DIADEM: Domains to Databases*

Tim Furche, Georg Gottlob, and Christian Schallhart

Department of Computer Science, Oxford University,
Wolfson Building, Parks Road, Oxford OX1 3QD
firstname.lastname@cs.ox.ac.uk

Abstract. What if you could turn all websites of an entire domain into a single database? Imagine all real estate offers, all airline flights, or all your local restaurants' menus automatically collected from hundreds or thousands of agencies, travel agencies, or restaurants, presented as a single homogeneous dataset.
Historically, this has required tremendous effort by the data providers and whoever is collecting the data: Vertical search engines aggregate offers through specific interfaces which provide suitably structured data. The semantic web vision replaces the specific interfaces with a single one, but still requires providers to publish structured data.
Attempts to turn human-oriented HTML interfaces back into their underlying databases have largely failed due to the variability of web sources. In this paper, we demonstrate that this is about to change: The availability of comprehensive entity recognition together with advances in ontology reasoning have made possible a new generation of knowledge-driven, domain-specific data extraction approaches. To that end, we introduce DIADEM, the first automated data extraction system that can turn nearly any website of a domain into structured data, working fully automatically, and present some preliminary evaluation results.

1 Introduction

Most websites with offers on books, real estate, flights, or any number of other products are generated from some database. However, meant for human consumption, they make the data accessible only through, increasingly sophisticated, search and browse interfaces. Unfortunately, this poses a significant challenge in automatically processing these offers, e.g., for price comparison, market analysis, or improved search interfaces. To obtain the data driving such applications, we have to explore human-oriented HTML interfaces and extract the data made accessible through them, without requiring any human involvment.

Automated data extraction has long been a dream of the web community, whether to improve search engines, to "model every object on the planet"[1], or

* The research leading to these results has received funding from the European Research Council under the European Community's Seventh Framework Programme (FP7/2007–2013) / ERC grant agreement DIADEM, no. 246858.

[1] Bing's new aim, http://tinyurl.com/77jjqz6.

