# Tippanee - Making Web Annotations Dynamically Robust and Semantically Rich

Anonymous Author(s)

## ABSTRACT

Over the last 15 years, the World Wide Web has been steadily progressing towards the era of Web 3.0, evolving from just a vision and research area of a few computer scientists to a vital technology utilized by numerous web service providers and web page owners. More and more websites embed reasonable amount of semantic metadata within their documents; however, there are still massive amounts of unstructured web pages with no metadata at all. Why? The vast majority of metadata available on the web is utilized only by computer programs such as search engines, recommender systems, and expert systems. However, what we need is a next generation of end-user applications that enable us to utilize semantic metadata to lookup, extract and link web content according to our needs and interests – overcoming participation inequalities, breaking the filter bubble, encouraging society-oriented crowdsourcing and so forth. Still, a key hurdle for such applications is mastering anchors in dynamically generated web pages.

In this paper, we present the Tippanee platform that has been designed in response to all of these concerns. Tippanee comes as a lightweight, user-friendly Google Chrome extension that allows users to add, link and share personal annotations hooked onto elements within HTML documents. The system enables end-users to build their own *web of data* by adding semantic descriptions to annotated content utilizing Schema.org vocabulary. All this becomes possible due to Tippanee's novel and particularly robust anchoring algorithm that detects and reattaches anchors on both static and arbitrarily generated web pages, while only storing essential DOM information.

## CCS CONCEPTS

• **Information systems** → **Collaborative and social computing systems and tools**; *Semantic web description languages*; Web interfaces; • **Applied computing** → *Annotation*;

## KEYWORDS

Anchoring, metadata, semantic web, web annotations, Web 3.0

**ACM Reference Format:**
Anonymous Author(s). 2018. Tippanee - Making Web Annotations Dynamically Robust and Semantically Rich. In *Proceedings of The Web Conference (WWW'17).* ACM, New York, NY, USA, Article 4, 9 pages. https://doi.org/10.475/123_4


*WWW'17, April 2018, Lyon, France*

ACM ISBN 123-4567-24-567/08/06…$15.00
https://doi.org/10.475/123_4

## 1 INTRODUCTION

Over the last 15, years the World Wide Web has been slowly but steadily progressing towards the era of Web 3.0, thanks to constant breakthroughs in web technology. Web 3.0 or Semantic Web has evolved from just a vision and research area of a few computer scientists, to a vital technology utilized by numerous web service providers and web page owners [18, 24]. Most websites on the internet embed reasonable amount of semantic metadata within their documents, however, there are still massive amounts of unstructured web pages with no metadata at all. Most metadata available on the web is utilized only by computer programs such as search engines, recommender systems, and expert systems. End-user applications are rare in this field. However end-user applications could enable us to utilize semantic metadata to lookup, extract and link web contents according to our interests [3, 13].

Crowdsourcing platforms, on the other hand, have experienced tremendous growth over the last few years. Platforms such as reCAPTCHA, MTurk and others [1, 7, 29] have demonstrated their viability in a wide spectrum of fields, especially in data collection and analysis. While the power of crowdsourcing lies in the crowd itself, depending on its size and skills, a crowd could execute thousands of tasks in a short span of time [21]. Nevertheless, crowdsourcing platforms do increasingly struggle with the limitations of crowdsourcing. For instance, not all tasks can be easily broken down to simple micro tasks; also tasks that require integration of multiple actors (machines, individuals, and crowd) are not so easy to design or execute [8, 14, 15]. As alternative crowdsourcing media, the social web has definitely emerged as an outstanding way to recruit participants in recent days. Unfortunately, these crowdsourcing experiments rarely fulfill the required characteristics or the expected efficiency [12]. A significant reason behind that is, that merely 17% of all web users actually actively participate on the Web. Around 60% of the web users of only participate passively (primarily by posting pictures, sharing posts, commenting on social media, writing reviews on e-commerce websites, etc.), whereas 23% of the users are lurkers and do not participate at all. The reasons behind such behavior range from age, gender, community to lack of technical skills and the filter-bubble [5, 11].

Now, it might be difficult to solve all of these problems at once; but we are convinced that providing end-users with intuitive and easy-to-use tools to interact with the Web can greatly help to resolve such issues. Annotation tools are one such solution; they allow users to interact to with the Web without the need for technical/coding skills. They allow users to improve and adapt web content, they allow collaboration and act as a medium to reinterpret and enrich the Web [26, 28].

To understand how web annotation platforms enhance user experience, it important to understand how such tools work. To do so, let consider one of the most widely used web annotation tools, *Hypothes.is*. The Hypothes.is project has been started in 2011 and

is based on the *annotation standards for digital documents* developed by the W3C Web Annotation Working Group. The platform provides end-users with a conversation layer over the entire Web, without actually interacting with the underlying site. The platform enables users to create sentence-level notes or critiques on top of news, blogs, scientific articles, and more. The system relies on an efficient approach that stores only the XPath range, string offset pair and the annotated string in order to detect and reattach annotated text. This approach is referred to as *fuzzy anchoring*. A known problem of the approach is that nearly 27% of Hypothes.is' annotations are already *orphaned*, i.e., the annotated web pages no longer contain the annotated text. Moreover, due to constant changes in web content and architecture, 61% of the annotations are in danger of becoming orphans over time [2]. Following the initial proposal for the Open Annotation Data Model [25], the W3C Annotation Working Group submitted it's recommendation for standardization of Web annotation [1]. The recommended Annotation Data Model provides a standard description model and format for Web annotation platforms and web publishers to have a common means of publishing a semantically rich content layer over conventional web pages, enhancing the end-users web experience.

