Survey of Semantic Annotation Platforms

Lawrence Reeve
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104 USA
larry.reeve@drexel.edu

Hyoil Han
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104 USA
hhan@cis.drexel.edu

ABSTRACT
The realization of the Semantic Web requires the widespread availability of semantic annotations for existing and new documents on the Web. Semantic annotations are to tag ontology class instance data and map it into ontology classes. The fully automatic creation of semantic annotations is an unsolved problem. Instead, current systems focus on the semi-automatic creation of annotations. The Semantic Web also requires facilities for the storage of annotations and ontologies, user interfaces, access APIs, and other features to fully support annotation usage. This paper examines current Semantic Web annotation platforms that provide annotation and related services, and reviews their architecture, approaches and performance.

Categories and Subject Descriptors
A.1 [**General Literature**]: Introduction and Survey

General Terms
Performance, Design.

Keywords
Semantic Web, Semantic Annotation, Information Extraction.

1. INTRODUCTION
Full implementation of the Semantic Web requires widespread availability of semantic annotations for existing and new documents on the Web. Manual annotation is more easily accomplished today, using authoring tools such as Semantic Word [20], which provide an integrated environment for simultaneously authoring and annotating text. However, the use of human annotators is often fraught with errors due to factors such as annotator familiarity with the domain, amount of training, personal motivation and complex schemas [1]. Manual annotation is also an expensive process, and often does not consider that multiple perspectives of a data source, requiring multiple ontologies, can be beneficial to support the needs of different users. For example, vision-impaired users can use annotations to provide faster navigation through a web site [23], while sighted users can use annotations of the same document to provide a detailed view of a domain. Another problem with manual annotation is the volume of existing documents on the Web that must be annotated to become a useful part of the Semantic Web. Manual semantic annotation has lead to a knowledge acquisition bottleneck [15].

SAC'05, March 13-17, 2005, Santa Fe, New Mexico, USA.

To overcome the annotation acquisition bottleneck, semi-automatic annotation of documents has been proposed. Semi-automatic means, as opposed to completely automatic, are required because it is not yet possible to automatically identify and classify all entities in source documents with complete accuracy. All existing semantic annotation systems rely on human intervention at some point in the annotation process, using the paradigm of balanced cooperative modeling [15]. Automated annotation provides the scalability needed to annotate existing documents on the Web, and reduces the burden of annotating new documents. Other potential benefits are consistently applying ontologies, and using multiple ontologies to annotate a single document.

This paper presents a survey of current semantic annotation platforms that can be used to perform semi-automatic annotation. The platforms vary in their architecture, information extraction tools and methods, initial ontology, amount of manual work required to perform annotation, performance and other features, such as storage management.

The rest of paper is organized as follows. Section 2 presents the classification of semantic annotation platforms. Sections 3 and 4 describe platform overview and performance evaluation, respectively. Section 5 summarizes and compares semantic annotation platforms (SAPs). Section 6 concludes the paper.

2. PLATFORM CLASSIFICATION
Semantic annotation platforms (SAPs) can be classified based on the type of annotation method used. There are two primary categories, Pattern-based and Machine Learning-based, as shown in Figure 1. In addition, platforms can use methods from both types of categories, called Multistrategy, in order to take advantage of the strengths, and compensate for the weaknesses, of the methods in each category.

Pattern-based SAPs can perform pattern discovery or have patterns manually defined. Most pattern-discovery methods follow the basic method outlined by Brin [3]. An initial set of entities is defined and the corpus is scanned to find the patterns in which the entities exist. New entities are discovered, along with new patterns. This process continues recursively until no more entities are discovered, or the user stops the process. Annotations can also be generated by using manual rules to find entities in text.

Machine learning-based SAPs utilize two methods: probability and induction. Probabilistic SAPs use statistical models to predict the locations of entities within text. For example, the Hidden Markov Model is used within the DATAMOLD algorithm [2] to find instance data within HTML pages. Also, the LP^2 algorithm [21] is the core information extraction (IE) algorithm in the Amilcare toolkit [12], which is used by both the Armadillo [10] and Ont-O-Mat [12] SAPs to perform wrapper induction.

