly scattered. So, there is a need of automated data collector which can collect the needed data from deep web quickly and accurately.

This paper aims to design and implement an efficient automated data collector. The paper includes eight sections. Section (1) is an introduction and section (2) introduces an overview of Web Data Acquisition (WDA). Section (3) presents the challenges in web data acquisition and section (4) introduces the previous and related work in the field of data collection. Section (5) discusses the proposed system of data collector including its description and algorithms. Section (6) explains the application of system algorithms for a typical example of data from site: www.arconic.com and section (7) discusses the results and compares the behavior of our system with some of the available ones. Section (8) depicts some conclusions and future work points and a list of the used references is given at the end of the paper.

## 2. Overview of web data acquisition

The field of data science has become a widely discussed topic in recent years due to a big data. Large companies and factories are keen to enhance their competitiveness by learning about their customers to provide tailor made products and services, dramatically increasing the usage of sensor devices. Traditional techniques of collecting (e.g. lightweight Python framework), storing (e.g. Oracle) and analyzing (e.g. PL/SQL) data are no longer optimal with the overwhelming amount of data that are being generated [6]. The challenge of handling big volumes of data has been taken on by many companies, particularly those in the

internet domain, leading to a full paradigm shift in methods of data collecting, processing and storing. A number of new technologies have appeared, each one targeting specific aspects of large-scale distributed data-processing.

All these technologies, including batch computation systems (e.g. Hadoop) and non-structured databases (e.g. Mongo DB), can handle very large data volumes with little financial cost. Hence, it becomes necessary to have a good understanding of the currently available technologies to develop a framework which can support efficient data collection and storage.

There are various kinds of web data acquisition for example web data extraction, web data scraping, web harvesting or screen scraping. The overall aim of all these techniques is to extract information from websites and transform them into understandable structures like spreadsheets and database. Information differs depending on the field, which is its own data collection. For example extracting targeted data from websites helps to take effective decisions in your business.

Web data acquisition has gained importance because of the need to collect huge amount of information in a little time. Many people, whose work in research and development and professionals need huge amount of information in order to process it, analyze it and extract meaningful results. On the other hand, people dealing with B2B use cases need to access data from multiple sources to integrate it in new applications that provide added value and innovation. So the need demands holistic data collection solutions from web sites.

## 3. Challenges in Web data acquisition

In this section we discuss some of the challenges result in web data acquisition.

### 3.1. Scale

The web is growing at a very large scale day by day. For the collectors to achieve a broad coverage and a good performance, it needs to give very high throughput. This led to the creation of a large number of engineering based problems. To overcome these problems, the companies need developing data collection methods continuously

### 3.2. Content Selection

There are a number of crawlers which provide high throughput but are unable to crawl the whole web and cope up with the changes. The goal of crawling is to acquire content with higher value more quickly and gather information which contains all the reasonable content. Crawlers should ignore all the irrelevant, redundant and malicious contents.

### 3.3. Safety

Data collectors should follow safety mechanisms to avoid the denial-of-service attacks. Data collectors should establish a good coordination with the different websites for which they work.

### 3.4. Adversaries

There are some content providers which try to inject useless content into the corpus of the data collectors. Such types of activities are motivated by financial incentives like misdirecting Traffic to commercial websites.

### 3.5. Copyright

Data collectors ostensibly do something illegal: they make permanent copies of copyright material (data in web pages) without the owner's permission. Copyright is perhaps the most important legal issue that impedes data collectors.

### 3.6. Privacy

For data collectors, the privacy issue appears clear-cut because everything on the web is in the public domain. Web information may still invade privacy if it is used in certain ways, principally when information is aggregated on a large scale over many web pages.

## 4. Previous and related work

In [7], an effective strategy is designed and implemented to collect and store huge information. It indicates how important it is to choose the correct form to get high performance. It is clear from the study that maintaining the main logic in a central location will simplify technical and architectural migration. Performance test results show data level elimination. Collected data is stored at the storage level and the required conversion may be performed when needed using a framework such as Map Reduce.