## 1.1 Problem Definition

We argue that the issues of lack of semantic metadata and participation inequality could be solved by an end-user oriented annotation platform. However, such a platform would require a robust anchoring algorithm that could tackle both changes in web content-architecture and dynamically generated web pages. An accurate but space-and time-consuming solution to this problem is to archive pages at the time of annotation. These archived pages could then be compared to the current version of the same web page to detect changes and to identify anchored text using Memory Efficient Tree Edit Distance (METED) [23] or similar algorithms. But this would require additional storage space to archive web pages, and excess of time to match DOM (Document Object Model) trees and to detect and reattach anchors depending upon the number of web elements. This means that even with the algorithm's low time complexity of O (|F|.|G|), the METED algorithm would still take a considerable amount of time from the end-user's point of view.

The Tippanee[2] [3] aims at solving the above-mentioned issues. The system is designed as a lightweight, user-friendly Google Chrome extension that allows users to add, link and share personal annotations hooked onto elements within HTML documents. It enables end-users to build their own *web of data* by adding semantic descriptions to the annotated content utilizing *Schema.org vocabulary*. Annotations are treated as first-class citizens, i.e., Tippanee allows for linking of annotations. An essential contribution is Tippanee's novel and particularly robust anchoring algorithm that detects and reattaches anchors on both static and arbitrarily generated web pages, while only storing essential DOM information.

## 1.2 Research Questions

The research questions tackled in this paper are:

- R1: Can we identify a DOM element in a dynamic web page, even if the content or structure of this element has been changed?
- R2: Are we able to detect and reattach anchors in dynamically generated web pages?
- R3: Can we rescue an orphaned anchor?

The contributions of this paper are the features of the Tippanee platform. We present the semantic features of the system using snapshots of the user interface and discuss the anchoring approach using test cases of dynamically generated web pages. We present experiments comparing our anchoring algorithm against the fuzzy anchoring algorithm. We show that the proposed methodology is more reliable and robust than fuzzy anchoring.

## 1.3 Paper Organization

The paper is organized as follows. Section 2 presents related work that influenced the Tippanee platform, i.e., the Hypothes.is web anchoring platform, anchoring algorithms, the W3C Web Annotation Recommendation and Schema.org. Section 3 presents the Tippanee platform. We explain the platform's architecture, anchoring algorithm, it's semantic feature, it's anchor weaving feature and it's limitations. Section 4 presents the evaluation of the proposed anchoring algorithm, through case-based comparison to the fuzzy anchoring algorithm. Section 5 discusses the legal aspects of DOM storage and web scraping regarding Tippanee. Finally, we provide a Conclusion in Section 6.

# 2 RELATED WORK

Our work is influenced by prior research in areas of Web annotation and the Semantic Web.

## 2.1 Hypothes.is - Web Annotation Platform

Annotation has been recognized as a fundamental notion of hypertext systems since the inception of the WWW [16]. Over the years several Web annotation systems have been designed and deployed. The most recent and most effective ones include Pundit[4], Genius[5] and Hypothes.is[6]. Among these, Hypothes.is has gained most interest and community support as it is a non-profit, free and open-source platform. Based on the open-source JavaScript library Annotator.js[7], Hypothes.is allows users to add sentence-level annotations over web pages. It aims at allowing open critique and collaborative note-taking. The platform can be used either through a Chrome browser extension or via the Hypothes.is website. Users can visit *https://web.hypothes.is/* and paste in the URL of the page they want to annotate. Then users can highlight text and add/read annotations using the sidebar on the right. Users can make their annotations public, share annotations within private groups or can keep them private.

## 2.2 Anchoring Algorithms

*2.2.1 Anchoring using XPath.* Until 2013, Hypothes.is used the anchoring strategy inherited from the Annotator.js project. The

[1] https://www.w3.org/TR/2017/REC-annotation-model-20170223/
[2] Tippanee is the Hindi noun for 'annotation'
[3] https://tinyurl.com/tippanee
[4] http://thepund.it/
[5] https://genius.com/web-annotator
[6] https://web.hypothes.is/
[7] http://annotatorjs.org/

strategy involved using XPath expression of the annotated DOM element and the string offsets inside them (referred to as *RangeSelector*). The strategy only worked for stable documents and contents, i.e., documents for which the structure or content didn't change. Any change in the document structure could change the anchored element's XPath, and render the stored XPath invalid.

*2.2.2 Fuzzy Anchoring.* In order to prevent invalid XPath anchors and to achieve robust reattachment of annotations Hypothes.is currently utilizes a combination of multiple approaches. Contrary to the previous approach that was primarily based on XPath, Hypothes.is now relies on three selectors and four strategies.

For the reference in Section 4, we provide an explanation of these selections and strategies. As can been seen in **Figure 10**, the selectors are:

- **RangeSelector**: a pair of XPath paths pointing to the start and end of the selected content, plus the string offsets inside them.
- **TextPositionSelector**: a pair of string offsets, marking the start and end of the selected text in the character string representing the whole document.
- **TextQuoteSelector**: this selector stores three strings:
  - *exact*: the selected text itself
  - *prefix*: the (32-char long) text immediately before the selected text
  - *suffix*: the (32-char long) text immediately after the selected text

In order to reattach the anchor, the following strategies are utilized: (1) by matching text stored in TextQuoteSelector to the text selection between the start and end XPath expressions stored in the RangeSelector, (2) if the document structure is changed but text content is unchanged, the global character offsets stored in TextQuoteSelector are used to produce text selection, the text is matched to the stored TextQuoteSelector and the anchor is attached, (3) in case both text and document structure are changed, text selection generated using stored prefix and suffix from TextQuoteSelector is matched to text between expected start and end positions from RangeSelector using fuzzy search. If the similarity between the texts is above the accepted threshold, the anchor is reattached; lastly, if all of the previous strategies fail (4) the complete document is searched for saved exact text using fuzzy text search. If the exact text is found, the anchor is attached else the anchor is considered orphaned[8].

