

Wrapper Induction and Maintenance in Documentum ECI

Boris Chidlovskii
Xerox Research Centre
France
chidlovskii@xrce.xerox.com

Bruno Roustant
Documentum, IDE
France
roustant@documentum.com

Marc Brette
Documentum, IDE
France
brette@documentum.com

ABSTRACT

Documentum Enterprise Content Integration (ECI) services is a content integration middleware that provides one-query access to the Intranet and Internet content resources. The ECI Adapter technology offers an interface to any application for data and metadata extraction from unstructured Web pages. It offers a unique framework of wrapper production, automatic recovery and maintenance, developed at Xerox Research Centre Europe and based on state-of-art algorithms from machine learning and grammatical inference. In this paper we analyze the performance of ECI adapters deployed in current commercial installations. We benefit from accessing reports on daily tests for all ECI commercially deployed adapters collected from June 2003 to December 2005. Using the daily reports, we analyze different aspects of the wrapper technology.

Categories and Subject Descriptors

H.4.m [Information Systems Applications]: Miscellaneous; I.2.6 [Learning]: Induction; D.2.8 [Software Engineering]: Metrics—*performance measures*

General Terms

Algorithms, Performance, Measurement

Keywords

Documentum, information integration, Web wrappers

1. INTRODUCTION

During the last decade, wrapping Web sources has been a hot research topic in the data extraction, mediation and integration domains. Nowadays, despite the progress in XML publishing and Web services, wrapping Web sites that publish human-oriented HTML pages remains an important component of Semantic Web efforts.

The diversity of different wrapping techniques proposed in data mediation and machine learning communities is remarkable [10, 12, 13, 14]. Several efforts to survey and classify existing approaches and techniques have been undertaken in [1, 5]. However, the major open issue remains the comparative evaluation of different wrapping methods. Similarly to the situation in information extraction [11], comparing different methods remains a challenge for the reasons of lack of agreed evaluation testbed and multiple inconsistencies in the experimental procedures.

Furthermore, requirements to wrapper technologies have essentially evolved over the last decade. This evolution went through

three major steps. Initially, the wrapper techniques were competing in the *expressiveness* (i.e., a capacity to accurately wrap any given Web site) and *adaptability* (i.e., the automatic generation without coding). Later, the need for wrapper maintenance imposed additional requirements, such as *robustness* and *automatic repairing* of wrappers. Finally, the commercial deployment of wrappers add requirements of *scalability* and *optimal quality control*.

In the absence of an agreed evaluation testbed, in this paper we report on the commercial deployment of one specific wrapper product, namely, the ECI (former AskOnce) wrapper technology developed at Xerox Research Centre Europe in 1999-2002.

The first commercial deployment of AskOnce wrapper technology (version 3) dated 2000; it was addressing primarily the wrapper production needs. Version 3 included the Graphical Wrapper Toolkit, where any user with no programming skills could induce an accurate Web wrapper from few example pages [2]. In 2002, AskOnce committed version 4, where the wrapper architecture was extended with the maintenance component [4]. In 2004, the AskOnce content integrator has been acquired by Documentum to become a part of ECI Services [9]. Currently, ECI wrappers are deployed in more than 100 full Documentum installations in Europe and the United States serving 100,000 to 150,000 users.

Among Web content providers solicited by ECI clients, some do cooperate with content integrators by offering an access to data through the Web services, XForm querying, XML publishing or RSS feeding. However, a vast majority of domain-specific Web providers do not cooperate with integrators; they target the human visitors only and support the conventional mechanisms of PHP/CGI querying and human-oriented HTML publishing.

We analyze the commercial performance of Web wrappers over 31 months, from June 2003 to December 2005, including the migration from version 3 to version 4. We benefit from a unique opportunity of accessing reports on wrapper tests run daily during the whole period of commercial deployment. The daily reports account for a total of 2,200,000 tests for about 500 wrappers. To our best knowledge, this is the first attempt of conducting such a study.

Since wrappers are deployed in a large-scale environment and communicate with non-cooperative sites, ECI-4 pays a particular attention to the wrapper maintenance. Once a wrapper production platform became operational, the creation of wrapper instances for a majority of Web providers represents an important but one-time investment. Instead, in the content integrator life cycle, the wrapper maintenance cost is considerably superior to the initial investment in the wrapper creation. The goal of the wrapper maintenance component consists in reducing the maintenance cost but still guaranteeing the top quality of extracted data.