In [8], they suggested an effective algorithm for Deep Web, called Smart Crawler. Smart Crawler is work on two-stage: effectively locating and exploring a balanced site. This algorithm locates a site-based site by reversing the search of known Web sites for central pages, which may effectively find a lot of information in many domains. By using collected sites and care about Deep Web, it gives high efficiency.

In [9, 10], the researchers did a survey on various ways of crawling. In the past, many problems have emerged for example efficiency, end-to-end delays, link quality and failure to find deep locations because they are not registered with any scattered and dynamic crawler.

So they suggested an efficient tool for collecting deep web pages, this tool achieves widespread coverage of the Internet deep and implements a proficient craw l technique. It results in site discovery using reverse search technology to get pages from known deep web pages, and thus finds many relevant information sources from a lot of fields for example Xenu.

Grid computing has appeared as systems for collecting geographically distributed heterogeneous resources that enable secure access to computing, storage and networking resources for Big Data [11]. Its applications have wide sets of data and / or complex arithmetic operations that require safe resource sharing between geographically distributed systems.

It allows for coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations [12]. A virtual organization (VO) comprises a set of individuals and/or institutions having access to computers, software, data, and other resources for collaborative problem-solving or other purposes. A grid can also be defined as a system that coordinates resources that are not centrally controlled using standard, open, general purpose protocols and interfaces for the provision of imperfect qualities of service [13].

Parallelisation is used in order to enhance computations of Big Data. The well known Map Reduce framework [14] that has been used in a lot of companies has been well developed in the area of Big Data science and has the parallelisation feature. Its other key features are: its inherent data management and fault tolerant capabilities.

The Hadoop framework has also been employed in many places. It is an open-source Map Reduce software framework. For its functions, it relies on the Hadoop Distributed File System (HDFS) [15], which is a derivative of the Google File System (GFS) [16]. In its function as a fault-tolerance and data management system, as the user provides data to the framework, the HDFS splits and replicates the input data across a number of cluster nodes.

The approaches for collecting and storing Big Data for analytic description were implemented on a community-driven software solution (e.g., Apache Flume, Sitemap Generator [17] and Screaming Frog [18]) in order to understand how the approaches integrate seamlessly the data pipelines. Apache Flume is used for effectively gathering, aggregating, and transporting large amounts of data. It has a flexible and simple architecture, which makes it fault tolerant and robust with tunable reliability and data recovery mechanisms. It uses a simple extensible data model that allows online analytic applications [19].

## 5. The proposed data collector

With the increasing volume of data on the Internet, there has been an increasing interest in data collection technologies. Given the increasing data size and the complex nature of the Internet, achieving a wide coverage and a high efficiency may be a difficult issue and it requires anongoing development of existing tools. We propose a data collector which contains a two stage specific smart collector to efficiently gather data in any specific fields. This system is aimed to be an effective tool to collect and store large amount of data in specific field. It increases the speed of collecting data and solves the problems of delay, redundancy, network load and load on servers as well as energy management.

We would like to design a flexible system that can be adapted to different applications and strategies with a reasonable amount of work. To do that, we describe collecting strategies to efficiently locate the entry points to the hidden Web sources. The fact that hidden Web sources are very sparsely distributed makes the problem of locating them challenging. Thus, we deal with this problem by using the contents of pages to focus on a topic; by prioritizing promising links within the topic; and by also following links that may not lead to immediate benefit by using status database. We propose a new framework whereby collectors automatically learn patterns of promising links and adapt their focus as the collect progresses, thus greatly reducing the amount of required manual setup and tuning. The smart collector retrieves up to three times compared to collectors that use a fixed focus strategy. The general processes that a smart collector carries out are as follows:-

1) The smart collector takes inputs sequentially from the input database and the checks up for their priority levels.
2) The system keeps inputs in a queue based on their descended priority levels for downloading.
3) The needed data are downloaded from the pages.
4) If any links are given in the downloaded pages (e.g., additional website and page addresses), the system extracts them and adds them to the priority queue for later downloading.