The fuzzy text search algorithm utilized in strategies 3 and 4, is often referred to as approximate string matching algorithm and is based on the Bitap matching algorithm [30] and Myer's diff algorithm [19].

## 2.3 Memory Efficient Tree Edit Distance

The TED (Tree Edit Distance) algorithm[31] is used to evaluate the similarity between ordered labeled trees. It is the minimal-cost sequence of node edit operations (insertion, deletion, or relabeling) required to transform one tree into another. Two trees are considered similar if their tree edit distance is below a predefined threshold. The TED problem has a recursive solution and it's best known solutions are dynamic programming implementations with runs times $O(n^4)$ [31], $O(n^3)$ [6], and $O(n^2)$ [22]. These solutions are effective but computationally expensive.

## 2.4 W3C Web Annotation Recommendation

The W3C Web Annotation Working Group published it's recommendations for web anotations in February' 2017. The recommendation includes the Web Annotation Data Model[9], the Web Annotation Vocabulary[10], and the Web Annotation Protocol[11]. The recommended data model is the successor of the W3C Open Annotation Data Model [25]. The model specification describes a structured format through which annotations can be shared and reused across different hardware and software platforms. The basic data model has three parts: a body, a target and a relation (stating how the body is related to the target). These parts together form an annotation. The nature of the relationship between the body and the target may change depending upon the intention of the annotation; however, the body will always be 'about' the target.

For example, a basic annotation model[12] looks like:

```
{
"@context": "http://www.w3.org/ns/anno.jsonld",
"id": "http://example.org/anno1",
"type": "Annotation",
"body": "http://example.org/post1",
"target": "http://example.com/page1"
}
```

The W3C Web Annotation Vocabulary defines the set of RDF classes, predicates and named entities that are used in the W3C Web Annotation Data Model. The vocabulary is divided into two sections: (1) the terms defined in the Web Annotation ontology, and (2) the terms from other ontologies used in the model.

Finally, in accordance with the W3C Web Architecture and REST principles, the W3C Web Annotation Protocol describes the transport mechanisms for creating and managing annotations across systems and operating systems.

Currently, the W3C Web Annotation Working Group is working on potential approaches for embedding Web annotations into HTML documents[13].

## 2.5 Schema.org

Schema.org was founded by Google, Microsoft, Yahoo and Yandex in 2011. It provides clean and stable semantic vocabulary supported by major search engines. The goal of the Schema.org community is to provide a uniform schema across a broad range of topics including people, places, events, products, offers, reviews, ratings and so on. The provided vocabulary can be used in RDFa, Microdata and JSON-LD formats and it helps to describe entities and relations between entities and actions. The vocabulary is currently used in over 10 million websites and its shared collection of schemas allows

[8] https://web.hypothes.is/blog/fuzzy-anchoring/

[9] https://www.w3.org/TR/annotation-model/
[10] https://www.w3.org/TR/annotation-vocab/
[11] https://www.w3.org/TR/annotation-protocol/
[12] https://www.w3.org/TR/2017/REC-annotation-model-20170223/
[13] https://www.w3.org/TR/annotation-html/

programmers and webmasters to add and utilize recognized semantic markup [10, 17]. The core vocabulary of Schema.org currently consists of 597 Types, 867 Properties, and 114 Enumeration values.

For instance, the following semantic markup in JSON-LD format describes a banana bread recipe[14],

```
{
"@context": "http://schema.org",
"@type": "Recipe",
"author": "John Smith",
"cookTime": "PT1H",
"prepTime": "PT15M",
"recipeIngredient": ["3 or 4 ripe bananas, smashed" ...],
"name": "Mom's World Famous Banana Bread",
"recipeInstructions": "Preheat the oven to 350 degrees ..."
}
```

## 3 THE TIPPANEE PLATFORM

The proposed Tippanee platform utilizes a novel pattern-based anchoring system that attempts to solve the fuzzy anchoring algorithm's orphan problem while using less space and time on the user end. The system does so by storing the DOM elements, attributes and the contents of an anchored element sequentially. The system stores all essential information about the anchored element and its children. This way it ensures that in case the element gets orphaned it is still possible for the system to reconstruct the orphaned anchor. The system preserves the meaning of the anchored text by storing children or siblings depending upon the scope of the anchored text. The proposed anchoring algorithm uses sequential tree matching to find possible matches and mismatches from a smaller pool of similar elements based on stored DOM attributes. It then identifies the anchored element on the basis of matches and mismatches. This means that the system still detects an anchor – even if its contents or layout is changed.

### 3.1 Architecture

The current version of the Tippanee platform is based on client-server architecture. The Tippanee Chrome browser extension acts as the client side and allows users to target, highlight and share annotated text and images in web documents. Since the anchoring algorithm has been built into the extension itself, users can use the extension as a stand-alone tool. If used as a stand-alone tool, users can use the annotation features of the extension, but cannot share these annotations. To share annotations users must have a Tippanee account. Users can create an account, login via the Tippanee extension and then they can share annotations with individual users and groups. The server primarily acts as a messaging server, forwarding shared annotations to users. The server also stores annotations, so that users may access them over different devices.