Multistrategy SAPs are able to combine methods from both pattern-based and machine learning-based systems. No SAP currently implements the multistrategy approach for semantic annotation, although it has been implemented in systems for ontology extraction, such as On-To-Knowledge [13]. SAPs can provide both pattern-based and machine learning-based methods when they are designed with extensible architectures.

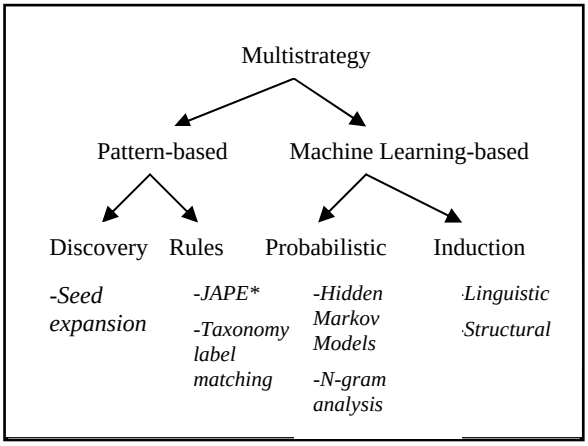

Figure 1: Classification of Semantic Annotation Platforms

Note: JAPE means Java Annotation Pattern Engine.

3. PLATFORM OVERVIEW

Semantic annotation platforms provide support for IE implementations, ontology and knowledgebase management, access APIs, storage (e.g., RDF [22] repositories), and user-interfaces for ontology and knowledgebase editors [18]. Platforms may include only a subset of these features, and may include other features not generally included by all SAPs, such as annotation storage.

This section briefly describes several currently available SAPs and shows their empirically-measured performance. The SAPs were chosen from a literature review of recent semantic annotation journal articles. The idea is to survey, within a brief format, the supporting features, annotation methods used, and effectiveness of a set of SAPs that represent each area of the classification shown in Figure 1. The survey shows the convergence of common features (such as ontology and knowledgebase storage/retrieval) and the integration of newer ideas (such as support for multiple ontologies annotating a single source document).

3.1 AeroDAML

AeroDAML [14] uses a pattern-based approach and is designed to map proper nouns and common relationships to corresponding classes and properties in DARPA Agent Markup Language (DAML) [11] ontologies. AeroDAML uses AeroText for its information extraction (IE) component. The AeroText Java API is used to access IE parts and map them into RDF triples using an ontology as a guide. The default ontology consists of two parts. The upper level uses the WordNet [19] noun synset hierarchy. The lower level uses the knowledgebase provided by AeroText [14]. The integrated AeroText system consists of four main components: 1) KnowledgeBase (KB) compiler for transforming linguistic data into a run-time knowledgebase; 2) KnowledgeBase Engine for applying the KB to source documents; 3) an IDE for building and testing KBs, and 4) Common KnowledgeBase containing domain independent rules for extracting proper nouns and relations [14]. AeroDAML operates in two modes depending on the version used. The web-based system allows users to enter a URI and have the DAML-based annotation of the page returned using the default ontology. The client-server version allows users to enter a filename and then returns the DAML-based annotation of the text using a custom ontology.

3.2 Armadillo

Armadillo [10] uses the Amilcare IE system to perform wrapper induction on web pages to mine web sites that have a highly-regular structure. Armadillo uses a pattern-based approach to find entities. It finds its own initial set of seed-patterns rather than requiring an initial set of seeds [3]. Manual patterns are used for the named entity recognizer. No manual annotation of corpus documents is required. Once the seeds are found, pattern expansion is then used to discover additional entities. Information redundancy, via queries to Web services such as Google and CiteSeer, is used to verify discovered entities by analyzing query results to confirm or deny the existence of an entity, similar to the way the PANKOW algorithm [5] operates.

