White Hat Data Harvesting: Industrial-Strength Web Crawling for Serious Content Acquisition

DCL™

Data Conversion Laboratory Inc.

DCL
61-18 190th Street
Suite 205
Fresh Meadows, NY
11365
+1.800.321.2816
www.dclab.com

Mark Gross
President, Data Conversion Laboratory

Tammy Bilitzky
Chief Information Officer, Data Conversion Laboratory

Rich Dominelli
Lead Software Engineer, Data Conversion Laboratory

Allan Lieberman
Special Projects Manager, Data Conversion Laboratory

ABOUT DCL

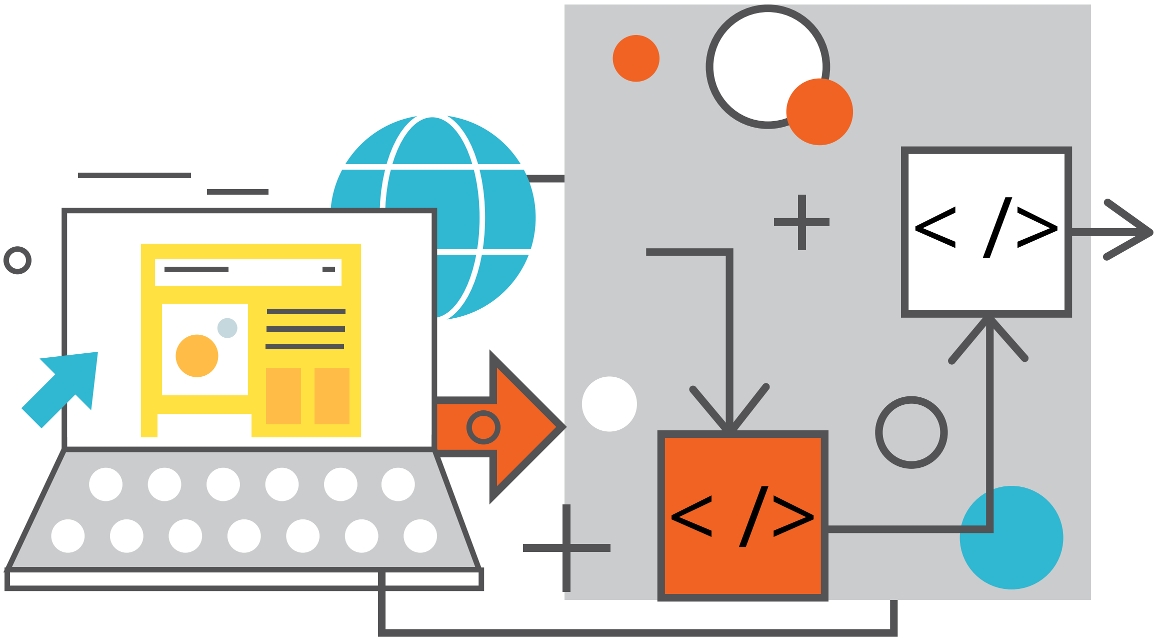
Intelligent data transformations

DCL (www.dclab.com) provides data and content transformation services and solutions. Using the latest innovations in artificial intelligence, including machine learning and natural language processing, DCL helps businesses organize and structure data and content for modern technologies and platforms. With expertise across many industries including publishing, life sciences, government, manufacturing, technology and professional organizations, DCL uses its advanced technology and U.S.-based project management teams to solve the most complex conversion challenges securely, accurately and on time.

Your data:

transformed, validated, enriched

CONTENTS



An Overview of Web Crawling

Vast amounts of business-critical information appears only on public websites that are constantly updated to present both new and modified content. While the information on many of these websites is extremely valuable, no standards exist today for the way content is organized, presented, and formatted, or for how individual websites are constructed or accessed.

This creates a significant challenge for companies that require data sourced from these websites in a timely manner, which they need downloaded and structured to support business practices and downstream systems.

This paper focuses on specific impediments that we typically encounter and tactics we've adopted to overcome them in the process of creating a streamlined, automated processes to crawl websites, scrape content and metadata, and transform the content into a standardized XML format. Our comments and recommendations are based on having successfully traversed hundreds of variated, multilingual, multi-platform, global websites.

We will elaborate on our methodology and bots used to facilitate high-volume data retrieval in a variety of source formats (HTML, RTF, DOCX, TXT, XML, etc.), in English, European and Asian languages, with varying organizational approaches.

White Hats vs. Black Hat—Good Guys or the Bad Guys?

While web crawling sounds somewhat nefarious, there is an important white-hat side to it. Much original source material today appears only on the Web. For many government agencies and various NGOs, the web version is the "document of record," the most current version available, and where you are referred when you make inquiries regarding reports, articles, white papers, etc.