The second issue of our study is how wrapper technologies get accommodated to the commercial deployment context. The hu-

man factor and the commercial deployment policy reveal specific requirements of scalability, an optimal quality control, etc. We discuss them and their impact on the ECI technology.

2. ECI ADAPTER MECHANISM

Documentum Enterprise Content Integration (ECI) services [9] is a content integration middleware that provides one-query access to the Intranet and Internet content resources. ECI services enhance enterprise’s solutions in place without imposing any change and respect native security policies. The ECI functionalities include search, synthesis, sharing and integrating information, dynamic linguistic clustering, personalized ranking, cross-lingual search, scheduled queries and notification [9]. The ECI Adapter technology offers an interface to any content resource, including the mechanism of query mapping capabilities, data and metadata extraction from semistructured Web pages. It represents a unique framework for rapid adapter production and maintenance.

The current adapter library [8] covers content providers (Factiva, Lexis Nexis), Web providers full-text engines (Verity, PortalOne, MS Index Server, etc.), enterprise repositories for Documentum products, Lotus Notes, Lotus Domino, MS SiteServer, MS Exchange, Oracle, JDBC/ODBC, etc. It also packs available adapters in domain-specific “bundles”. Currently, bundles of 10 to 30 adapters are available for aerospace, computer and pharmaceutical industries, science, legislation and some other domains.

3. WRAPPER TECHNOLOGY

Wrapping Web providers is a special case of the information extraction (IE) problem. Web pages processed by wrappers are more structured than pages processed by conventional IE methods that target unstructured text like FAQ or seminar announces [11]. Instead, wrappers often can achieve a high accuracy of information extraction, with both precision and recall being as least 98%, whereas such a high accuracy is often unreachable with IE from plain text. The high accuracy is critical in data integration and Semantic Web solutions since the extracted data is often entailed with sophisticated post-processing operations, like an answer rating or a database-like join operation, which are very sensitive to inaccurate data. In the following, *Web wrapping* will refer to a *highly accurate information extraction from HTML pages*.

A majority of wrapper techniques assume that no changes can happen to the pages; as consequence, wrappers often become brittle when, for some reasons, the page mark-up or structure is changed. The wrapper maintenance is challenging in cases when provider pages undergo massive and sweeping modifications, due to a complete redesign. In much more frequent cases of *concept shift*, provider’s pages undergo *local and small changes*. It is therefore important that a wrapper platform includes maintenance components capable to recover from small changes.

The *automatic wrapper maintenance* ensures the robustness of a wrapper under the assumption of small changes. When a provider changes the page format, the wrapper runs in an error and triggers the recovery procedure which attempts to resume the information extraction and repair the wrapper. It is common to distinguish between two sequential maintenance steps, *extraction recovery* and *wrapper repairing* [13]. The extraction recovery resumes the labeling of tokens in a page; it is aimed at extracting as much relevant data as possible. The wrapper repairing becomes possible only if the recovery went successfully; pages in the new format can be automatically re-labeled and extraction rules can be re-learned to match the new page format.

The wrapper maintenance in ECI-4 is based on the ensemble

principle [7]. Ensembles of classifiers (views) are more accurate than individual classifiers, provided that the individual classifiers are *independent* (their errors are uncorrelated) and *accurate* (error rates are better than a random guess). While the accuracy of a classifier depends on chosen learning methods, the independence of classifiers can be ensured if different and uncorrelated features are selected to build the classifiers [7].

For the view independence, ECI considers two disjoint sets of *context* and *content* features. A context feature of a token in HTML characterizes its surroundings, that is, tags and tokens that precede (prefix) or follow the token (suffix). Instead, content features characterize the string itself, like the token length or number of words.