### 3.2 Anchoring Algorithm

*3.2.1 Selecting Anchors.* Tippanee's anchoring algorithm is designed to deal with dynamic web content; therefore it focuses on storing the layout and context of the annotated target. The algorithm analyses the target DOM as a tree, where the target element is the root and *childNodes* are tree nodes; whereas HTML *#text* nodes are leaves. The system only stores a predefined list of properties that are available for any given HTML element/node. For a leaf node, i.e., *#text* node, Tippanee's anchoring algorithm stores only the node's *nodeName* and *nodeValue*. For other tree nodes Tippanee's anchoring algorithm stores the HTML element's *id*, *nodeName*, *className*, *alt*, *dataset*, *href* and *src*. The system also calculates and stores the depth of all elements/nodes with respect to the target element (root), referred to as *nodeDepth*. The anchoring algorithm traverses the target DOM sequentially; by doing so it stores the target DOM tree in prefix notation. As the target DOM is traversed, it's above-mentioned properties are stored as an array of JSON (JavaScript Object Notation) objects. Hence, storing the annotated text's surrounding content, preserving it's context even if the annotation is orphaned.

Tippanee's anchoring algorithm uses the following strategies for anchoring different types of selections:

- **Annotating an element**: If the user annotates an element (i.e., all the text within an element), the anchoring algorithm selects the target DOM and transforms it into the above-mentioned JSON object.
- **Annotating a *#text* node within an element**: If the user annotates some *#text* node within an element (but not all the text), the anchoring algorithm selects the target DOM and transforms it into the above-mentioned JSON object. However in this case, it adds an extra boolean property *annotated* to the selected *#text* node.
- **Annotating text in two or more elements**: If the user annotates texts from multiple elements, the algorithm selects the common ancestor element as the target DOM and generates it's DOM object. Again, the *annotated* property is added to the selected *#text* nodes.
- **Annotating substring within a *#text* node**: If the user annotates a substring from within a *#text* node, the system additionally stores the selected substring and it's starting and ending offsets with the *#text* node.

The following JSON object array is an example of the anchor data extracted by Tippanee's anchoring algorithm. As can be seen in **Figure 1**, the user selected the text 'to Wikipe'. Since the selected content ranges from inside the *#text* node containing 'Welcome to ' to inside the *hyperlink* element containing 'Wikipedia' as can be seen in **Figure 2**. The anchoring algorithm targets the common ancestor which is the *div* element in this case.

```
[
{ "nodeDepth": 0, "nodeName": "DIV"},
{ "nodeDepth": 1, "nodeName": "#text", "nodeValue": "Welcome to ",
"annotated": true, "startOffset": 8, "endOffset": 0},
{ "href": "https://en.wikipedia.org/wiki/Wikipedia", "nodeDepth": 1,
"nodeName": "A"},
{ "nodeDepth": 2, "nodeName": "#text", "nodeValue": "Wikipedia",
"annotated": true, "startOffset": 0, "endOffset": 3},
{ "nodeDepth": 1, "nodeName": "#text", "nodeValue": ","}
]
```

[14] http://schema.org/Recipe


**Figure 1: Selected content on Wikipedia homepage, highlighted in blue.**

```
<div style="font-size:162%; padding:.1em;">
    Welcome to
        <a href="/wiki/Wikipedia" title="Wikipedia">
        Wikipedia
        </a>
    ,
</div>
```

**Figure 2: HTML code segment of the selected element from Wikipedia homepage.**

*3.2.2 Reattaching Anchors.* In order to reattach an anchor, the algorithm first identifies possible target elements using Document Object Methods (*getElementById()*, *getElementsByClassName()*, *getElementsByTagName()*). The returned DOM elements are then compared to the anchor's JSON object. The algorithm sequentially traverses the returned DOM element and its children. If the stored JSON object attributes match the attributes of the corresponding DOM element, this is considered a match. Elements that do not have the same attributes are considered a mismatch. In order to compare *#text* JSON objects to respective *#text* nodes, the algorithm uses approximate string matching[19, 30]. In the current version of Tippanee we use a JavaScript implementation of approximate string matching available on github[15]. The JavaScript program matches the two texts and scores their similarity with a value between 0 and 1, with 1 being a perfect match. After combining the match-mismatch score of DOM elements with the similarity score of the *#text* nodes, the algorithm calculates a similarity index for each returned DOM element. The algorithm then selects the DOM element that is the most similar to the stored anchor and reattaches the annotation to it. In case that no DOM element qualifies the minimum similarity threshold, the system assumes that the annotation is orphaned. Via the similarity threshold limit, it is possible to increase the chance that the system will only attach anchors that are perfect matches. Decreasing the threshold too much may cause the selection of incorrect targets. Finally, if an annotation is orphaned, the annotation can still be reconstructed in its original form and can be viewed in the system's *orphaned annotation*.

## 3.3 Semantics

To describe the annotated text the Tippanee platform provides users with a drop-down menu with Schema.org vocabulary. Some of these properties are: about, citation, contentRating, dateModified, etc. When a user adds semantic annotation to an anchor, the system generates a JSON-LD (JSON for Linked Data) object and adds it to the anchor's JSON object. Users can review and change the stored semantic metadata or use this metadata to search through stored annotations. In the current version, this feature can only be used by users, and not by machines.

## 3.4 Weaving Anchors

During anchor selection, the algorithm generates a unique identification key for each JSON object anchor. The system allows users to link annotations. When a user links an annotation to another, it creates a directed link between the source and the target. The system uses these links to provide users a graph visualization of all annotations using the visualization library vis.js[16]. This visualization as shown in **Figure 3** is what we refer to as user's *web of data*.