While there are many tools available to handle the basic crawling and scraping of websites, they mostly work on one website at time. Analyzing and traversing volumes of complex websites—somewhat like developing autonomous vehicles—requires the ability to adapt to changing conditions, across websites and over time. This presentation will examine the thought processes behind our approaches, including website analysis, techniques to detect and deal with website and content anomalies, methods to detect meaningful content changes, and approaches to verifying results.

Why Do We Need This Information?

There is a vast amount of data available on websites, from informative to entertaining to legal, with critical content that services a wide variety of purposes, depending on the business need. The most common endgame is the normalization, decomposition, and transformation of the information into a structured format to power derivative databases, data analytics platforms, and other downstream systems.

Why All the Fuss?

It is estimated that there are 4.52 billion webpages out in the wild (http://www.worldwidewebsize.com/). Many of these are maintained by webmasters who are certain that their architecture for running a web website is the best one, as opposed to the guy one page over.

Of course, it would be nice if all websites offered a convenient, reliable method to download and monitor their content, but most do not. It would also be helpful if all websites complied with standards to make new and modified content easier to find and extract, but no such luck; the variations are endless and often at the whim of the developer and content owner.

It would also be helpful if, once the website were in place, its design and structure remained static, but that doesn't happen either. Compounding it, in our hack-worried world, some restrict or limit access to prevent malicious intruders at the expense of legitimate users. Finally, software bugs introduced inadvertently by developers add to the challenge.

It is Not "One Size Fits All"

"We want the content from www.very_important_content.com." These marching orders launch our focused functional and technical analysis of a website.

Understanding the design of each individual website is a prerequisite for successful crawler automation and data harvesting. Our methodology guides both the website analysts, and later, the developers, through a series of questions designed to arrive at the best approach for each unique website and content set.

Some critical questions include:

- How does the website work? Where is the data of interest and how is it accessed? The possibilities are endless and include traversing menus, sequencing through tables of content, clicking on headlines, and entering search terms.
- How is the website content organized? Date order, subject matter, etc.? Understanding the way a website is organized is critical to locating the content you need and avoiding duplicates.
- What is the website depth? How many links do we need to traverse to access the content? Depending on your business need, you may want to limit your search depth.
- Is all the required metadata available on the website or does it need to be extracted from the content itself? Metadata is often even more important than the published content and is needed for validation and search. Getting the metadata from the best source is key.
- How large is the website? Is one crawler able to process it in its entirety? The resulting crawl process must be executed in a timely manner; this is a major consideration for the developer when configuring each website crawl.
- How consistent is the design of the website? How large a sampling is required to successfully specify requirements for the crawler automation? Some websites are highly structured and organized. Others have a surprise on every page.

One Best Approach? Wishful Thinking…

One learns quickly that one solution does not fit all. It is not feasible to design one approach to intelligently crawl even a small subset of these websites. Even within the same department, the web page layout and backend technology often varies, requiring frequent customizations.

Modern websites have progressed far beyond simple html pages to interactive, database-driven applications using logic residing both on the client page and server-based code. This forced a transition from NCSA Mosaic (last released OSX version size 1.7 mb) to the current version of Google Chrome, weighing in at a hefty 554 mb at the time of this writing. This growth in application size represents the ever-expanding feature set supported in modern browsers.

Our Methodology—How to Focus on What Matters

To attack these problems, we've identified a focused series of questions that guide the developer through the decision process and determine an optimized approach to extract content and metadata from each specific website. These include:

Does the website use a standard CMS (e.g. Drupal, Joomla, or WordPress)?

Consistency is the primary advantage to crawling a website that uses a standard CMS. The page layouts follow a pattern and the lists of content are organized with the same tagging scheme, often sharing the same metadata tagging across pages.

What is the Underlying Technology Stack?

If the website is hosted using ASP.net webforms, paging and navigation is typically implemented as form posts. If it is an Angular website, it may make heavy use of Ajax or a Single Page Application paradigm (SPA). The actual URL holding the content may not be immediately obvious, requiring emulation of the JavaScript-enabled browser or monitoring requests in the browser's html debugging tools to see how the data is being sourced. A similar situation will occur if a website makes heavy use of frames; often the actual content url is not the url in the address bar.

What security and authentication are in place?

Does the website require a logon? Does it require cookies or other headers that accompany the call and must be maintained between calls? The fastest way to crawl a website is to connect to the specific web address (URI) and retrieve the response, using the HTTP GET command, and then stream the results to a file. If the interaction between the server and browser is complex, it's unlikely it is that this approach will work.

Rules for polite web crawling to avoid being blocked

The difference between a DDOS attack and an aggressive crawler is slim. It is a fairly simple task to write a web crawler which spawns many threads, all simultaneously grabbing content from a given website to quickly extract all the content. However, this method will quickly get your IP address blacklisted and block you from the website.

A preferred method is to minimize the simultaneous connections and insert artificial pauses between the requests, mimicking normal user browser behavior. Even so, some websites will limit the number of files you can download in a given day from the same IP address. To avoid this, you either have to request files from multiple addresses or hook into the TOR network to use a different IP address on every request.

