Enhancing General Hypertext Editors with Automatic Supplemental Content Suggestions using a Semantic Knowledge Graph [*]

David Hyland-Wood
Ephox Corporation
225 Montague Road, Level 4
West End, Queensland, Australia
david.wood@ephox.com

Anna Harrison
Ephox Corporation
225 Montague Road, Level 4
West End, Queensland, Australia
anna.harrison@ephox.com

Ben Kolera
Ephox Corporation
225 Montague Road, Level 4
West End, Queensland, Australia
ben.kolera@ephox.com

ABSTRACT

Many types of writing, such as blog posts and business documents, could benefit from supplemental explanatory material to provide context, background, and additional information for exploration. The World Wide Web provides huge quantities of such supplementary material, but it is not generally suitable for inclusion in new writing due to issues of licensing, copyright, formatting, or syntax. Exceptions include clearly identifiable content meant to be embedded in hypertext documents, such as YouTube videos, and Twitter feeds. We demonstrate how the textual content in a general purpose hypertext editor may be used to automatically identify relevant supplementary material via artificial intelligence (AI) processes, and that material may be presented to a document author for possible inclusion in whole or in part. User interface elements are provided to allow an author to refine content suggestions with a minimum of input.

CCS Concepts

•Computing methodologies → Semantic networks; •Human-centered computing → Empirical studies in interaction design; •Social and professional topics → *Socio-technical systems;* **•Applied computing →** *Hypertext / hypermedia creation;*

Keywords

ACM proceedings; RDF; knowledge graph; Linked Data; IBM Watson

1. INTRODUCTION

Many types of writing, such as blog posts and business documents, could benefit from supplemental explanatory material to provide context, background, and additional information for exploration. The World Wide Web provides huge quantities of such supplementary material, but it is not generally suitable for inclusion in new writing due to issues of licensing, copyright, formatting, or syntax. Exceptions include clearly identifiable content meant to be embedded in hypertext documents, such as YouTube videos, and Twitter feeds.

We wanted to determine whether the textual content in a general purpose hypertext editor may be used to automatically identify relevant supplementary material via artificial intelligence (AI) processes, and whether that material may be presented to a document author for possible inclusion in a document in a useful manner.

The techniques necessary for such an exploration are now generally available. Linguistic parsing of text content may be conducted relatively easily using generally available tools, at least in many European languages including English. Concept identification from the results of linguistic parsing has been similarly successful.

Concepts may be matched to relevant additional information by querying a graph of information in the Resource Description Framework (RDF) data model[3]. Such information form "knowledge graphs" of related information. The first general purpose knowledge graph of scale was Freebase[2], which became the basis for the Google Knowledge Graph[8]. Web search engines such as Google, Bing, and Yandex now use knowledge graphs to provide summaries of relevant information related to search terms. It is exactly this functionality that we wanted to add to our general purpose hypertext editors.

TODO: REFERENCES: [4].

2. ARCHITECTURE

We wrapped one of our JavaScript hypertext editors, Textbox.io, to include a sidebar for the presentation of supplemental material. The sidebar is a simple HTML component that accepts HTML fragments for display and possible drag-and-drop of components into the editor?s content pane. JavaScript is used to query on online service to search for relevant information based upon the current contents of the editor.

Textual content is written into the editor by a document author. That content is uploaded to an Ephox cloud service for analysis (Figure 1). The Ephox service sends the content to another service to parse the content, and extract con-

[*]A full version of this paper is available at `TODO: Ephox/personal URL`

WWW 2017 Perth, Western Australia, Australia
© 2016 ACM. ISBN 978-1-4503-2138-9...$15.00
DOI: 10.1145/1235

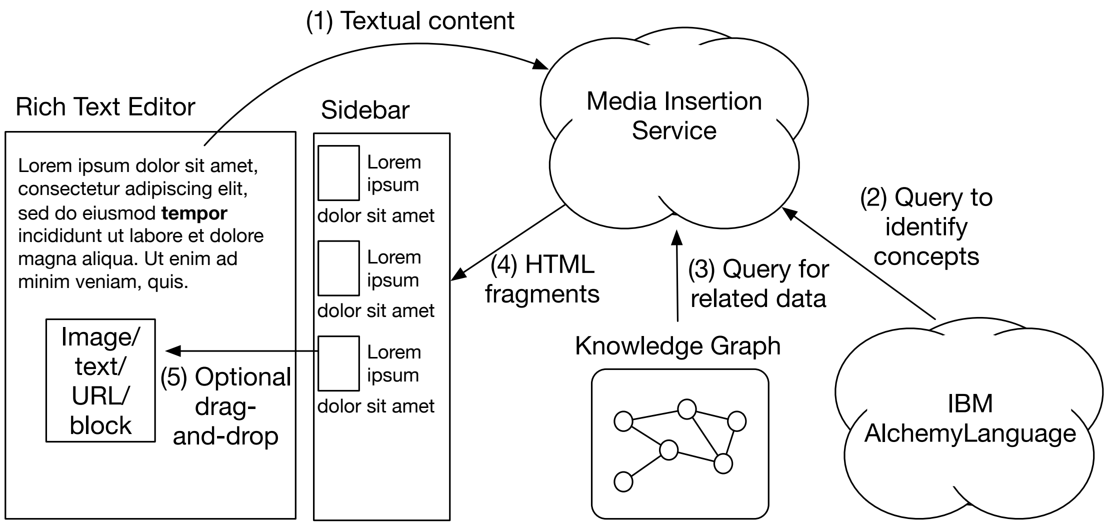
(1) Textual content
Rich Text Editor
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod **tempor** incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis.
Image/ text/ URL/ block
(5) Optional drag-and-drop
Sidebar
Lorem ipsum dolor sit amet
Lorem ipsum dolor sit amet
Lorem ipsum dolor sit amet
Media Insertion Service
(4) HTML fragments
(3) Query for related data
(2) Query to identify concepts
Knowledge Graph
IBM AlchemyLanguage