5) The priority queue is stored in a pre-determined location.
6) The status of links is saved in the status database.
7) A summary of the processed link and its processing date are saved for the downloaded processes such that the system knows when it should re-check the inputs at a later stage.
8) The smart collector maintains a priority queue for all the sub-links and hyperlinks which are classified by categories according to their importance.

### 5.1 Processes of the smart data collector

This smart collector contains two parts as shown in figure (2):

a- WDAmanager

b- WDA driver

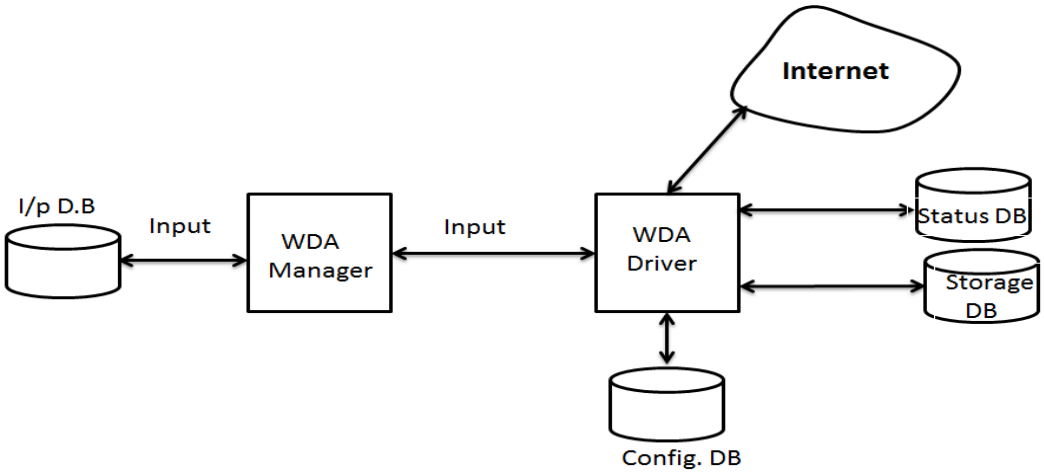


**Figure (2): Elements of the smart collector**

### A. WDA Manager

The automatic collection system depends on inputs that are identified for the system and present in input database, then the system determines the highest priority for these inputs and then determines the appropriate configuration for these inputs. After doing the previous steps the system passes the input and configuration to WDA driver. Figure (3) gives the flow chart of the WDA manager processes.