The use-case implemented in Armadillo is extracting worker details from a university computer science department web site in order to find personal data, such as name, position, home page, email address, and other contact information. The seed-discovery and expansion finds worker names in the web pages. Since many names may be discovered, the Web services are queried to confirm a person actually works in the department. The names are then used to discover home pages, where detailed information about a person can often be found and extracted. Armadillo is also interesting in that it attempts to discover bibliographic citations for each person discovered. The information redundancy approach was also applied to bibliographic entries, but with a lower success rate than discovering and extracting information about people from home pages [10].

3.3 KIM

The Knowledge and Information Management (KIM) platform [18] contains an ontology, a knowledgebase, a semantic annotation, an indexing and retrieval server, as well as front-ends for interfacing with the server. For ontology and knowledgebase storage it uses the SESAME RDF repository [4], and for search it uses a modified version of the Lucene [8] keyword-based search engine. The semantic annotation process relies on a pre-built lightweight ontology called KIMO as well as an inter-domain knowledgebase. KIMO defines a base set of entity classes, relationships, and attribute restrictions. The knowledgebase is populated with 80,000 entities consisting of locations and organizations, gathered from a general news corpus. Named-

entities found during the annotation process are matched to their type in the ontology and also to a reference in the knowledgebase. The dual mapping allows the information extraction process to be improved by providing disambiguation clues based on attributes and relations [18].

The information extraction component of semantic annotation is performed using components of the GATE toolkit [6]. GATE provides IE implementations of tokenizers, part-of-speech taggers, gazetters, pattern-matching grammars (JAPE), and co-reference resolution [18]. Some components of GATE have been modified to support the KIM server. For example, pattern-matching grammar rules are based on ontology classes rather than simple types [18]. Other components of semantic annotation have been custom developed. The gazetteer, for example, performs entity alias lookups using the knowledgebase rather than from an external source.

3.4 MnM

MnM [21] provides an environment to manually annotate a training corpus, and then feed the corpus into a wrapper induction system based on the Lazy-NLP (natural language processing) algorithm. The resulting output is a library of induced rules that can be used to extract information from corpus texts [21]. Lazy-NLP systems are based on linguistic information, and rules are generated using sets of conjunctive conditions on adjacent words [21]. The rule induction process generates two types of rules: tagging and correction. A rule is composed of a pattern of conditions on a connected sequence of words, followed by an action performed when the pattern is found to match a part of text [21]. For tagging rules, the action performed is the insertion of a semantic tag into the text. For correction rules, the action performed is the insertion of information that causes semantic tags to shift location, based on training information. The corrective tags are inserted during the training period, when the training corpus is re-annotated using the induced rules. If an induced tagging rule is found to be incorrect in its location, it is corrected using a correction rule rather than replaced.

3.5 MUSE

MUSE [16] was designed to perform named entity recognition and coreferencing. It is implemented using the GATE framework [6]. The IE components, called *processing resources* (PRs), form a processing pipeline used to discover named entities. MUSE executes PRs conditionally based on text attributes. Conditional processing is handled using a Switching Controller, which calls the appropriate PR in the specified order. The use of conditional processing allows MUSE to obtain accuracies similar to machine learning systems [16]. Semantic tagging is accomplished using the Java Annotations Pattern Engine (JAPE) [7]. Rules using the JAPE grammar are constructed to generate annotations. The Semantic Tagger can use tags generated by processing resources run earlier in the pipeline. For example, if the gazetteer recognizes a first name and the part-of-speech tagger recognizes a proper noun, a JAPE rule can use both tags to annotate an entity of type Person [16]. The MUSE system is more sophisticated than a gazetteer because a gazetteer cannot provide an exhaustive list of all potential named-entities, and cannot resolve entity ambiguities (e.g., Washington can be a city or a person) [16].

3.6 Ont-O-Mat

Ont-O-Mat [12] is an implementation of the S-CREAM (Semi-automatic CREAtion of Metadata) semantic annotation framework. The IE component is based on Amilcare. Amilcare is machine learning-based and requires a training corpus of manually annotated documents. Amilcare uses the ANNIE ("A Nearly-New IE system") part of the GATE toolkit to perform IE [12]. The result of ANNIE processing is passed to Amilcare, which then induces rules for IE using a variant of the LP^2 algorithm. The wrapper induction process uses linguistic information, and is the same Amilcare wrapper induction process as MnM [21], generating tagging and correction rules.

