Volume 5, Issue 4, April 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Advance Resource Discovery System using Data Mining Technique

Yogita Deshmukh, Prof Yogesh Bhute
Department of CSE, Abha Gaikwad Patil College of Engineering
Nagpur, Maharashtra, India

Abstract—One of the challenges facing the current web is the efficient use of all the available information. The Web 2.0 phenomenon has favored the creation of contents by average users, and thus the amount of information that can be found for diverse topics has grown exponentially in the last years. Initiatives such as linked data are helping to build the Semantic Web, in which a set of standards are proposed for the exchange of data among heterogeneous systems. However, these standards are sometimes not used, and there are still plenty of websites that require naive techniques to discover their contents and services. This paper proposes an integrated system for content and service discovery and extraction. The system is divided into several layers where the discovery of contents and services is made in a representational stateless transfer system such as the web. It employs several web mining techniques as well as feature oriented modeling for the discovery of cross-cutting features in web resources. The system is used in a scenario of electronic newspapers. An intelligent agent crawls the web for related news, and uses services and visits links automatically according to its goal. This scenario illustrates how the discovery is made at different levels and how the use of semantics helps implement an agent that performs high-level tasks.

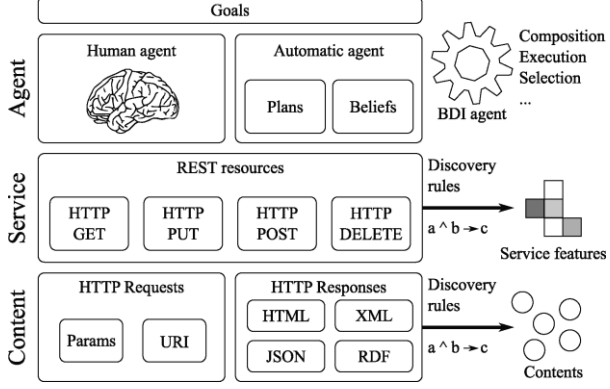
Keywords-Agents, contents, discovery, information extraction,representational stateless transfer system (REST), services.

I. INTRODUCTION

The vast amount of information and services available on the web makes it a platform for the development of mashed up applications with intensive information usage.. However, there are still plenty of websites that do not provide appropriate semantic metadata in the resources they publish. This causes their services and content to be processable only by the human users who visit these websites. The reasons for this can be various: because of limited knowledge by developers of the Semantic Web standards, or because of the limited effort that developers can spend on these tasks. This paper proposes a system for the discovery of services and contents in the web. Our system allows intelligent focussed crawling [3] and agents' accessing the deep web. Therefore, an agent architecture that uses this discovery system for goal-oriented discovery of resources is also described. This agent architecture allows implementing agents that intelligently crawl and use services for retrieving contents in the web that correspond to some top goals, usually, stated by the user. This has let us implement an agent that covers a use case in a related news search scenario. The agent searches the web and combines different information sources to carry out effective reasoning that determines its decisions and behavior during the crawling.

II. DISCOVERY SYSTEM

An integrated system for content and service discovery is defined in this section. We understand discovery as the process of identification and construction of an element's semantically meaningful description at some particular level. The system is shown in Fig. 1. It is stacked on top of the representational stateless transfer (REST) architectural style [4], the architectural style on which the World Wide Web is based. This system uses three levels of abstraction on the content and services that are available on the web. They are listed next from top to bottom.



1. Agent level: This layer comprises the orchestration of services for fulfilling a user goal. Searching blogs to obtain relevant information about a product, look for the best price,
2. Service level: This layer comprises the services that agents are able to use on the web, which nowadays vary from search services to booking services, social networking services, and so on. These services are generally orchestrated at a higher layer. They make use of contents in the lower layer by exchanging requests and responses with representations of resources that are present on the web.
3. Content level: This level comprises the requests and responses that are exchanged between clients and servers when interacting with web resources. In the REST architectural style, requests consist mainly of a verb and a uniform resource identifier (URI), optionally, with parameters, while responses vary in format, although in the web they will usually be in hyper text markup language (HTML).

Discovery will take place at these levels. At the service level, REST resources would be the discoverable elements to be analyzed, with semantic service descriptions being obtained. Discovery of data at the content level would produce meaningful semantically-annotated information. Therefore, a method to extract semantic descriptions out of unstructured data at every level of the system will be described. A uniform approach that employs first-order logic rules is employed to model discovery rules that allow identifying features at all levels. The combination of these features will comprise a semantic description for a discovered element.