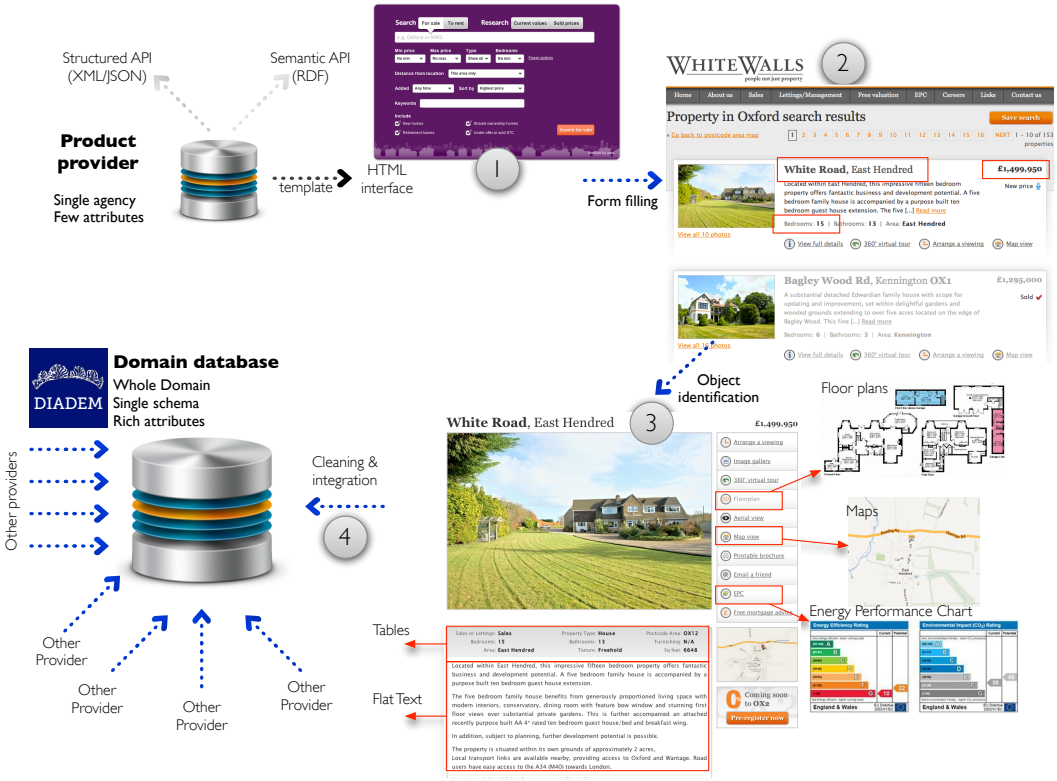


Fig. 1: Data extraction with DIADEM

to bootstrap the semantic web vision. Web extraction comes roughly in two shapes, namely web *information extraction* (IE), extracting facts from flat text at very large scale, and web *data extraction* (DE), extracting complex objects based on text, but also layout, page and template structure, etc. Data extraction often uses some techniques from information extraction such as entity and relationship recognition, but not vice versa. Historically, IE systems are domain-independent and web-scale [15, 12], but at a rather low recall. DE systems fall into two categories: domain-independent, low accuracy systems [3, 14, 13] based on discovering the repeated structure of HTML templates common to a set of pages, and highly accurate, but site-specific systems [16, 4] based on machine learning.

In this paper, we argue that a **new trade-off** is necessary to make *highly accurate, fully automated web extraction possible at a large scale*. We trade off scope for accuracy and automation: By limiting ourselves to a specific domain where we can provide substantial knowledge about that domain and the representation of its objects on web sites, automated data extraction becomes possible at high accuracy. Though not fully web-scale, one domain often covers thousands or even tens of thousands of web sites: To achieve a coverage above 80% for typical attributes in common domains, it does not suffice to extract only from large,

popular web sites. Rather, we need to include objects from thousands of small, long-tail sources, as shown in [5] for a number of domains and attributes.

Figure 1 illustrates the principle of *fully automated data extraction at domain-scale*. The input is a website, typically generated by populating HTML templates from a provider's database. Unfortunately, this human-focused HTML interface is usually the only way to access this data. For instance, of the nearly 50 real estate agencies that operate in the Oxford area, not a single one provides their data in structured format. Thus data extraction systems need to explore and understand the interface designed for humans: A system needs to automatically navigate the search or browse interface **(1)**, typically forms, provided by the site to get to result pages. On the result pages **(2)**, it automatically identifies and separates the individual objects and aligns them with their attributes. The attribute alignment may then be refined on the details pages **(3)**, i.e., pages that provide comprehensive information about a single entity. This involves some of the most challenging analysis, e.g., to find and extract attribute-value pairs from tables, to enrich the information about the object from the flat text description, e.g., with relations to known points-of-interest, or to understand non-textual artefacts such as floor plans, maps, or energy performance charts. All that information is cleaned and integrated **(4)** with previously extracted information to establish a large database of all objects extracted from websites in that domain. If fed with a sufficient portion of the websites of a domain, this database provides a comprehensive picture of all objects of the domain.

That *domain knowledge* is the solution to high-accuracy data extraction at scale is not entirely new. Indeed, recently there has been a flurry of approaches focused on this idea. Specifically, domain-specific approaches use background knowledge in form of ontologies or instance databases to replace the role of the human in supervised, site-specific approaches. Domain knowledge comes in two fashions, either as instance knowledge (that "Georg" is a person and lives in the town "Oxford") or as schema or ontology knowledge (that "town" is a type of "location" and that "persons" can "live" in "locations"). Roughly, existing approaches can be distinguished by the amount of schema knowledge they use and whether instances are recognised through annotators or through redundancy. One of the dominant issues when dealing with automated annotators is that *text annotators have low accuracy*. Therefore, [6] suggests the use of a top-k search strategy on subsets of the annotations provided by the annotators. For each subset a separate wrapper is generated and ranked using, among others, schema knowledge. Other approaches exploit *content redundancy*, i.e., the fact that there is some overlapping (at least on the level of attribute values) between web sites of the same domain. This approach is used in [11] and an enumeration of possible attribute alignments (reminiscent of [6]). Also [2] exploits content redundancy, but focuses on redundancy on entity level rather than attribute level only.

Unfortunately, all of these approaches are only half-hearted: They add a bit of domain knowledge here or there, but fail to exploit it in other places. Unsurprisingly, they remain stuck at accuracies around $90 - 94\%$. There is also no single system that covers the whole data extraction process, from forms over