Figure 1: Data flow from editor to sidebar via enhancement services.

cepts. We used IBM?s online AlchemyLanguage service[1] for linguistic parsing and concept identification, although any number of similar services could have been used for those purposes. One advantage of the AlchemyLanguage API is that its concept tagging service[2] returns DBpedia[1] URIs for each identified concept.

DBpedia URIs conveniently serve as indexes into an RDF knowledge graph. To implement the knowledge graph service, we used an Apache Tomcat[3] Web application server with a RDF4J[4] version 2.0 RDF database engine. Relevant portions of DBpedia were sharded into six RDF4J databases to speed both data indexing and assist query performance. DBpedia topical concepts, SKOS[6] categories, labels, image URLs, short abstracts, categories and their labels, transitive instance types, mapping-based objects and literals, and person data were used for our initial experiments. We limited our DBpedia extracts to the English language due to the language limitations of the AlchemyLanguage concept tagging service.

Two SPARQL 1.1[5] queries against the knowledge graph were used to construct supplemental material representations:

- The first SPARQL query is used to identify the type of a concept (i.e. each concept will be either a person, place, organization, movie, species, product, or a general concept).
- The second SPARQL query is used to identify available data about a specific concept, as anticipated by its type.

Thus, a concept of type ?person? might have a birth date and a birth place. A place might have area and a population estimate, etc. Most data elements for any given type are considered to be optional.

[1]`http://www.alchemyapi.com/products/alchemylanguage/`
[2]`http://www.alchemyapi.com/products/alchemylanguage/concept-tagging`
[3]`https://tomcat.apache.org/`
[4]`http://rdf4j.org/`

The results of the second query are used to fill a type-specific HTML template. An HTML fragment is produced for each concept, and any empty divs are removed. The completed fragment is returned to the editor?s sidebar for display.

SPARQL 1.1 SERVICE clauses were used to execute federated queries[7] across the various databases making up the knowledge graph.

We chose to construct our own knowledge graph instead of using one provided by the AlchemyLanguage service. We wish to eventually refine and contextualize the contents of the knowledge graph over time to provide industry- or even customer-specific data. We also wanted to be able to swap linguistic parsing and concept tagging services if desired at a later time.

3. USAGE

TODO: Anna? How did people approach the sidebar? How did they use it? Did anything unexpected happen during use?

TODO: User interface elements are provided to allow an author to refine content suggestions with a minimum of input.

4. EVALUATION

TODO: Anna to conduct user testing to determine an objective basis for the value provided.

5. CONCLUSIONS AND FURTHER WORK

We have demonstrated that the automatic suggestion of reuseable supplementary information in a general purpose hypertext editor can assist authors to create content. The combination of linguistic parsing, concept extraction, knowledge graph query, and contextual templating were effective techniques to provide such supplementary material. **TODO:** Confirm based on Anna?s results.

Interface testing of content authors showed that? **TODO:** (e.g. Adding a KG helps users to create supplemental content more rapidly, or adding a KG helps avoid content recreation by linking and reusing existing resources.)

Our immediate next step is to productize the server-side portion of our work. We intend to deploy knowledge graph services in several Amazon Web Services availability zones to facilitate rapid network access in geographical regions where our largest customers are located. We will also add the client-side portion into production versions of our JavaScript rich text editors, Textbox.io and TinyMCE.

We intend to increase the number of concept types, and their associated templates.

We intend to grow the size and complexity of the Ephox knowledge graph beyond general encyclopaedic content in order to present supplementary material in progressively more contextual ways. We would like to include data that is specific to particular industries, such as manufacturing or pharmaceutical content, and even allow for extension by customers to allow querying of company-specific data. Finally, we intend to extend our user interface to link the presented concepts to related concepts, so that a user will be able to explore relevant content within the editing environment.

6. ACKNOWLEDGMENTS

The authors would like to thank James Johnson for prototyping IBM Watson integration into Ephox editors, and Ephox Corporation for supporting this work.

7. REFERENCES

[1] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007.

[2] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM, 2008.

[3] R. Cyganiak, D. Wood, and M. Lanthaler. Rdf 1.1 concepts and abstract syntax. *W3C Recommendation*, 25:1–8, 2014.

[4] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[5] S. Harris, A. Seaborne, and E. Prud'hommeaux. Sparql 1.1 query language. *W3C Recommendation*, 21, 2013.

[6] A. Miles and S. Bechhofer. Skos simple knowledge organization system reference. *W3C recommendation*, 18:W3C, 2009.

[7] E. Prud'hommeaux, C. Buil-Aranda, et al. Sparql 1.1 federated query. *W3C Recommendation*, 21, 2013.

[8] A. Singhal. Introducing the knowledge graph: things, not strings. official google blog (may 2012), 2012.