III. AGENT MODEL

This section defines an agent model on top of the service level layer in the proposed service discovery system. This agent model makes use of the REST architectural style through semantically annotated services and contents. As shown in the general system in Fig. 1, the agent is designed to perform the same tasks as a regular, human user of the web. This problem statement leads to the agent's ability to browse the web and run services in just the same way a human user would, with discovery rules as the means for extracting semantic descriptions from regular resource representations. The agent would attempt to achieve a top level goal in the same way as a human user, e.g., finding a fact, a place, or a picture for a particular person. As stated in Section II, the discovery rules are the result of a machine learning algorithm applied to supervised data. This supervised data can be added manually as a training dataset that is processed in a later stage. Furthermore, by introspecting into the discovery rules, the agent can anticipate the content and services that can be extracted out of resources, thus modifying its behavior without interacting with those resources.

A. Architecture

The agent follows the belief–desire–intention (BDI) pattern, which differentiates the independent modules that comprise a reactive system interacting with other systems. In our case, beliefs are resource description system (RDF) contents that are extracted from web pages. The agent has plans that represent the possible actions that it can perform, such as executing discovered services or visiting links, while the intentions are stacks of these plans to reach a particular goal. These top-level goals therefore represent discovery targets, i.e., contents which the agent attempts to add to its knowledge base. Thus, both beliefs and goals are sets of RDF triples, while intentions are stacks of plans that are fired upon the creation or deletion of triples in the beliefs and goals triple sets. This makes up a naive adaptation of Agent Speak's agent model [5] to RDF, which results in an agent model, which is similar to approaches that integrate RDF and Semantic Web standards with BDI agents [6]. The resulting architecture is shown in Fig. 2.

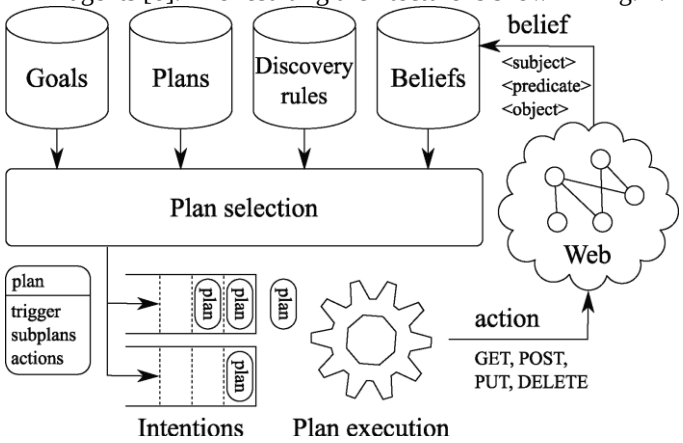


Fig 2 Agent Model

B. Plans

As long as the agent makes use of a RESTful architecture, it is able to interact with resources by exchanging requests and responses. This allows following hyperlinks and executing services such as web forms. Therefore, the four main hyper text transfer protocol (HTTP) methods make up the set of actions employed by the agent. The set of plans can be extended, adapting to different domains, in order to establish domain-specific behavior, especially, in the presence of certain services in the system considered.

1 Focused Crawling Plan: Whenever a resource's representation is expected to have contents with triples that are present in the goal set (trigger), perform a GET on that resource (action). This way, the agent will crawl the web on a greedy basis, looking to fulfill its top-level goals. This can be done thanks to the content discovery rules in the knowledge base, which allow anticipating the contents to be found in a resource's representation. In Agent Speak language, this plan
is represented as

$$+![x, rdf:type, t] : content\ rule(y, x) \wedge [y, sc:type, t] \leftarrow get(x). \quad (1)$$

Note that, according to the scraping ontology [7], a sc:type predicate indicates that in a triple < a,sc:type, b >, the HTML fragment a is of type b after the extraction is performed.

2) Deep Web Crawling Plan: Whenever a keyword-filtered retrieval resource is expected to provide results that meet one or more triple in the goal set (trigger), perform a GET on that resource. This allows using search forms to look for desired contents.