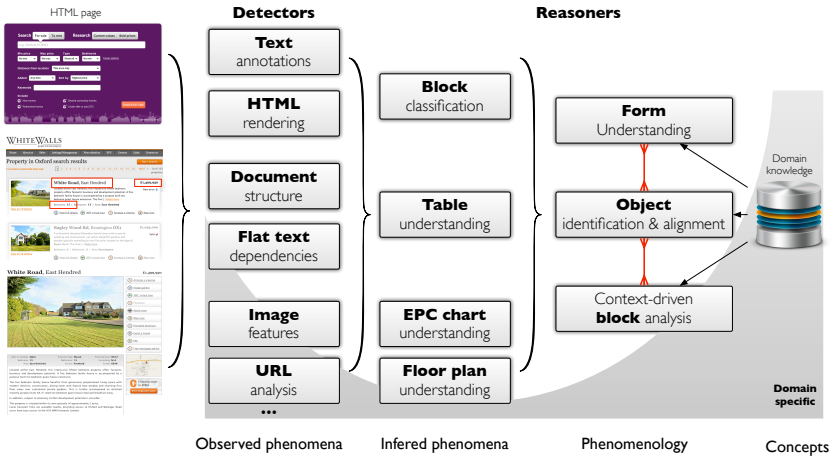


Fig. 2: DIADEM knowledge

result pages to details pages, but rather most either focus on forms, result or details pages only.

To address these shortcomings, we introduce the **DIADEM engine** which demonstrates that through domain-specific knowledge in all stages of data extraction we can indeed achieve high accuracy extraction for entire domain. Specifically, DIADEM implements the full data extraction pipeline from Figure 1 integrating form, result, and details page understanding. We discuss DIADEM, the way it uses domain knowledge (Section 2) and performs an integrated analysis (Section 3) of a web site of a domain in the rest of this paper, concluding with a set of preliminary results (Section 4).

2 DIADEM Knowledge

DIADEM is organised around knowledge of three types, see Figure 2:

1. *What to detect?* The first type of knowledge is all about detecting instances, whether instances of domain entities or their attributes, or instances of a technical concept such as a table, a strongly highlighted text, or an advertisement. We call such instances **phenomena** and distinguish phenomena into those that can be directly *observed* on a page, e.g., by means of text annotators or visual saliency algorithms, and those *inferred* from directly observed ones, e.g., that similar values aligned in columns, each emphasising its first value, constitute a table with a header row.

2. *How to interpret?* However, phenomena alone are fairly useless: They are rather noisy with accuracy in the $70-80\%$ range even with state of the art techniques. Furthermore, they are not what we are interested in: We are interested in structured objects and their attributes. How we assemble these objects and assign their attributes is described in the **phenomenology** that is used by a set of reasoners to derive structured instances of domain concepts from the phenomena. Thus a table phenomenon may be used together with price and location

annotations on some cell values and the fact that there is a price refinement form to recognise that the table represents a list of real estate offers for sale. Similarly, we assemble phenomena into instances of high-level interaction concepts such as real-estate forms or floor plans, e.g., to get the rooms and room dimensions from edge information and label annotations of a PDF floor plan.

3. *How to structure?* Finally, the domain knowledge guides the way we structure the final data and resolve conflicts between different interpretations of the phenomena (e.g., if we have one interpretation that a flat has two bedrooms and one that it has 13 bedrooms, yet the price is rather low, it is more likely a two bedroom flat).

For all three layers, the necessary knowledge can be divided into domain-specific and domain-independent. For quick adaptability of DIADEM to new domains, we formulate as much knowledge as possible in general, domain independent ways, either as reusable components, sharing knowledge, e.g., on the UK locations between domains, or as domain independent templates which are instantiated with domain specific parameters. Thus, to adapt DIADEM to a given domain, one needs to select the relevant knowledge, instantiate suitable templates, and sometimes provide additional, truly domain specific knowledge.

Where phenomena (usually only in the form of textual annotators) and ontological knowledge are fairly common, though never applied to this extent in data extraction, DIADEM is unique in the use of explicit knowledge for the mapping between phenomena. These mappings (or phenomenology) are described in $\mathsf{Datalog}^{\pm,\neg}$ rules and fall, roughly, into three types that illustrate three of the most profligate techniques used in the DIADEM engine:

1. *Finding repetition.* Fortunately, most database-backed websites use templates that can be identified with fair accuracy. Exploiting this fact is, indeed, the primary reason why DE systems are so much more accurate that IE that do not use this information. However, previous approaches are often limited by their inability to distinguish noise from actual data in the repetition analysis (and thus get, e.g., confused by different record types or irregular advertisements). Both is addressed in DIADEM by focusing the search for repetition carrying relevant phenomena (such as instances of domain attributes).

2. *Identifying object instances through context.* However, for details pages not enough repetition may be available and thus we also need to be able to identify *singular* object occurrences. Here, we exploit context information, e.g., from the search form or from the result page through which a details page is reached.

3. *Corroboration of disparate phenomena.* Finally, individual results obtained from annotations and patterns must be corroborated into a coherent model, building not only a consistent model of individual pages but of an entire site.