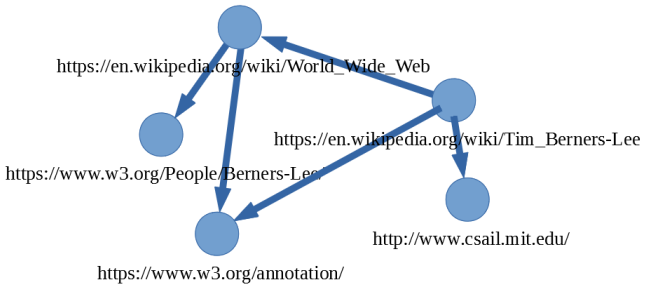


**Figure 3: An illustrations of Tippanee's Web of Data.**

## 3.5 Limitations

Our work has the following limitations: (1) If a selected anchor exists more than once within the DOM and carries exactly the same tree structure, the anchoring algorithm would identify two or more elements with same similarity index. In such a scenario, the system would always reattach the element that occurs first in the DOM tree with a risk that the annotation might be attached to the wrong anchor. (2) Restructuring the web page DOM and simultaneously changing the content, might lead to orphaned annotations.

# 4 EVALUATION

In order to evaluate the robustness of Tippanee's anchoring algorithm, we have conducted a comparative evaluation study of the system against Hypothes.is' fuzzy anchoring algorithm. For this, we choose two specific cases: first, a dynamic financial information website where the document structure remains unchanged, but the content gets updated frequently (in seconds) and second, a dynamic information portal where the content remains same, but the document structure changes as the user clicks on different options. As an important detail, it is necessary to know that in both cases the URLs remain unchanged.

[15] https://github.com/Glench/fuzzyset.js

[16] http://visjs.org/

### 4.1 Case 1 (Dynamic Content - Static Structure)

For the first test case, we evaluated the Tippanee and the Hypothes.is platforms on a dynamic financial information website *https://www.marketwatch.com/*. The *marketwatch* website uses JavaScript on the client-side and ASP (Active Server Pages) and XML (eXtensive Markup Language) on the server-side with AJAX as client-side scripting framework. **Figure 5** shows a section of the website's homepage prior to annotation. The section's DOM tree viewed in Google Chrome Inspector is shown in **Figure 6**. During the evaluation the DOM structure of the webpage remained unchanged; however, the text within the elements changed whenever new text was pushed by the server. **Figure 7** shows the DOM structure of one such element.

As explained in section 2.2 and shown in **Figure 4**, the Hypothes.is' fuzzy anchoring algorithm relies only on the XPath and the text of the annotated content. When elements such as that shown in **Figure 5** are annotated using Hypothes.is, they only stay anchored until the majority of the annotated content is unchanged. When we annotated these elements on the Hypothes.is platform, it was observed that the annotated content changed within minutes, as can be seen in **Figure 8**. In the figure, the text highlighted in yellow represents the content where anchors could be attached, whereas the text that is not highlighted represents changed content. Because the annotated content changes frequently, the annotated text will be orphaned eventually. Also since all annotated text is stored as string as can been seen in **Figure 9**, there is no way to reconstruct the annotated element's DOM tree.


```
"selector": [
  {
    "conformsTo":
    "https:\/\/tools.ietf.org\/html\/rfc3236",
    "type": "FragmentSelector",
    "value":
    "gigya-share-btns-1_gig_containerParent"
  },
  {
    "endContainer": "\/div[1]\/div[3]\/article[1]
    \/div[2]\/header[1]\/h1[1]",
    "startContainer": "\/div[1]\/div[3]\/article[1]
    \/div[2]\/header[1]\/h1[1]",
    "type": "RangeSelector",
    "startOffset": 0,
    "endOffset": 29
  },
  {
    "type": "TextPositionSelector",
    "end": 2068,
    "start": 2039
  },
  {
    "exact": "Climate change might be worse",
    "prefix": " \n          NewsScience    \n    ",
    "type": "TextQuoteSelector",
    "suffix": " than thought after scientists f"
  }
]
```


**Figure 4: Selector data of Hypothes.is' annotation described in JSON format.**


**Figure 5: Target element (Case 1).**

```
▼<div class="element element--markets">
  ▼<ul class="tabs" data-track-query="a[data-track-code]" style>
    ▶<li class="tab__item option " style>…</li>
    ▶<li class="tab__item option is-selected">…</li>
    ▶<li class="tab__item option ">…</li>
    ▶<li class="tab__item option ">…</li>
    ▶<li class="tab__item option ">…</li>
    ▶<li class="tab__item option ">…</li>
      <!-- Time Dropdown -->
    ▶<li class="tab__item dropdown">…</li>
    </ul>
  ▼<div class="markets" data-track-query="a[data-track-code]" style>
    ▶<div class="markets__table">…</div>
    ▶<div class="markets__chart is-loaded" style>…</div>
    </div>
  </div>
```

**Figure 6: The target element's DOM tree (Case 1), as viewed in Google Chrome Inspector.**

```
▼<td class="table__cell price">
    <bg-quote class="ignore-
    color" field="last" format=
    "0,0.00" channel="/quotes/
    zigman/2380150/delayed">
    395.22</bg-quote>
  </td>
```

**Figure 7: Child element of the target element (Case 1), as viewed in Google Chrome Inspector.**

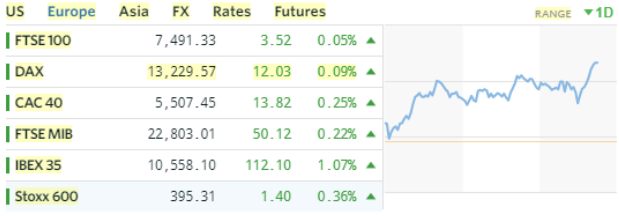


**Figure 8: Target element for Case 1 annotated using Hypothes.is.**

In contrast to Hypothes.is' approach, Tippanee's anchoring algorithm relies on the DOM structure of the selected element. The algorithm sequentially stores the anchored element and it's children in an array of JSON objects, as can be seen in **Figure 10**. Using the anchor reattachment strategies discussed in Section 3.2.2, Tippanee's anchoring algorithm hooks on to the most similar element within the webpage DOM. Since there is no change in the DOM structure, the algorithm remains hooked on to the actual target element. The anchored DOM element is highlighted in blue background as can be seen in **Figure 11** .