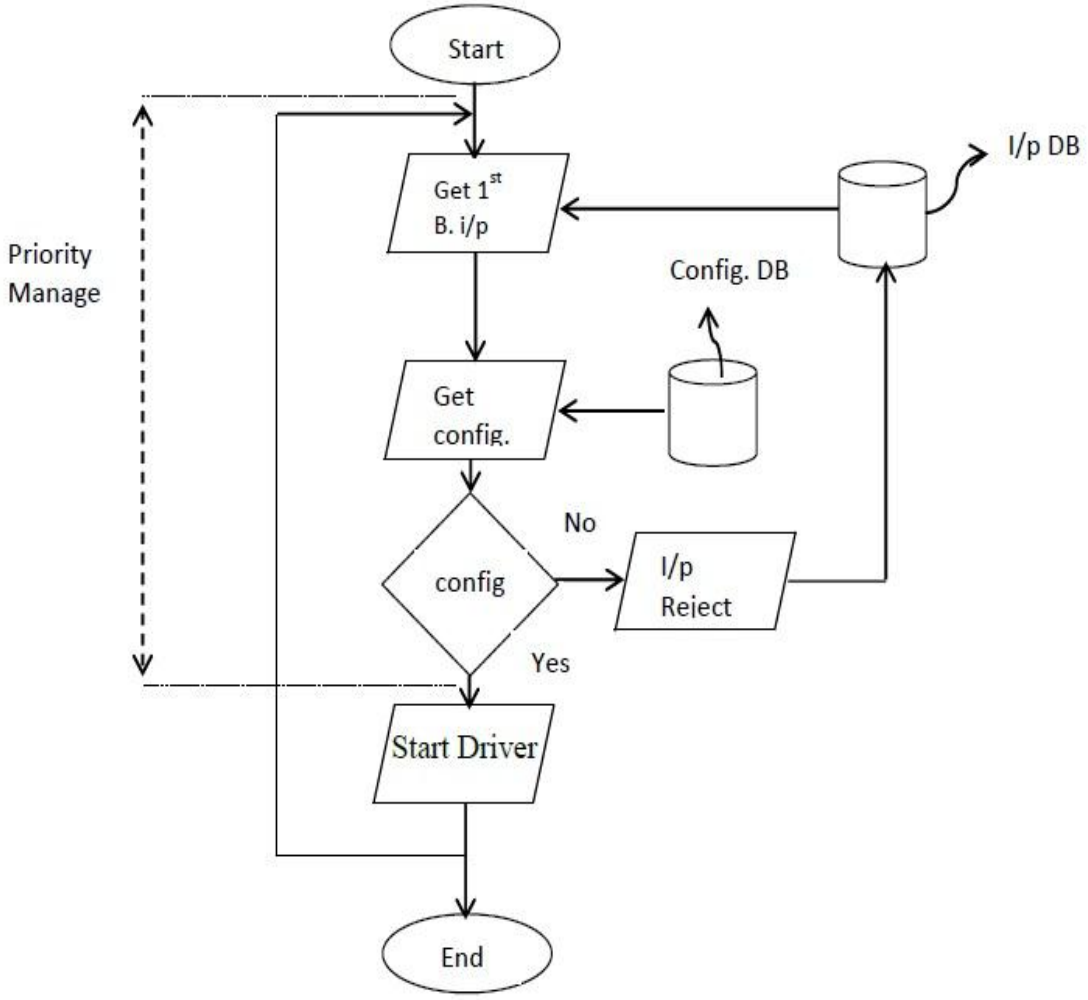


**Figure (3): The flow chart of the WDA manager processes**

**B) WDA Driver**

The system works on the first input that was passed to WDA driver from WDA manager and opens the start page that is specified in the configuration. If there is a fault in the opening of the start page, the system will record a fault in the status DB and go to the second input. The system opens all the hyperlinks on the start page and then the following pages. If there is a fault in any page, it registers a fault in status DB. The system collects data from all open pages and then stores them in the Data DB and places a tag found in status DB. This process is repeated until all inputs are finished. Figure (4) shows the WDA driver processes.