3 DIADEM Engine

All this knowledge is used in the DIADEM engine to analyse a web site. It is evident that this analysis process is rather involved and thus not feasible for every single page on a web site. Fortunately, we can once again profit from the

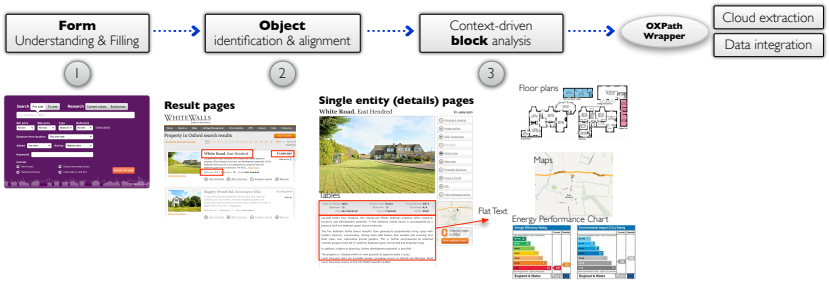


Fig. 3: DIADEM pipeline

template structure of such sites: First, DIADEM analyzes a small fraction of a web site to generate a wrapper, and second, DIADEM executes these wrappers to extract all relevant data from the analyzed sites at high speed and low cost. Figure 3 gives an overview of the high-level architecture of DIADEM. On the left, we show the analysis, on the right the execution stage. In practice, there are far more dependencies and feedback mechanisms, but for space reasons we limit ourselves to a sequential model.

In the first stage, with a sample from the pages of a web site, DIADEM generates *fully automatically* wrappers (i.e., extraction program). This **analysis** is based on the knowledge from Section 2, while the extraction phase does not require any further domain knowledge. The result of the analysis is a wrapper program, i.e., a specification how to extract all the data from the website without further analysis. Conceptually, the analysis is divided into three major phases, though these are closely interwoven in the actual system:

(1) *Exploration:* DIADEM automatically explores a site to locate relevant objects. The major challenge here are *web forms*: DIADEM needs to understand such forms sufficiently to fill them for sampling, but also to generate exhaustive queries for the extraction stage, such that *all* the relevant data is extracted (see [1]). DIADEM's form understanding engine OPAL [8] uses an phenomenology of relevant domain forms for these tasks.

(2) *Identification:* The exploration unearths those web pages that contain actual objects. But DIADEM still needs to identify the precise boundaries of these objects as well as their attributes. To that end, DIADEM's result page analysis AMBER [9] analyses the repeated structure within and among pages. It exploits the domain knowledge to distinguish noise from relevant data and is thus far more robust than existing data extraction approaches.

(3) *Block analysis:* Most attributes that a human would identify as structured, textual attributes (as opposed to images or flat text) are already identified and aligned in the previous phase. But DIADEM can also identify and extract attributes that are not of that type by analysing the flat text as well as specific, attribute-rich image artefacts such as energy performance charts or floor plans. Finally, we also aim to associate "unknown" attributes with extracted objects, if

these attributes are associated to suitable labels and appear with many objects of the same type,
At the end of this process, we obtain a sample of instance objects with rich attributes that we use to generate an OXPath wrapper for extraction. Some of the attributes (such as floor plan room numbers) may require post-processing also at run-time and specific data cleaning and linking instructions are provided with the wrapper.

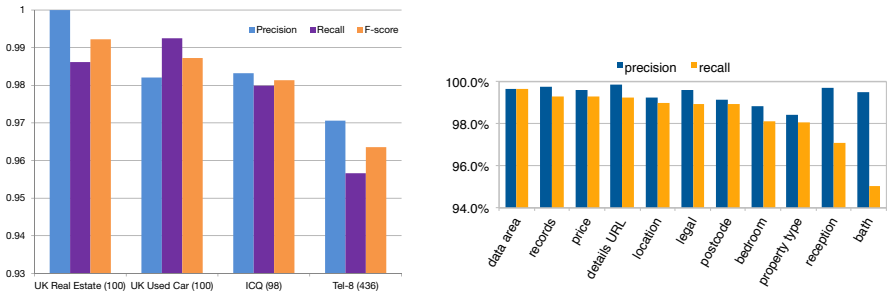
The wrapper generated by the analysis stage can be **executed** independently. We have developed a new wrapper language, called OXPath [10], the first of its kind for large scale, repeated data extraction. OXPath is powerful enough to express nearly any extraction task, yet as a careful extension of XPath maintains the low data and combined complexity. In fact, it is so efficient, that page retrieval and rendering time by far dominate the execution. For large scale execution, the aim is thus to minimize page rendering and retrieval by storing pages that are possibly needed for further processing. At the same time, memory should be independent from the number of pages visited, as otherwise large-scale or continuous extraction tasks become impossible. With OXPath we obtain all these characteristics, as shown in Section 4.