## 4.2 (Dynamic Structure - Static Content)

For the second test case, we chose our University Study Information Portal. The portal uses JavaScript on client-side and PHP on server-side with AJAX as client-side scripting framework. We used the current stable version of the portal for evaluation, as we knew that the content of the selected webpage would remain static during the evaluation. However, the fact that the structure of the webpage changes on the fly, is an important issue for annotation platforms**Figure 12**. This means that simple page interactions could change the structure of DOM sections. Content annotated in a page state/view could be assumed orphaned for a different page state/view, as can be seen in **Figure 13**.

Unlike the Hypothes.is platform that searches for anchors only during the page load, the Tippanee platform checks for anchors each time the browser receives an AJAX response. Tippanee's anchoring algorithm checks the page state/view after each user interaction, hence reduces the occurrence of orphaned annotations.

# 5 DISCUSSION OF LEGAL ASPECTS

Web scraping, caching and archiving has often been viewed as severe ethical issue on the World Wide Web. Despite the fact that, over the last couple of years the Open Data Movement has received tremendous support from governments, research institutes, laboratories and libraries. Nevertheless, the idea of a truly open web seems to be a far-fetched notion. Several national libraries around

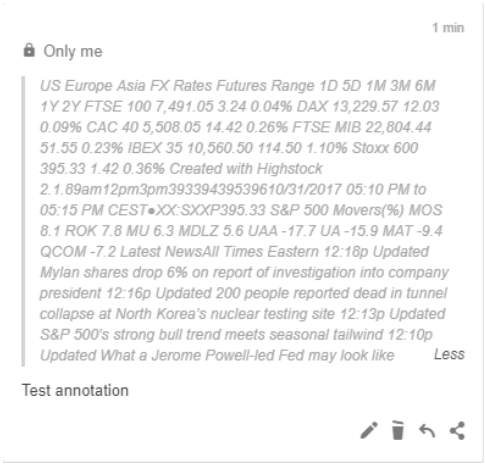


**Figure 9: Representation of textual content (Case 1) annotated using Hypothes.is.**


```
{
    "className": "element element--markets",
    "nodeDepth": 0,
    "nodeName": "DIV"
},
{
    "nodeDepth": 1,
    "nodeName": "#text",
    "nodeValue": "\n        "
},
{
    "className": "tabs",
    "dataset": "{\"trackQuery\":\"a[data-track-code]\"}",
    "nodeDepth": 1,
    "nodeName": "UL"
}....
{
    "dataset": "{\"trackCode\":\"MW_Header_Market Data_Date Range_1D\"}",
    "nodeDepth": 4,
    "nodeName": "OPTION"
}....
{
    "dataset": "{\"trackCode\":\"MW_Header_Market Data_Date Range_5D\"}",
    "nodeDepth": 4,
    "nodeName": "OPTION"
}
```


**Figure 10: Anchor data of target element (Case 1) annotated using Tippanee.**

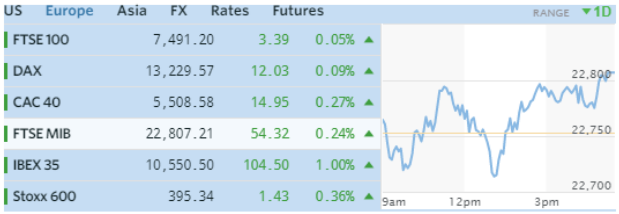


**Figure 11: Target element for Case 1 annotated using Tippanee.**

the world have been supporting the concept of digital preservation and archiving, but most of these programmes are hindered or affected by legal concerns, i.e., copyright infringement and piracy acts[27], e.g., [4, 9, 20].

While designing the Tippanee platform we studied these legal obligations in order to realize the scope of our system. Although web scraping, caching or archiving requires explicit permission from the web page owners, we argue that our system does not violate any piracy or copyright concerns, because of the following reasons: first, the system stores the context of the annotated content only to ensure that users do not lose their annotations and to improve the usability of orphaned anchors. In order words, content is saved primarily for the interest of the end user. Second, the saved content is utilized solely for reattaching anchors and has no other commercial purpose whatsoever. Third, the context is stored exactly as it was published by the owner. And, the source of the content is clearly stated in the anchor's JSON object, meaning that the content is always linked to the source and the link is visible to the end user. Lastly, enabling users to add semantic markup to annotations and creating links between annotations, helps in enhancing the user's web experience by encouraging semantic content creation

```
▼<div id="opvorm"> == $0
  ▶<div id="tulemusvorm" style=
  "display: block;width: 100%;">…</div>
  </div>
 </div>
</div>
▶<div style="display: block; height: 90px;
width: 20px;   ">…</div>
▶<div id="u5" class="ax_shape">…</div>
 <script>
          initois();
        </script>
▶<div class="modal">…</div>
 <div class="cssload-loader"></div>
</div>

▼<div id="opvorm"> == $0
  ▶<div id="yldandmed">…</div>
   <div id="otsing">
   </div>
  ▶<div style="display: block; height:
  20px;   ">…</div>
  ▶<div style="display: block; height:
  20px;   ">…</div>
  ▶<div id="op_valmis" style="display:
  block!important;">…</div>
  </div>
 </div>
</div>
▶<div style="display: block; height: 90px;
width: 20px;   ">…</div>
```

**Figure 12: Same target element with different DOM trees (Case 2)**

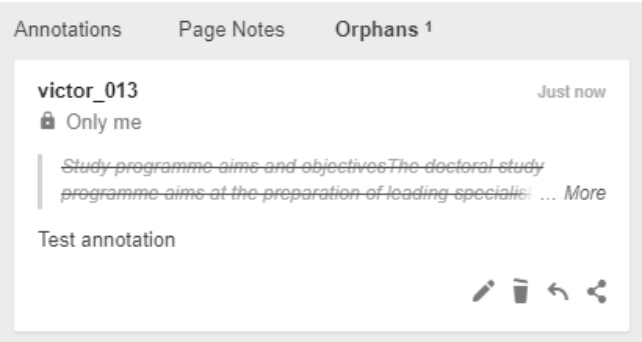


**Figure 13: Orphaned target element (Case 2) annotated using Hypothes.is.**

and participation, therefore reinforcing and improving the social aspects of the Web.