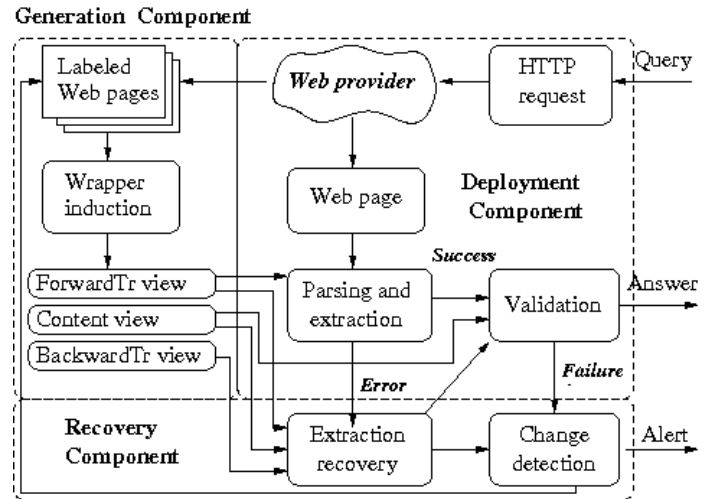


Figure 1: ECI wrapper architecture.

Figure 1 shows the ECI wrapper architecture with three major components, Generation, Deployment and Recovery. A wrapper life cycle begins in the Generation component, with labeling sample HTML pages fetched from a given Web provider. Then, an induction method generalizes the labeled pages into a wrapper instance which is used in Deployment component to process any new provider’s page. All data extracted by the wrapper gets validated. If the provider changes format of HTML pages, the wrapper can fail to complete the data extraction. In such a case, an extraction recovery procedure tries to extract as many data as possible from the page. A successful recovery results in automatic re-labeling of new pages which can be used to generate a new wrapper version that accommodates the new page format.

3.1 Wrapper induction and recovery

The wrapper induction in ECI is a special case of *sequential learning* [6], where an input sequence of tokens is labeled with a sequence of labels from a set L . ECI develops alternative and redundant views of pages; these views are useful for the recovery in the case of concept shift. Three alternative views on the sequential learning, given by 1) a *forward transducer* which parses the input file from the beginning to the end, 2) a *backward transducer*, parsing the file in the opposite direction, and 3) a *content classifier*.

Forward transducers. The basic ECI wrapper component has a form of *string transduction*, where input sequences of tokens from set T are transduced in output sequences over a label set L . In regular transducers, consuming an input token does not necessarily lead to emitting an output symbol, because of facing multiple emission choices in transducer states; in such cases the emission is

postponed till reaching a state where the ambiguity is resolved. ECI transducers extends the transducer induction in [15] and represents transducers as sets of extraction rules [3].

Backward transducers. Backward transducers are an alternative view on HTML pages processed by a wrapper. A backward transducer scans a file backward; its extraction rules use optimal and minimal set of *labeled suffices* and *unlabeled prefixes* to uniquely label textual tokens. Like the forward transducer $T_r^{forward}$, the backward transducer T_r^{back} is partial and can run in error when the format changes. However, it would fail at positions different from those where the forward transducer would. Therefore, backward extraction rules can help complete information extraction in positions where the forward wrapper fails.

Content classifier. To classify textual tokens by content features only, ECI extracts a set F_C of $k=54$ content features, including 42 syntactic and 12 semantic ones. *Syntactic features* include the token length, word counts, density of digits, and standard delimiters (comma, semicolon, dot, etc.). *Semantic features* count typed components of textual tokens, such as proper names, url and time strings and noun phrases.

Content classifier C is trained on the content features of textual tokens in sample pages. Any of existing techniques for classifier generation can be used here; ECI uses decision trees available with Weka package [16]. For textual token t , classifier C returns a pair $C(t) = (l_c, pr)$ where l_c is the most probable label for t , $l_c \in L$ and pr is its probability. Similarly, $C(t, l)$ returns the probability of labeling token t with l .

The content classifier C validates information extracted by the forward transducer in the basic page parse. If, for the current token t , $T_r^{forward}$ finds a matching rule with label l_{Tr} , t is labeled with l_{Tr} only if C validates l_{Tr} by observing content features of t , for some threshold validation value, $C(t, l_{Tr}) \geq thVal$.

Multi-scan recovery. The recovery algorithm is multi-scan, it combines the forward and backward transducers with the content classifier and tries to take an advantage of each view in the most appropriate manner [4]. The recovery algorithm starts by scanning a page from the place where the parsing error took place. It keeps probing the transducer $T_r^{forward}$ with a current token t ; if $T_r^{forward}$ finds a matching rule with label l_w , t is labeled with l_w iff C validates the label. If an error occurs, C provides the most probable label l_c for t . If the confidence of l_c is superior to a given threshold value $thRec$, t is labeled with l_c , otherwise t remains unlabeled. The algorithm switches the scan direction and tries to label not yet labeled textual tokens probing their context with forward and backward wrappers and content classifiers. Algorithm stops when none of the tokens is labeled during the last scan.