IV. SERVICE LEVEL

The discovery at service level involves building semantic service descriptions out of the HTTP interaction data and the discovered semantic contents from the lower level in the proposed discovery system. The input service model, therefore, comprises the output content model and the HTTP interaction data, i.e., the involved URI, the HTTP verb, and the HTTP parameters.

The output service model used for service discovery applies ideas inspired by mix ins, aspect-oriented, and feature oriented programming paradigms to semantic service description. These paradigms extend the paradigm of object-oriented programming by allowing the modeling of secondary concerns in an isolated way. Feature oriented programming (FOP) [8],[9] is a composition model that allows refining classes through the definition of features, i.e., subclasses with core functionality.

A service is modeled by the set of features *f*1, *f*2,,, *fi* that it has. In web applications, some examples of service features are performing a retrieval operation, requiring user authentication, performing a storage operation, handling images, and outputting a set of resources. The library of features constitutes the output service model. Service discovery rules are used to provide a mapping between the service feature set and the input model: discovery rules map a set of features *f*1, *fk* to a set of conditions, defined using the output content model and the HTTP interaction data. The output service model is therefore a vocabulary of terms that can be extended, with each term representing a feature.

V. CONTENT LEVEL

The web is a hypermedia system that follows the REST architectural style [4]. When a client accesses a web resource on a server, the server returns a representation of the resource. Usually, these representations are formatted in HTML, a language that allows defining the structure of a document for rendering on a web browser. An HTML document is structured as a document object model (DOM) tree that defines the logical structure of the HTML document that will be used for rendering the representation on a web browser. In order to have information about the resource's content, and not about its rendering structure, linked data proposes using resource representations that include metadata, by enhancing HTML with semantic annotations or by providing RDF representations.

Whenever a resource provides unannotated HTML, a technique that in some way processes the DOM tree needs to be used to identify the structure of the data present in the HTML document and build the associated RDF graph. Also, in a web resource there are DOM fragments that do not provide information, such as advertisements, headers, footers, and decorative elements, while other fragments, such as posts or comments, have valuable information. In our system, discovery rules will be employed to identify which pieces of information are relevant in a web resource and to identify which relations are stated in a web fragment. For instance, a heading in a piece of news might represent the news title. A discovery rule will use content style sheets (CSS) information, rendering information, or natural language processing (NLP) to identify the relevant data in the resource's representation. A rule that generalizes is one which is valid for all web resources that contain the same kind of data. If a rule is only valid for the web resource (or resources) that it was defined for, it does not generalize to different resources. The main limitation of wrapper induction is that wrappers are only valid for the web pages for which they were designed [19]. Using NLP and visual selectors as input content model improves the generalization capabilities of the discovery rules [20].

V. SCENARIO

A scenario that makes use of the aforementioned agent model and discovery system has been defined. A barebones implementation of the agent has been developed, based on the Open Source tool Scrappy,2 a screen scraper that was extended with the previously defined intelligent agent model. The result is an agent which is able to address top-level goals for discovering contents and services on the web and will be here used in a scenario to validate the proposed system. The agent is then configured to perform high-level tasks under a scenario involving electronic newspapers.

A. Description

This scenario will address the task of comparing similar pieces of news when surfing the web. Electronic newspaper readers often read the same piece of news in several sources, to cross-check the views of the different

newspapers, therefore, spending a considerable amount of time searching the web for news. Also, users often browse news by similar topics, as they represent their own interests..

The main challenges that are addressed in this scenario are as follows.

1) Extracting the data from the addressed contents. By following the proposed discovery system, discovery rules are used to perform this extraction. Defining the discovery rules for the information extraction is done semi automatically, thanks to rule induction algorithms [21].

2) Using services for relating the news posts and identifying the recommendations. Using the discovery system, the agent can use services by using feature oriented descriptions of these services. The services will be used by the agent according to the plans that are available in the agent's plan set.

As a result, two stages take place during the agent's lifecycle.

1) Feeding phase: The agent is provided with a set of HTML pages from the addressed newspapers that are annotated with the RDF data that their HTML represent. The agent then inducts discovery rules for extracting the semantic representation of the newspapers' resources. Open- Calais is used to enrich the semantic descriptions of news posts in order to identify recommendations. Open- Calais is a service that returns *linked data informationabout a piece of text, returning disambiguated entities about places or people mentioned in the text.*

2) Execution phase: A plugin that is installed into the web browser3 allows launching the agent with the goal of returning services that are related to the current news post being browsed. After clicking the button, the agent performs its focussed crawling and discovers a set of results, which are then returned to the web browser so that the user can review the related news that the agent found in the newspapers.