## 6 CONCLUSION

In this paper, we have presented the Tippanee platform that allows for semantically rich annotations to dynamically evolving web content. Given the current interest in emerging Web annotation tools and the importance of semantic Web technology, it is just a consequent step to team together these two notions. Albeit the combination of these notions opens a wide design space for a whole class of next-generation annotation tools, the key challenge for making such tools a success is in providing robust anchoring mechanisms.

With Tippanee's anchoring we have provide mechanisms to (1) identify a DOM element in a dynamic web page, even if the element's content or structure has been changed, to (2) detect and reattach anchors in dynamically generated web pages, and (3) to rescue an orphaned anchor by caching its context and, therefore, have positively answered all three research questions posed in Section 1.1. We could positively evaluate Tippanee's anchoring with respect to typical problematic cases of dynamically generated web pages and, as part of this evaluation, also provided a comparison with the current state-of-the-art in anchoring, i.e., fuzzy anchoring.

Web 2.0 changed our point of view regarding the World Wide Web. Consumers become producers. This is a fundamental change that is still with us, and that involves the attitude of Web users as well as the availability of appropriate platforms and tools. However, even if consumers become producers, we are still trapped in a consumer/producer divide. Up to the platforms that realize some models of collaborative content creation, until now, Web 2.0 was not about a complete deconstruction of the Web space. As simple as plain Web annotation tools might look, they have the potential to change this. In a sense, they push Web 2.0 to the extreme as they create a Web on top of the Web. If Web annotation was merely about placing some comments, it could not become a game changer. However, what we have in mind is rich Web annotation, such that annotations become first-class and inter-linked. A promising means to make Web annotation rich is to make them *semantically* rich. Again, Web 3.0 changed our point of view. Content becomes data, i.e., open linked data. Tippanee values the Web 3.0 vision and is designed for integration with the leading Web 3.0 community platform and process Schema.org.

## REFERENCES


[1] Noushin Rezapour Asheghi, Serge Sharoff, and Katja Markert. 2016. Crowdsourcing for web genre annotation. *Language Resources and Evaluation* 50, 3 (01 Sep 2016), 603–641. https://doi.org/10.1007/s10579-015-9331-6

[2] Mohamed Aturban, Michael L. Nelson, and Michele C. Weigle. 2015. Quantifying Orphaned Annotations in Hypothes.is. In *Research and Advanced Technology for Digital Libraries: 19th International Conference on Theory and Practice of Digital Libraries, TPDL 2015, Poznań, Poland, September 14-18, 2015, Proceedings*. Springer International Publishing, Cham, 15–27. https://doi.org/10.1007/978-3-319-24592-8_2

[3] David R. Brake. 2014. Are We All Online Content Creators Now? Web 2.0 and Digital Divides. *Journal of Computer-Mediated Communication* 19, 3 (2014), 591–609. https://doi.org/10.1111/jcc4.12042

[4] Jhonny Antonio Pabón Cadavid. 2014. Copyright Challenges of Legal Deposit and Web Archiving in the National Library of Singapore. *Alexandria* 25, 1-2 (2014), 1–19. https://doi.org/10.7227/ALX.0017 arXiv:https://doi.org/10.7227/ALX.0017

[5] Namkee G Choi and Diana M Dinitto. 2013. The Digital Divide Among Low-Income Homebound Older Adults: Internet Use Patterns, eHealth Literacy, and Attitudes Toward Computer/Internet Use. *Journal of Medical Internet Research* 15, 5 (Feb 2013). https://doi.org/10.2196/jmir.2645

[6] Erik D. Demaine, Shay Mozes, Benjamin Rossman, and Oren Weimann. 2009. An Optimal Decomposition Algorithm for Tree Edit Distance. *ACM Trans. Algorithms* 6, 1, Article 2 (Dec. 2009), 19 pages. https://doi.org/10.1145/1644015.1644017

[7] Richard Eckart de Castilho, Éva Mújdricza-Maydt, Seid Muhie Yimam, Silvana Hartmann, Iryna Gurevych, Anette Frank, and Chris Biemann. 2016. A Web-based Tool for the Integrated Annotation of Semantic and Syntactic Structures. In *Proceedings of the workshop on Language Technology Resources and Tools for Digital Humanities (LT4DH) at COLING 2016*. 76–84.