Content classifier C plays a double role in the recovery algorithm and intervenes with two threshold parameters. The validation threshold $thVal$ confirms the label choice done by the forward transducer, and therefore it is lower than recovery threshold $thRec$ in cases when the transducer runs in error and labeling decision is made only by the content view, $thVal < thRec$.

The recovery can trigger the automatic wrapper repairing if the recovery went well and all tokens have been labeled with a given threshold of accuracy. It can then automatically re-label sample pages and use them as input to the automatic re-train the wrapper.

4. DEPLOYMENT AND TESTING

The ECI adapters reside in central and client repositories. Client adapters operate independently from the central storage, they can evolve through the automatic maintenance and repairing procedures. In the case of visible problems with adapters, the client downloads the correct and up-to-date adapter instances from the central repos-

itory. Optionally, new versions can be uploaded automatically.

The central repository maintains fully operational and correct adapters. To maintain all wrappers and to promptly detect possible concept shifts, all working wrapper instances in the central repository are tested once a day, at midnight (GMT+1). Each wrapper instance is tested with a series of associated queries. Each query can return multiple pages for which it deploys a number of associated wrappers. There are three possible status given to a test:

Success: the wrapper completes the data extraction from answer pages and extracted data satisfies validation criteria. The success status can be achieved either by the basic parse or through applying the recovery procedure.

Error: the wrapper fails to complete the extraction process, even after the recovery.

Failure: the wrapper successfully completes the extraction process, but extracted data does not meet post-processing constraints set by the designer.

Constraints that cause the failure status are set by the designer upon the wrapper output. The most frequent case is a condition on the number of answers to be returned. If the provider returns less answers than expected, this triggers the failure status.

The failure status can be caused by any communication dysfunction between the ECI integration platform and a multitude of Web providers. ECI distinguish between internal and external causes. Internal causes come from other adapter components, like a query translation, for example, due to a change in the provider's query language. External causes of the failure status include unavailable service or long delays at provider's side, and network traffic.

The maintenance charges are distributed among members of ECI team, with the same person being responsible for both the wrapper creation and its further maintenance. In the case of error and failure, the wrapper designer receives an alert and test details. The designer may take no action if she considers the wrapper operational (*false alert*). If the situation is considered as a concept shift, she makes the wrapper evolve.

The wrapper evolution can be done in one of two modes. In the *evolutionary mode*, the wrapper training set is extended with new pages (where the errors were reported), without removing the previous pages. In the *breakout mode*, the wrapper is created from scratch. The decision on the evolution mode is left to the wrapper designer. The evolutionary mode requires less time and often represents a sufficient remedy for small format changes, as an important number of extraction rules may remain valid. Often, a high recovery performance is a clear indication for a minor concept change. Instead, a low recovery performance indicates an important change and is often followed by inferring the wrapper from new pages only.

The wrapper algorithms in Section 3.1 address the core requirements to Web wrappers, namely, their expressiveness, adaptability and automatic maintenance. They are rather sufficient for research prototypes and modest wrapper collections. However, in the context of the large-scale commercial deployment, the ECI team found it important to impose two additional requirements:

Scalability: with a growing number of wrappers to maintain, the set of parameters controlling the wrapper performance should not depend on the wrapper number.

Easy quality control: the technology should allow an easy adaptation and a smart support in the wrapper quality control.

To meet the deployment requirements, ECI-4 wrappers underwent certain adjustments. The target wrapper accuracy level was

set to 98%. The tokenization and feature selection remain specific to each wrapper, in order to ensure its optimal adaptation to the providers' formats. The major adjustment concerns the two threshold parameters $thVal$ and $thRec$ controlling the basic scan and the recovery procedure for each wrapper. Since controlling individual threshold values represents a high burden to the support team, the thresholds are set the same values for all wrappers. For the administrator, these thresholds offer a powerful and simple mechanism to globally control the wrapper performance and maintenance, in the function of quality criteria for a given installation.