Ont-O-Mat [12] provides an extensible architecture to replace selected components. Later semantic annotation research replaced the original annotation component of Ont-O-Mat with an implementation of the PANKOW (Pattern-based Annotation through Knowledge On the Web) algorithm [5]. The PANKOW process takes proper nouns from the IE phase and generates hypothesis phrases based on linguistic patterns and the specified ontology. For example, a sports ontology may generate hypothesis phrases from the proper noun “Pete Rose” using patterns such as “Pete Rose *is a* Player” and “Pete Rose *is a* Team,” where “Player” and “Team” are ontology concepts. The hypothesis phrases are then presented to the Google web service. The phrase with the highest query result count is then used to annotate the text with the appropriate concept. The core principle is called “disambiguation by maximal evidence” [5], and is similar to the approach used by Armadillo [10], which uses multiple web services to find evidence.

3.7 SemTag

SemTag [9] is the semantic annotation component of a comprehensive platform, called Seeker, for performing large-scale annotation of web pages. SemTag performs annotation in three passes: Spotting, Learning, and Tagging. The Spotting pass examines tokenized words from source documents and finds label matches from the taxonomy. If a label match is found, a window of ten words to either side of the source document match is kept. In the Learning pass, a sample of the corpus is examined to find the corpus-wide distribution of terms at each node of the taxonomy. The Tagging pass is then executed, scanning all of the windows from the Spotting pass and disambiguating the matches. Once a match is confirmed, the URL, text reference, and other metadata are stored. SemTag/Seeker is an extensible system, so new annotation implementations can replace the existing Taxonomy-based Disambiguation algorithm (TBD). The taxonomy used by SemTag is TAP. TAP is shallow and covers a range of lexical and taxonomic information about popular items such as music, movies, authors, sports, health and so forth [9]. The annotations generated by SemTag are stored separate from the source document. The intent of the SemTag/Seeker design is to provide a public repository with an API that will allow agents to retrieve the web page from its source and then request the annotations separately from a Semantic Label Bureau [9].

4. EVALUATION

In this section, the performance of SAPs described in section 3 is reported. Table 1 shows the author-reported performance of

various platforms, with the exception of AeroDAML, Ont-O-Mat using Amilcare and SemTag, whose authors did not provide complete performance information. The standard measures of Precision, Recall, and F-measure, taken from the information retrieval field, were used by the remaining SAP authors in determining annotation effectiveness. In the general definition of recall and precision shown below, "accurate" and "inaccurate" refer to annotations generated semi-automatically by a SAP, while "all" refers to all annotations generated by a human annotator.

$$Annotation\ Recall = \frac{accurate}{all}$$

$$Annotation\ Precision = \frac{accurate}{accurate + inaccurate}$$

F-measure is the harmonic mean of precision and recall. The highest performing machine learning-based platform is MnM. For pattern-based platforms, MUSE is best. The worst performing of all implementations is Ont-O-Mat using PANKOW. PANKOW is a recent effort to use unsupervised learning in a pattern-based system, and performance improvements are expected as the system develops further [5].

Table 1: Measures of platform effectiveness.

Framework	Precision	Recall	F-Measure
Armadillo	91.0	74.0	87.0
KIM	86.0	82.0	84.0
MnM	95.0	90.0	n/a
MUSE	93.5	92.3	92.9
Ont-O-Mat: PANKOW	65.0	28.2	24.9
SemTag	82.0	n/a	n/a

5. PLATFORM SUMMARY

The semantic annotation platforms shown in Table 2 and discussed briefly in Section 3 are distinguished by various attributes that have an impact on their automated semantic annotation effectiveness. For example, the method used to find entities is the major determinant in performance. The method column shows the classification that is defined in section 2. The most common SAP techniques are manually-created rules [16], pattern matching [18], automatic discovery of patterns [10], and wrapper induction, either linguistic [12] or structural based [17]. While the machine learning methods, such as those used by Amilcare [12], usually perform better [21], the rule-based MUSE system using conditional processing has shown that rule-based systems can equal the performance of machine learning-based systems [16].