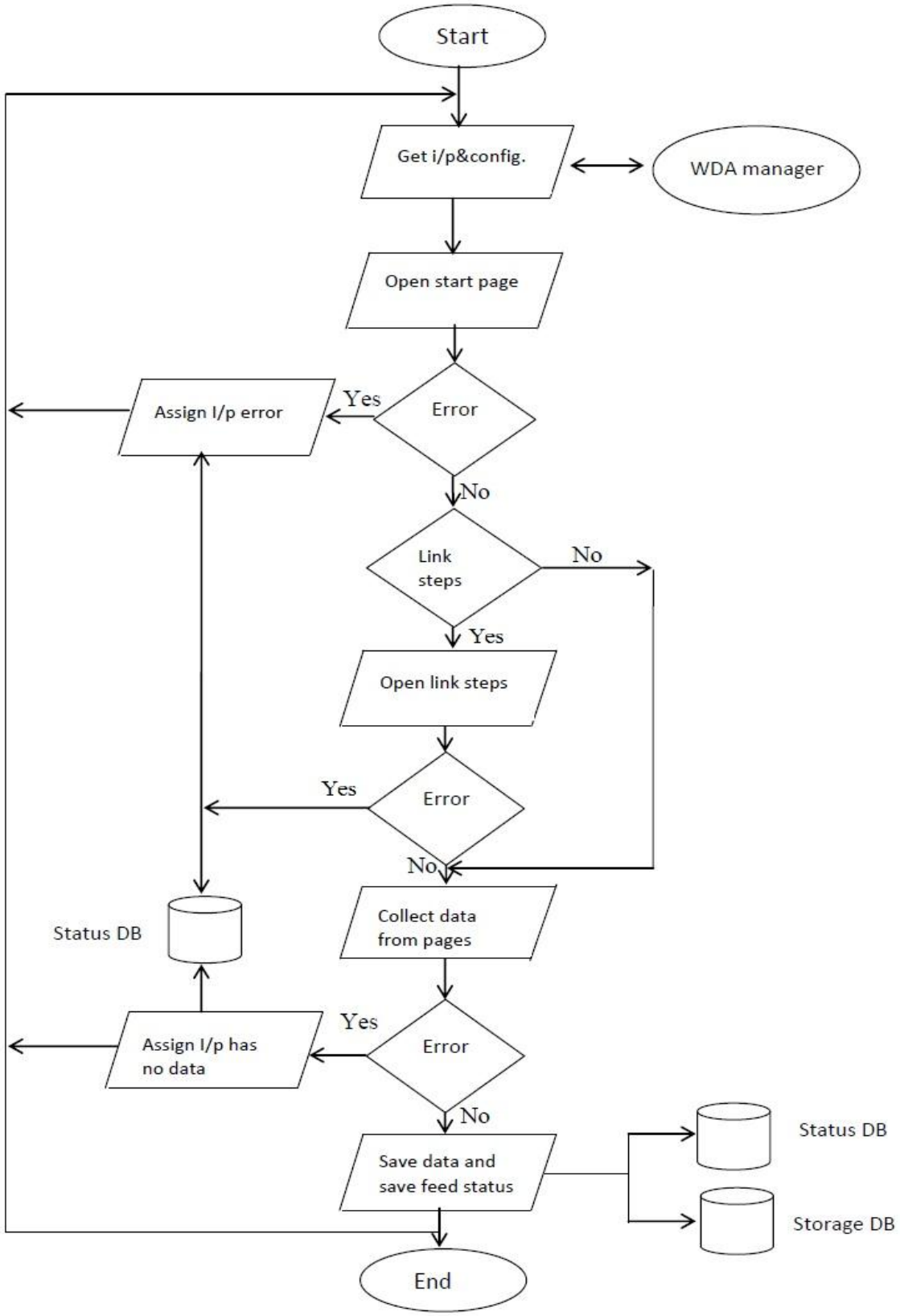


**Figure (4): The flow chart of the WDA driver processes**

### 5.2 Algorithms of the smart data collector

#### A) Algorithm 1

Given: Original Seed links list.
Output: Modified Seed links list.

```
While Seed Links List has More Links do
        Current Seed Link = get Current Seed Link (seed Links List)
        page = download Page From  Link (current Seed Link)
        links  List = extract  Links From Page (page)
        for Each link in Links Lins do
                approved = check Link(link)
                if approved html then
                        add To  Seed Links List(link)
                Else
                        // Check if it is a document to be downloaded  (pdf, zip, doc, xml, excel
                           etc …)
                        Is Document = check If Link Is Document(link)
                        If is Document Then
                                Down  Load  To Drive(link)
                        End
                End
        End
End
Return seed Links List
```

#### B) Algorithm 2

Given: HTML links list and data Selection Configuration
Output: Data Rows List

```
For Each link in links do
        areas = select Focused Areas(link, data Selection Configuration)
        for Each area in areas do
                images = get Images From  Area if(area)
                tables = get table From Area(area)
                texts = get Text From Area(area)
                hyper Links = get Links From Area(area)
                if contains Data Do
                        add To data Row List(data Rows List, link, images, tables, texts,
Hyperlinks)
                Else
                        Put no data status
                End
        End
End
Return Data Rows List
```