B. Results

To evaluate the agent, we provided users with the different
news posts and the recommendations by the agent in the browser plugin . The users were then asked to answer different questions about their experience when browsing the web using the agent..

VI. RELATED WORK

There has already been plenty of research work on service and content discovery on the web, with several approaches this paper. At the content level, scraping techniques have been used to extract data, in a similar way, as other information extraction approaches. We can point to the systems Piggy Bank [23], Reform [24], Thresher [25] and Marmite [26], Chickenfoot [27], or Denodo [28]. These approaches propose techniques that facilitate the scraping process, and which can complement our discovery rules. Also, gleaning resource descriptions from dialects of languages (GRDDL) [29] is the standard technology for extracting RDF from HTML documents. Unlike these approaches, ours provides a semantic system that enables dereferencing the scraped data and reasoning about the scraping process. This enables enhancements in the scraping process, such as reasoning about visual aspects of the web resource, distributed scraping using multi agent systems, or focused scraping and reasoning about the scraped resources.

VII. CONCLUSION AND FUTURE WORK

This paper proposed a system for the discovery of services and content in the web, describing an algorithm for the induction of rules for discovery, as well as its application to data and resource levels in the REST architectural style of the web. An agent architecture that fits the discovery system was also defined for implementing combined services or contents in cases where discovery is required. A scenario that made use of an agent to discover related news items was described. Future work will involve building a broader library of reusable feature descriptions and content types to enhance the output models at each discovery level. Here, a reduced subset of building blocks has been shown to solve a complex use case that involved typical elements in the web, such as news posts. Further training sets for the semantic definitions would improve the generalizability of the discovery rules to other scenarios, which could differ greatly from the ones considered in this paper, thus increasing the agent's versatility.

REFERENCES

[1] T. Berners-Lee, J. Hendler, and O. Lassila, "The semantic web," *Sci. Am.,* vol. 284, no. 5, pp. 28–37, 2001.

[2] C. Bizer, T. Health, K. Idehen, and T. Berners-Lee, "Linked data on the web," in *Proc. 17th Int. Conf. WWW*, 2008, pp. 1265–1266.

[3] S. Chakrabarti and B. Dom, "Focused crawling: A new approach to topic-specific web resource discovery," *Comput. Netw.*, vol. 31, pp. 1623–1640, Feb. 1999.

[4] R. T. Fielding, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, Univ. California, Berkeley,CA, USA, 2000.

[5] A. S. Rao, "Agentspeak (l): BDI agents speak out in a logical computable language," in *Proc. 7th Eur. Workshop Modelling Autonomous Agents MultiAgent World: Agents Breaking Away*, LNCS 1038. Eindhoven, The Netherlands: Springer, 1996, pp. 42–55.

[6] M. Laclavık, Z. Balogh, M. Babık, and L. Hluch`y, "Agentowl: Semantic knowledge model and agent architecture," *Comput. Inf.*, vol. 25, no. 5, pp. 419–437, 2006.

[7] J. I. Fern´andez-Villamor, C. A. Iglesias, and M. Garijo, "A semantic scraping model for web resources–Applying linked data to web page screen scraping," in *Proc. Third Int. Conf. Agents Artif. Intell.*, 2011, pp. 451–456.

[8] C. Prehofer, "Feature-oriented programming: A fresh look at objects," in *Lecture Notes in Computer Science*, vol. 1241. Berlin, Germany: Springer, 1997, pp. 419–443.

[9] S. Apel, T. Leich, M. Rosenmüller, and G. Saake, "Combining feature orientedand aspect-oriented programming to support software evolution," in *Proc. AMSE05 ECOOP05*, 2005, pp. 3–16.

[10] G. Bracha and W. Cook, "Mixin-based inheritance," in *Proc. Eur. Conf. Object-Oriented Programming Object-Oriented Programming Syst. LanguagesAppl.*, 1990, pp. 303–311.

[11] S. Apel, T. Leich, and G. Saake, "Aspectual mixin layers: Aspects and features in concert," in *Proc. 28th Int. Conf. Softw. Eng.*, 2006, p. 131.