4 DIADEM Results

To give an impression of the DIADEM engine we briefly summarise results on three components of DIADEM: its form understanding system, OPAL; its result page analysis, AMBER; and the OXPath extraction language.

Figures 4a and 4b report on the quality of form understanding and result page analysis in DIADEM's first prototype. Figure 4a [8] shows that OPAL is able to identify about 99% of all form fields in the UK real estate and used car domain correctly. We also show the results on the ICQ and Tel-8 form benchmarks, where OPAL achieves $> 96\%$ accuracy (in contrast recent approaches achieve at best 92% [7]). The latter result is without use of domain knowledge. With domain knowledge we could easily achieve close to 99% accuracy as well. Figure 4b [9] shows the results for data area, record, and attribute identification on result pages for AMBER in the UK real estate domain. We report each attribute separately. AMBER achieves on average 98% accuracy for all these tasks, with a tendency to perform worse on attributes that occur less frequently (such as the number of reception rooms). AMBER is unique in achieving this accuracy even in presence of significant noise in the underlying annotations: Even if we introduce an error rate of over 50%, accuracy only drops by 1 or 2%.

For an extensive evaluation on OXPath, please see [10] . It easily outperforms existing data extraction systems, often by a wide margin. Its high performance execution leaves page retrieval and rendering to dominate execution ($> 85\%$) and thus makes avoiding page rendering imperative. We minimize page rendering by buffering any page that may still be needed in further processing, yet manage to keep memory consumption constant in nearly all cases including extraction tasks of millions of records from hundreds of thousands of pages.



(a) OPAL (b) AMBER

Fig. 4: DIADEM results

References

1. Benedikt, M., Gottlob, G., Senellart, P.: Determining relevance of accesses at runtime. In: *PODS*. (2011)
2. Blanco, L., Bronzi, M., Crescenzi, V., Merialdo, P., Papotti, P.: Exploiting Information Redundancy to Wring Out Structured Data from the Web. In: *WWW*. (2010)
3. Crescenzi, V., Mecca, G.: Automatic information extraction from large websites. *J. ACM* **51**(5) (2004) 731–779
4. Dalvi, N., Bohannon, P., Sha, F.: Robust web extraction: an approach based on a probabilistic tree-edit model. In: *SIGMOD*. (2009)
5. Dalvi, N., Machanavajjhala, A., Pang, B.: An analysis of structured data on the web. In: *VLDB*. (2012)
6. Dalvi, N.N., Kumar, R., Soliman, M.A.: Automatic wrappers for large scale web extraction. In: *VLDB*. (2011)
7. Dragut, E.C., Kabisch, T., Yu, C., Leser, U.: A hierarchical approach to model web query interfaces for web source integration. In: *VLDB*. (2009)
8. Furche, T., Gottlob, G., Grasso, G., Guo, X., Orsi, G., Schallhart, C.: Opal: automated form understanding for the deep web. In: *WWW*. (2012)
9. Furche, T., Gottlob, G., Grasso, G., Orsi, G., Schallhart, C., Wang, C.: Little knowledge rules the web: Domain-centric result page extraction. In: *RR*. (2011)
10. Furche, T., Gottlob, G., Grasso, G., Schallhart, C., Sellers, A.: Oxpath: A language for scalable, memory-efficient data extraction from web applications. In: *VLDB*. (2011)
11. Gulhane, P., Rastogi, R., Sengamedu, S.H., Tengli, A.: Exploiting content redundancy for web information extraction. In: *VLDB*. (2010)
12. Lin, T., Etzioni, O., Fogarty, J.: Identifying interesting assertions from the web. In: *CIKM*. (2009)
13. Liu, W., Meng, X., Meng, W.: Vide: A vision-based approach for deep web data extraction. *TKDE*. **22** (2010) 447–460
14. Simon, K., Lausen, G.: Viper: augmenting automatic information extraction with visual perceptions. In: *CIKM*. (2005)
15. Yates, A., Cafarella, M., Banko, M., Etzioni, O., Broadhead, M., Soderland, S.: Textrunner: open information extraction on the web. In: *NAACL*. (2007)
16. Zheng, S., Song, R., Wen, J.R., Giles, C.L.: Efficient record-level wrapper induction. In: *CIKM*. (2009)