[8] Keelan Evanini, Derrick Higgins, and Klaus Zechner. 2010. Using Amazon Mechanical Turk for Transcription of Non-native Speech. In *Proceedings of the NAACL HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk (CSLDAMT '10)*. Association for Computational Linguistics, Stroudsburg, PA, USA, 53–56. http://dl.acm.org/citation.cfm?id=1866696.1866704

[9] Lachlan Glanville. 2010. Web archiving: ethical and legal issues affecting programmes in Australia and the Netherlands. *The Australian Library Journal* 59, 3 (2010), 128–134. https://doi.org/10.1080/00049670.2010.10735999

arXiv:http://dx.doi.org/10.1080/00049670.2010.10735999

[10] R. V. Guha, Dan Brickley, and Steve Macbeth. 2016. Schema.Org: Evolution of Structured Data on the Web. *Commun. ACM* 59, 2 (Jan. 2016), 44–51. https://doi.org/10.1145/2844544

[11] Eszter Hargittai and Aaron Shaw. 2015. Mind the skills gap: the role of Internet know-how and gender in differentiated contributions to Wikipedia. *Information, Communication & Society* 18, 4 (2015), 424–442. https://doi.org/10.1080/1369118X.2014.957711 arXiv:http://dx.doi.org/10.1080/1369118X.2014.957711

[12] Christian Pieter Hoffmann, Christoph Lutz, and Miriam Meckel. 2015. Content creation on the Internet: a social cognitive perspective on the participation divide. *Information, Communication & Society* 18, 6 (2015), 696–716. https://doi.org/10.1080/1369118X.2014.991343 arXiv:http://dx.doi.org/10.1080/1369118X.2014.991343

[13] D. R. Karger. 2014. The Semantic Web and End Users: What's Wrong and How to Fix It. *IEEE Internet Computing* 18, 6 (Nov 2014), 64–70. https://doi.org/10.1109/MIC.2014.124

[14] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E. Kraut. 2011. CrowdForge: Crowdsourcing Complex Work. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 43–52. https://doi.org/10.1145/2047196.2047202

[15] Anand Kulkarni, Matthew Can, and Björn Hartmann. 2012. Collaboratively Crowdsourcing Workflows with Turkomatic. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work (CSCW '12)*. ACM, New York, NY, USA, 1003–1012. https://doi.org/10.1145/2145204.2145354

[16] Catherine C. Marshall. 1998. Toward an Ecology of Hypertext Annotation. In *Proceedings of the Ninth ACM Conference on Hypertext and Hypermedia : Links, Objects, Time and Space—structure in Hypermedia Systems: Links, Objects, Time and Space—structure in Hypermedia Systems (HYPERTEXT '98)*. ACM, New York, NY, USA, 40–49. https://doi.org/10.1145/276627.276632

[17] P. Mika. 2015. On Schema.org and Why It Matters for the Web. *IEEE Internet Computing* 19, 4 (July 2015), 52–55. https://doi.org/10.1109/MIC.2015.81

[18] Matt Moore. 2012. The semantic web: an introduction for information professionals. *The Indexer* 30, 1 (2012), 38–43. http://www.ingentaconnect.com/content/index/tiji/2012/00000030/00000001/art00007

[19] Eugene W. Myers. 1986. AnO(ND) difference algorithm and its variations. *Algorithmica* 1, 1-4 (1986), 251âĂŞ266. https://doi.org/10.1007/bf01840446

[20] Jhonny Antonio Pabón Cadavid, Johnkhan Sathik Basha, and Gandhimani Kaleeswaran. 2013. Legal and technical difficulties of web archival in Singapore. (2013).

[21] Stefano Tranquillini Pavel Kucherbaev, Florian Daniel and Maurizio Marchese. 2016. Crowdsourcing Processes: A Survey of Approaches and Opportunities. *IEEE Internet Computing* 20, 2 (2016), 50–56. https://doi.org/doi.ieeecomputersociety.org/10.1109/MIC.2015.96

[22] Mateusz Pawlik and Nikolaus Augsten. 2011. RTED: A Robust Algorithm for the Tree Edit Distance. *Proc. VLDB Endow.* 5, 4 (Dec. 2011), 334–345. https://doi.org/10.14778/2095686.2095692

[23] Mateusz Pawlik and Nikolaus Augsten. 2014. *A Memory-Efficient Tree Edit Distance Algorithm.* Springer International Publishing, Cham, 196–210. https://doi.org/10.1007/978-3-319-10073-9_16

[24] René Peinl. 2016. Semantic Web: State of the Art and Adoption in Corporations. *KI - Künstliche Intelligenz* 30, 2 (01 Jun 2016), 131–138. https://doi.org/10.1007/s13218-016-0425-0

[25] Robert Sanderson, Paolo Ciccarese, and Herbert Van de Sompel. 2013. Designing the W3C Open Annotation Data Model. In *Proceedings of the 5th Annual ACM Web Science Conference (WebSci '13)*. ACM, New York, NY, USA, 366–375. https://doi.org/10.1145/2464464.2464474

[26] V. Vasudevan and M. Palmer. 1999. On Web annotations: promises and pitfalls of current Web infrastructure. In *Proceedings of the 32nd Annual Hawaii International Conference on Systems Sciences. 1999. HICSS-32. Abstracts and CD-ROM of Full Papers*, Vol. Track2. 9 pp.–. https://doi.org/10.1109/HICSS.1999.772663

[27] Richard S. Vermut. 1996-1997. File Caching on the Internet: Technical Infringement or Safeguard for Efficient Network Operation. *Journal of Intellectual Property Law* 4 (1996-1997), 273.

[28] Niels-Oliver Walkowski. 2016. The Landscape of Digital Annotation and Its Meaning. (2016).

[29] Shomir Wilson, Florian Schaub, Rohan Ramanath, Norman Sadeh, Fei Liu, Noah A. Smith, and Frederick Liu. 2016. Crowdsourcing Annotations for Websites' Privacy Policies: Can It Really Work?. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 133–143. https://doi.org/10.1145/2872427.2883035

[30] Sun Wu and Udi Manber. 1992. Fast Text Searching: Allowing Errors. *Commun. ACM* 35, 10 (Oct. 1992), 83–91. https://doi.org/10.1145/135239.135244

[31] K. Zhang and D. Shasha. 1989. Simple Fast Algorithms for the Editing Distance Between Trees and Related Problems. *SIAM J. Comput.* 18, 6 (Dec. 1989), 1245–1262. https://doi.org/10.1137/0218082