All SAPs require some type of lexicon and resource. Rule-based systems require rules, pattern discovery systems require an initial set of seeds, machine learning systems require a training corpus (usually annotated), while others require the construction of dictionaries for named-entity recognition. Ontologies must also be supplied to SAPs because the semantic annotations are to tag ontology class instance data and map it into ontology classes. Some ontologies are simple taxonomies and structures such as address books, while others are complex ontology implementations with relationships defined. These ontologies bootstrap the process of annotation by providing enough information to begin annotating. Some systems contain a feedback cycle, where the ontology and a supporting knowledgebase learn more information each time the annotation process is run against a document set. This feedback cycle results in more accurate annotations over time [18]. Pattern-based systems often require the manual generation of rules, as shown in Table 2. The notable difference is the recent work done with PANKOW [5] to automatically discover an initial set of seed patterns. This approach can be contrasted with the MUSE system [16], where rules must be completely defined before the annotation process is started.

SAP architectures can be categorized as extensible or not. Non-extensible architectures usually focus on a single domain, method, or toolkit. For example, AeroDAML relies on Aerotext [14], and SemTag [9] relies on its own taxonomy label matching. Extensible SAP architectures allow various system components to be replaced or extended with other components. Examples are MUSE [16] and Ont-O-Mat [12]. Extensible SAPs allow newer annotation methods to be tested and integrated while reusing all other platform features.

Most SAPs rely on an external information extraction (IE) system, most of which have been previously developed from the natural language processing community. For example, GATE [6] has been developed and refined for over 8 years, and Amilcare [12] also has several years of development. IE components typically perform language tasks such as tokenization, part-of-speech tagging, sentence splitting, and dictionary lookup. Some IE systems provide additional services such as named entity recognition, IE rule induction using machine learning [12], and finding identity relations between entities in text (co-referencing) [6].

Table 2: Semantic annotation platform summary

Platform	Method	Machine Learning	Manual Rules	Bootstrap Ontology
AeroDAML [14]	Rule	N	Y	WordNet
Armadillo [10]	Pattern Discovery	N	Y	User
KIM [18]	Rule	N	Y	KIMO
MnM [21]	Wrapper Induction	Y	N	KMi
MUSE [16]	Rule	N	Y	User
Ont-O-Mat: Amilcare [12]	Wrapper Induction	Y	N	User
Ont-O-Mat: PANKOW [5]	Pattern Discovery	N	N	User
SemTag [9]	Rule	N	N	TAP

6. CONCLUSION

In this paper a short survey of semantic annotation platforms was presented. In addition, a classification of semantic annotation platform types was developed. Semantic annotation platforms (SAPs) can be distinguished primarily by their annotation method, as that component has the largest impact on the effectiveness of semantic annotation. The two primary approaches are pattern-based and machine learning-based. Machine learning algorithms often perform more effectively than pattern-based methods, but the MUSE system shows that a rule-based system using conditional processing can perform as well as a machine learning system [16]..

SAPs designed with extensible architectures can adapt to evolving technology. Information extraction components can be replaced as different approaches are developed. The continuing evolution of SAPs to provide better annotation and new features while extending existing ones is vital to the realization of the Semantic Web.

7. ACKNOWLEDGEMENT

The authors would like to thank the anonymous reviewers for their helpful feedback.

8. REFERENCES

[1] Bayerl, P.S., Lungen, H., Gut, U. and Paul, K.I., Methodology for reliable schema development and evaluation of manual annotations in *Workshop on Knowledge Markup and Semantic Annotation at the Second International Conference on Knowledge Capture (KCAP)*, (2003).

[2] Borkar, V., Deshmukhy, K. and Sarawagiz, S., Automatic Segmentation of Text into Structured Records in *ACM SIGMOD International Conference on Management of Data*, (2001), 175-186.