**These algorithms are implemented using Java programming language**

# 6. Application and Results

## 6.1 Application Steps

1. Identify list of sites where we get data from it, and give them Initial priority
2. Smart collector start to collect data from these sites
   - The first step of collecting data, Resulted in a list of links which contain the data in these sites.
   - The second step , The system classify the links according to the type of data found in each link (files, doc, html,....) and types of documents ( manual , datasheet , application note,...)
3. The system downloaded and stored all data according to all different categories.
4. For web pages that contain data that have been identified using selected criteria and sent it to the data extraction stage.
5. Specific configuration has been done to select the pages containing the required data and settings for the data places within the pages.
6. These data were downloaded and stored according to their classification.
7. Smart collector will repeat this process automatically every fixed time (for example 3 days) and compare the new result with previous result.
8. The system takes decision if it found broken links.
9. Based on the result, the system changes the priority of inputs.

## 6.2 Results

We collect many types of data from many sites for example: www.arconic.com. Applying the previous steps, we got the following:

1. The result of first step of collecting data is a list of links which contain the data in these sites, we got a list of 86359 links after 5hours and 35 minutes.
2. The second step classification the links according to the type of data found in each link, 12236 zip files, 15243documents, 18790 html, 14330photo,12502 broken links and 13258 other extensions.
3. Specific configuration has been done to download zip files and documents only, the system downloaded 12146 zip files and 14974 documents.
4. All downloaded files were stored in the places specified for each classification.
5. The system failed to download some files in the practical application and these cases happened due to:
   - Security problem of site (ex. need user name and password)
   - Server response delay (download take very long time)
   - Broken link.
6. The system was set to repeat this process every week and compare the new result with previous result.
7. Finally, we have amount of data to be used in scientific research, industry, marketing and etc.

Figure (5) is a screenshot taken during the running of our collector. The figure shows the event manager, URLs extracted, document fetched, document imported and number of processed events related to the total number of events at the moment of taking the screenshot.