5. PERFORMANCE ANALYSIS

In this section, we analyze the WCI wrapper performance using the daily wrapper test reports. For ECI-3, we dispose 418 daily reports covering the period from June 1st, 2003 to October 31st, 2004. They account for a total of 611,987 test queries, with an average of 1464.1 tests per day and 11.6 tests per wrapper. For ECI-4, 598 daily reports cover the period from February 1st, 2004 to December 31st, 2005. In total, they account for 487,558 queries, with an average of 843.36 tests per day and 6.03 tests per wrapper.

ECI-3 reports included only the necessary information (global query status, execution time and wrapper version). During the migration from version 3 to version 4, the daily reports have been extended with detailed statistics on the wrapper performance for each page processed by the wrapper. In total, tests covered 2,262,735 pages with a minimum 1 page and a maximum 107 pages per query. Additionally, each entry reports whether the recovery was called, and if so, the accuracy of information extraction before and after the recovery. Finally, we could access the internal representation of wrapper components (in their latest versions) to analyze the transducers and the content classifiers.

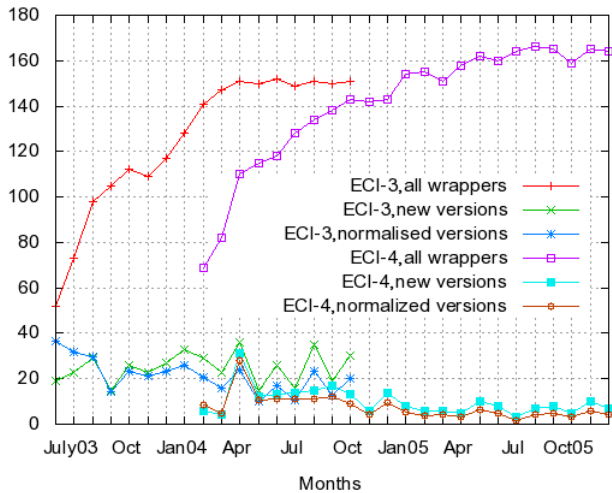


Figure 2: The wrapper set evolution (the total and new versions) for ECI-3 and ECI-4.

5.1 Wrappers and new versions

We start with a global view on the wrapper evolution for both versions. The major challenge of moving from ECI-3 to ECI-4 was the maintenance effort reduction. The permanent changes and the brittleness of wrappers in ECI-3 led to extending the basic wrapper architecture with the recovery component in ECI-4.

The main maintenance cost can be expressed in the number of new wrapper versions. Figure 2 presents a global view on the wrapper maintenance effort for both versions 3 and 4. The figure reports the total number of wrappers growing over months, and the number of new versions for existing wrappers committed monthly. Both absolute and normalized numbers (per 100 wrappers) of new versions are reported. The transition period of migrating ECI-3 wrappers to version 4 lasted 9 months, from April 2004 till October 2004 when the maintenance of ECI-3 wrappers was discontinued. As the figure suggests, the migration allowed to reduce the number of new versions from 21.2 to 7.3 (per month and per 100 wrappers).

5.2 Wrapper components

Out of 214 wrappers deployed at least once with ECI-4, we have selected eight top groups with wrappers present in at least 20% of daily tests. The eight selected groups are Aerospace (7 wrappers), Computer (9), General (8), Health (8), Library (4), Regulation (15), News (2) and Science (11). Below we analyze the wrapper components. We report on the size of forward and backward transducers (the number of rules), the average confidence of content views and the number of classes.

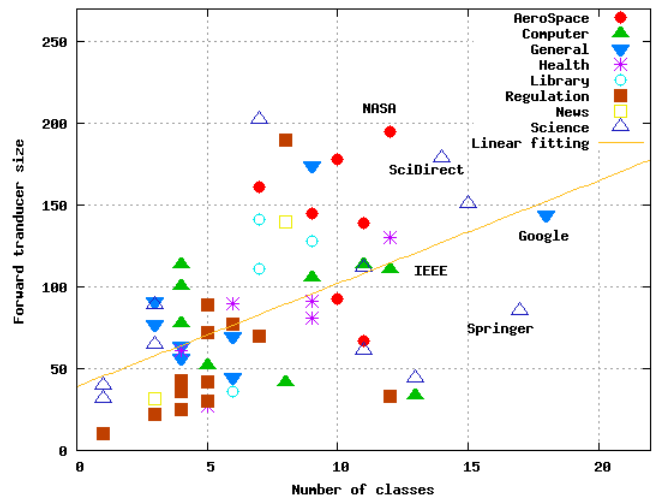


Figure 3: The size of forward transducers versus the number of classes.