[3] Brin, S., Extracting Patterns and Relations from the World Wide Web in *WebDB Workshop at 6th International Conference on Extending Database Technology*, (1998).

[4] Broekstra, J., Kampman, A. and Harmelen, F.v., Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema in *International Semantic Web Conference 2002*, (Sardinia, Italy, 2002).

[5] Cimiano, P., Handschuh, S. and Staab, S., Towards the Self-Annotating Web in *Thirteenth International Conference on World Wide Web*, (2004), 462-471.

[6] Cunningham, H., Maynard, D., Bontcheva, K. and Tablan, V., GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications in *40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, (2002).

[7] Cunningham, H., Maynard, D. and Tablan, V. JAPE: A Java Annotation Patterns Engine, Department of Computer Science, University of Sheffield, 2000.

[8] Cutting, D., Apache Jakarta Lucene http://jakarta.apache.org/lucene/docs/index.html. Last accessed August 31, 2004.

[9] Dill, S., Gibson, N., Gruhl, D., Guha, R., Jhingran, A., Kanungo, T., Rajagopalan, S., Tomkins, A., Tomlin, J.A. and Zien, J.Y., SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation in *Twelfth International World Wide Web Conference*, (Budapest, Hungary, 2003), 178-186.

[10] Dingli, A., Ciravegna, F. and Wilks, Y., Automatic Semantic Annotation using Unsupervised Information Extraction and Integration in ***K-CAP 2003 Workshop on Knowledge Markup and Semantic Annotation***, (2003).

[11] Greaves, M., DAML - DARPA Agent Markup Language http://www.daml.org/. Last accessed August 31, 2004.

[12] Handschuh, S., Staab, S. and Ciravogna, F., S-CREAM -- Semi-automatic CREAtion of Metadata in *SAAKM 2002 -Semantic Authoring, Annotation & Knowledge Markup - Preliminary Workshop Programme*, (2002).

[13] Kietz, J.-U. and Volz, R., Extracting a Domain-Specific Ontology from a Corporate Intranet in *Fourth Conference on Computational Natural Language Learning and of the Second Learning Language in Logic Workshop*, (Lisbon, Portugal, 2000), 167-175.

[14] Kogut, P. and Holmes, W., AeroDAML: Applying Information Extraction to Generate DAML Annotations from Web Pages in *First International Conference on Knowledge Capture*, (2001).

[15] Maedche, A. and Staab, S. Ontology Learning for the Semantic Web. *IEEE Intelligent Systems*, *16* (2). 72-79.

[16] Maynard, D. Multi-Source and Multilingual Information Extraction. *Expert Update*.

[17] Mukherjee, S., Yang, G. and Ramakrishnan, I.V., Automatic Annotation of Content-Rich HTML Documents: Structural and Semantic Analysis in *Second International Semantic Web Conference*, (2003).

[18] Popov, B., Kiryakov, A., Kirilov, A., Manov, D., Ognyanoff, D. and Goranov, M., KIM - Semantic Annotation Platform in *2nd International Semantic Web Conference (ISWC2003)*, (Florida, USA, 2003), 834-849.

[19] Fellbaum, C. *WORDNET: An Electronic Lexical Database*. The MIT Press, 1998.

[20] Tallis, M., Semantic Word Processing for Content Authors in *Second International Conference on Knowledge Capture*, (Sanibel, Florida, 2003).

[21] Vargas-Vera, M., Motta, E., Domingue, J., Lanzoni, M., Stutt, A. and Ciravegna, F., MnM: Ontology Driven Semi-Automatic and Automatic Support for Semantic Markup in *The 13th International Conference on Knowledge Engineering and Management (EKAW 2002)*, (Spain, 2002), 379-391.

[22] W3C, Resource Description Framework (RDF) http://www.w3.org/RDF/. Last accessed August 31, 2004.

[23] Yesilada, Y., Harper, S., Goble, C. and Stevens, R., Ontology Based Semantic Annotation for Enhancing Mobility Support for Visually Impaired Web Users in *K-CAP 2003 Workshop on Knowledge Markup and Semantic Annotation*, (2003).