[12] R. Lopez-Herrejon, "Understanding feature modularity in feature oriented programming and its implications to aspect oriented programming,"in *Proc. ECOOP PhDOOS Workshop Doctoral Symp.*, 2005.

[13] S. Trujillo, D. Batory, and O. Diaz, "Feature oriented model driven development: A case study for portlets," in *Proc. 29th Int. Conf. Softw. Eng.*, 2007, pp. 44–53.

[14] F. Steimann, "On the representation of roles in object-oriented and conceptual modelling," *Data Knowledge Eng.*, vol. 35, no. 1, pp. 83–106, 2000.

[15] W. Harrison and H. Ossher, "Subject-oriented programming: A critique of pure objects," *ACM Sigplan Notices*, vol. 28, no. 10, pp. 411–428, 1993.

[16] J. I. Fern´andez-Villamor, C. Iglesias, and M. Garijo, "Microservices:Lightweight service

[17] J. G. Breslin, A. Harth, U. Bojars, and S. Decker, "Toward semantically interlinked online," in *Proc. 2nd Eur. Semantic Web Conf.*, LNCS 3532.2004, pp. 500–514.

[18] K. Lerman, S. N. Minton, and C. A. Knoblock, "Wrapper maintenance:A machine learning approach," *J. Artif. Intell. Res.*, vol. 18, pp. 149–181,Jun. 2003.

[19] N. Kushmerick, "Wrapper induction for information extraction," University of Washington, 1997.

[20] D. Cai, S. Yu, J.-R. Wen, and W.-Y. Ma, "Extracting content structure for web pages based on visual representation," in *Proc. 5th Asia Pacific Web Conf.*, 2003, pp. 406–417.

[21] J. I. Fern´andez-Villamor, C. A. Iglesias, and M. Garijo, "First-order logic rule induction for information extraction in web resources," *Int. J. Artif. Intell. Tools*, vol. 21, no. 6, pp. 1250032-1–1250032-2, Dec. 2012.

[22] M.-G. Butuc, "Semantically enriching content using opencalais," in *Proc. Romanian Workshop Distributed Systems (EDITIA)*, Suceava, Romania, 2009, pp. 77–88.

[23] D. Huynh, S. Mazzocchi, and D. Karger, "Piggy bank: Experience the Semantic Web inside your web browser," *web Semantics*, vol. 5, no. 1, pp. 16–27, 2007.

[24] M. Toomim, S. M. Drucker, M. Dontcheva, A. Rahimi, B. Thomson, and J. A. Landay, "Attaching UI enhancements to websites with end users," in *Proc. Conf. Human Factors Comput. Syst.*, pp. 1859–1868, 2009.

[25] A. Hogue, "Thresher: Automating the unwrapping of semantic content from the World Wide Web," in *Proc. 14th Int. World Wide Web Conf.*, 2005, pp. 86–95.

[26] J. Wong and J. I. Hong, "Making mashups with marmite: Toward enduser programming for the web," in *Proc. Conf. Human Factors Comput. Syst.*, 2007, p. 1435.

[27] M. Bolin, M. Webber, P. Rha, T. Wilson, and R. C. Miller, "Automation and customization of rendered web pages," in *Proc. Symp. User Interface Softw. Technol.*, 2005, p. 163.

[28] A. Pan, J. Raposo, M. A´ lvarez, P. Montoto, V. Orjales, J. Hidalgo, L. Ardao, A. Molano, and A. Vi˜na, "The Denodo data integration platform," in *Proc. 28th Int. Conf. Very Large Data Bases (VLDB)*, Aug. 2002, pp. 986–989.

[29] D. Connolly. (2007). *Gleaning resource descriptions from dialects of languages* [Online]. Available: http://www.w3.org/TR/grddl/

[30] M. J. Hadley. (2006). *web application description language* [Online]. Available: https://wadl.dev.java.net/wadl20061109.pdf

[31] A. P. Sheth, K. Gomadam, and J. Lathem, "SA-REST: Semantically interoperable and easier-to-use services and mashups," *IEEE Internet Comput.*, vol. 11, no. 6, pp. 91–94, Nov.–Dec. 2007.

[32] Wright State University. (2008). *HTML Microformat for Describing RESTful web Services and APIs* [Online]. Avialble: http://knoesis.wright. edu/research/srl/projects/hRESTs/#hRESTs