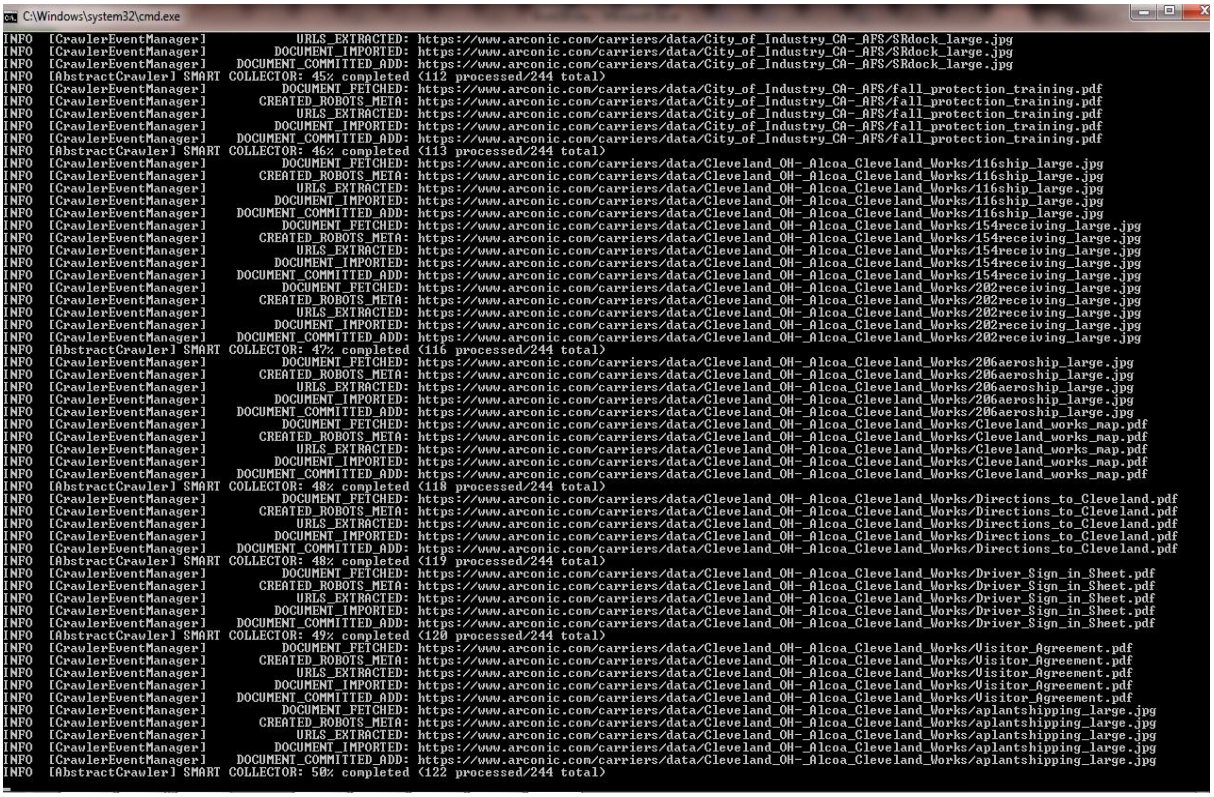


**Figure (5): A screenshot during the running of the collector**

## 7. Discussion and Comparison with Previous Work

The program works as an engine in the background. In other words, it is a cmd (a command line event, e.g., application works from the console)and it has no GUI. Thus, the progress of the program RUN can be shown and followed up in the console. Like all console running programs, the results can be saved into an xml file for monitoring purposes (either in its xml format or by converting it into a web format according to the used browser).

Our smart collector was written completely in Java, which gives flexibility through pluggable components. Added to that, our proposed collector can be compared with previous ones according to the following points:

a) Our proposed collector works on different strategies rather than traditional collectors. Thus, it is called "the focused collector" or "the smart collector". This collector is distinguished from the older ones as follows:

1) The collector that tries to download data that are related to each other.

2) It collects information which is specific and relevant to the given field. It determines how far the given data is relevant to the particular topic and how to proceed forward. Thus, it determines the Relevancy and Way forward.

3) In order to refresh its collection, periodically it replaces the old documents with the newly downloaded documents and incrementally refreshes the existing collection of inputs in data base by visiting them frequently. It also exchanges less important inputs by new and more important inputs.
4) Distinguishably, our collector is economically feasible in terms of hardware and network resources, network bandwidth is saved; data enrichment is achieved and downloads. It resolves the problem of the freshness of the data.
5) Our collector works on distributed computing technique in order to have the most coverage of the web and this makes it robust against system crashes and other events.

b) In [5, 12, and 13], the proposed collectors were designed for fairly powerful systems (servers) with several CPUs and fast disks. One major advantage of our collector is that it works for all systems and devices and it doesn't need much equipment or interfaces needed by other systems.

c) Comparing our collector with the available collectors (e.g. Xenu,Sitemap Generator and Screaming Frog), our collector is found to be faster. Also, our collector easily dealt with and solved the problem of complex sites (sites using complex script) which were not solved by these collectors. During the test, Xenu failed to do any actions such download, store and insert links in data base while our system doing these operations with high performance. Sitemap Generator and Screaming Frog could not get rid of duplicate links problem while our proposal solves this problem.

## 8. Conclusions and future work

A high performance system (collector) for Web Data Acquisition is designed and implemented in JAVA language. This gives flexibility through pluggable components. The collector is called focused or smart and it is distinguished from the traditional ones by its fast speed, accuracy, and ability to deal with complex sites. Our smart collector was compared with some available ones (Xenu,Sitemap Generator and Screaming Frog) and it gives higher features than them.

In the future, work can be done to improve the efficiency of the available algorithm and to develop new ones. Also, the accuracy and timeliness of the data collectors can also be improved. The work of the different collecting algorithms can be extended further in order to increase the speed and accuracy of Web data acquisition systems. A major open issue for future work about improving the scalability of the system and the behavior of its components. Building an effective data collector to carry out different purposes is not a difficult task, but choosing the right strategies and building an effective architecture lead us to implement highly intelligent data collector applications.