Is there an API/RSS feed available?

Some websites have a clean API available allowing you to pull the data in via a simple REST or SOAP call, including a few Federal websites. Others expose their content via RSS (Really Simple Syndication), eliminating the need to parse the HTML pages.

Does the website have bugs—and how severe?

Bugs can range from simple broken links and unavailable images, to flawed paging logic that only manifests itself when well into the development of a crawl. In some cases, webmasters are responsive and will address—or at least acknowledge—the flaws in their website, but often you simply have to find a way to work around the flaws.

Crawler Magic—From Their Website to Ours

Rarely are two websites alike. A viable crawl solution must accommodate the unique aspects of a website without starting from scratch each time we face a new nuance. Our toolchain approach, in which a set of components are assembled into a crawler, is our preferred method for crawling large numbers of diverse websites in an efficient, timely manner and has proven very effective.

Some of the components we configure in our toolchain approach include:

Page Downloading

At its core, a web crawler is a mechanism for bulk-downloading pages. The simplest mechanism is an HTTP GET, the HTTP command to access a URI and retrieve a response. This only returns the full page for simpler websites, but has a tremendous speed advantage and is our default mechanism. For sites that require cookies, we supplement the HTTP GET accordingly.

Pages are often loaded or changed dynamically by client-side scripts. Sections of text may be appended, deleted, or expanded. As our ultimate goal is downloading the complete contents of the page, we may need to emulate a browser.

Page Parsing

Parsing will grab elements from within the page and intelligently process them. There are several common approaches for selecting and navigating elements within web pages.

CSS selectors are commonly used by many JavaScript tools to quickly grab html elements and action them. Often many elements have no class, lack a distinct identifier, or repeat frequently.

Some pages rely on unique identifiers for the elements in question, but often only uniquely tag those elements they are interested in manipulating via CSS or JavaScript.

Many developer tools, e.g. Firebug and the Chrome Developer tool, let you query via Xpath to interactively preview your result, providing a more robust query language to quickly filter and navigate between elements.

The strength of using Xpath is based on the similarities between HTML and XML. HTML is relatively unstructured compared to XML and may not be well-formed. Thankfully, most languages have a forgiving parser that allow you to treat HTML as if it were XML. These parsers support a generic, Xpath-based mechanism for narrowing the relevant elements of a page and walking the elements of the page for metadata extraction and more complex filtering.

Metadata Extraction

In addition to the html documents, we are usually required to extract metadata from index or other pages. By walking the elements surrounding the link that led to this page, we are able to extract information surrounding the link that lead us to this page, similar to walking up and down the document object model.

Page Filtering

There are several options for filtering pages in a crawl:

- Limit the section of the page examined for links using Xpath.
- Examine the link itself for keywords that

indicate that the content is not in scope or duplicated.

- Apply logical filters, e.g. filtering out historic versions of a page.

Page Differencing

As you advance beyond simple file comparison, determining whether a page has changed on a website is a complex task; often requiring a multi-step process:

- Isolate only those areas of interest on the page.
- Strip tags that do not affect the meaning of the page such as head elements, style tags, JavaScript, and attributes within the tag.
- Assess if the difference is material. Switching from straight quotes to curly quotes or from normal spaces to non-breaking spaces are not usually meaningful. Other changes are more subtle such as paragraph transitions from preformatted text (<pre>) to lines contained within a paragraph or lines split by breaks—with no actual text differences.
- Apply intelligence to chunk sentences and sentence fragments to compare each word.

No Plan Survives Contact With a Webmaster

Sites change, pages are updated with new character sets, update notification pages are frequently wrong, links die or are changed. We couple our automated crawling with automated validation to ensure that we have all the required files and metadata. When we find discrepancies, alerts are issued and our website analysts often reach out to the webmasters. From then on, it is uncertain as to whether we will get a response or resolution, and we often have to implement workarounds.

What is Next?

We have developed a series of best practices for web crawling and harvesting technologies, achieving fully automated processing against a wide range of diverse, complex and often poorly structured websites. Our methodology has been iteratively refined to accommodate the ever-changing landscape of internet content and facilitate a model of continuous improvement.

We are far from done. While not there yet, we are well on our way to eliminate manual intervention and further automate website analysis, reducing greatly the manual effort to research and resolve problems. We are starting to leverage our volumes of training data to create training sets for machine learning based troubleshooting and information extraction, and it is already demonstrating significant potential.

Our current road map includes utilizing TensorFlow, NLP and supervised machine learning to classify sections of text, extract references and metadata and to supplement our quality control, all targeted to improve the consistency and reliability of our results - and to do it faster and better.

DCL™

Data Conversion Laboratory Inc.

Address	61-18 190th Street Suite 205 Fresh Meadows, NY 11365
Telephone	+1 800.321.2816
Web	www.dclab.com
Twitter	@dclaboratory
LinkedIn	linkedin.com/company/dclab