## Reference

[1]. JohnGantz and David Reinsel, "THE DIGITAL UNIVERSE IN 2020: Big Data, BiggerDigital Shadows and Biggest Growth in the Far East,"IDC's Digital Universe Study, December 2012, sponsored by EMC. URL: http://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf

[2]. Hoda A. Abdel Hafez, " Big Data in Smart Cites: Analysis and Applications in Arab World",Egyptian Computer Science Journal, Volume 41, Issue 1, January 2017.

[3]. Snijders C, Matzat U and Reips U-D.,"Big Data: big gaps of knowledge in the field of internet science,"International Journal of Internet Science, 2012, 7 (1), 1–5.

[4]. Magnoni L, Suthakar U, Cordeiro C, Georgiou M, Andreeva J, Khan A and Smith DR,"Monitoring WLCG with lambda-architecture: a new scalable data store and analytics platform for monitoring at petabyte scale,"Journal of Physics: Conference Series 664 (2015) 052023.

[5]. Anjum Asmaand and GihanNagib,"Energy Efficient Routing Algorithms for Mobile Ad Hoc Networks–A Survey," International Journal of Emerging Trends & Technology in computer Science, Vol.1,Issue 3, pp. 218-223,October 2012.

[6]. UthayanathSuthakar , Luca Magnoni, David Ryan Smith, Akram Khan and Julia Andreeva "An efficient strategy for the collection and storage of large volumes of data for computation" College of Engineering, Design and Physical Sciences, Brunel University London, UK , J Big Data , springer ,2016.

[7]. Ahmed Alhamzi, Mona Nasr, ShaimaaSalama" A Comparative Study of Association Rules Algorithms on Large Databases ", Egyptian Computer Science Journal Vol. 38 No.3 September 2014.

[8]. Feng Zhao, Jingyu Zhou, Chang Nie, Heqing Huang andHai Jin, "Smart Crawler: A Two-stage Crawler for Efficiently Harvesting Deep-Web Interfaces," IEEE Transactions on Services Computing Volume: 99, Year 2015.

[9]. Nimisha Jain, Pragya Sharma, SaloniPoddar and Shikha Rani, " Smart Web Crawler to Harvest the Invisible Web World," B.E. Student, Dept. of Computer Engineering, MIT College of Engineering, Pune, India, International Journal of Innovative Research in Computer and Communication Engineering, April 2016.

[10]. http://home.snafu.de/tilman/xenulink.htmlAccessed Jan. 2018.

[11]. Foster I, Kesselman C. The grid 2: blueprint for a new computing infrastructure. San Francisco: Morgan KaufmannPublishers Inc.; 2013.

[12]. Foster I, Kesselman C, Tuecke S. The anatomy of the grid: enabling scalable virtual organizations. Int J High PerformComput Appl. 2011;15(3):200–22.

[13]. Foster I. What is the grid? A three point checklist, GRIDToday.GRIDToday; 2015

[14]. Dean J and Ghemawat S.,"MapReduce: simplified data processing on large clusters," Communication of the ACM. P 107–113, Vol. 51, No. 1, 2008.

[15]. Shvachko K, Kuang H, Radia S andChansler R.,"The Hadoop distributed file system," IEEE 26th symposium on mass storage systems and technologies (MSST),p. 1–10, 2010.

[16]. Ghemawat S, Gobioff H and Leung ST.,"The Google file system" ACM SIGOPS OperSyst,375:29–43,Oct. 2003.

[17]. https://www.auditmypc.com/free-sitemap-generator.asp /Accessed Jan. 2018.

[18]. https://www.screamingfrog.co.uk/seo-spider/Accessed Jan. 2018.

[19]. Apache Flume project. https://flume.apache.org. Accessed Oct. 2017.