Wrappers in selected groups extract data of 1 to 18 classes, with the average of 7.1 classes per wrapper. Forward transducers have 28 to 240 states (see Figure 3), with an average of 85.7 rules per wrapper. For backward transducers, the corresponding numbers are 32, 405 and 109.9. The confidence of content classifiers vary between 71.7% and 100%.

As Figure 3 suggests, there exists a strong correlation between the transducer size and the class number. The more classes, the more rules are needed to disambiguate the class choice from the context. Similarly, Figure 4 shows that the confidence of content classifiers decreases with the number of classes.

A different look at the wrapper components in Figure 5 confirms the observed phenomenon. It shows that a larger size of a forward transducer correlates with a lower prediction power from the content, since both views describe a more complex problem. In other words, for small and simple Web sites, if the concept shift breaks the forward transducer, the content classifier and backward transducer have good chances to accomplish the extraction process. Instead, for complex sites, even their joint deployment in the case of

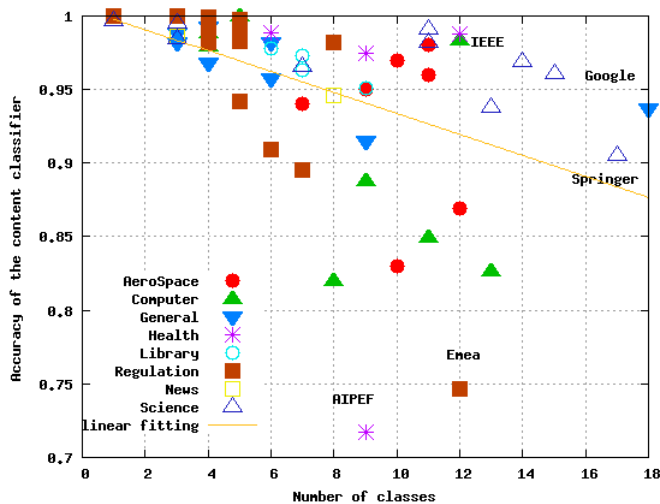


Figure 4: The number of classes versus the confidence of content classifiers.

concept shift may be still insufficient to achieve the 98% level.

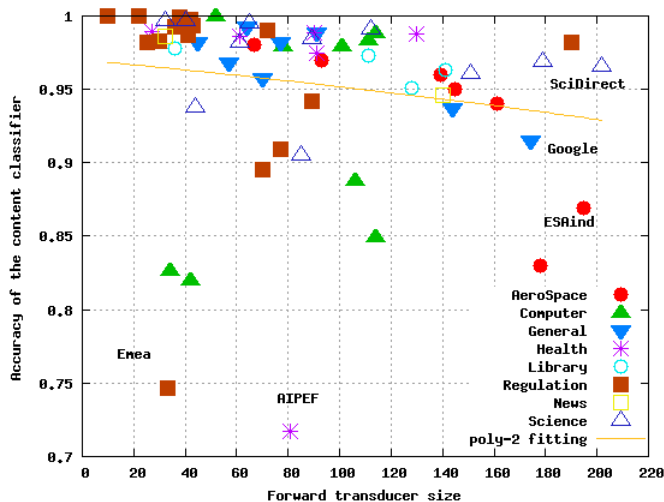


Figure 5: The size of forward transducers versus the confidence of content classifier.

5.3 Daily recovery performance

The wrapper recovery procedure (see Section 3.1) brings an important additional value to the wrapper performance, by increasing their robustness in cases of concept shifts and reducing by 65% the number of human interventions in all commercial installations.

We start by analyzing the Google wrapper which suffered the most from concept shifts; it represents by far the largest number of new versions committed in ECI-4. The initial version was a product of migration from ECI-3, then 18 new versions have been committed since February 2004 to December 2005.

In Figure 6 we plot the day-to-day performance of the Google wrapper. Figure 6.a shows the success, error and failure ratios and Figure 6.b show the basic scan and recovery performance. In both plots, we trace the emission of new versions.

The analysis shows that the emission of a new version is often caused by a sharp drop in the wrapper performance, in particu-

lar, when Google pages underwent important changes in April-June 2004, often invisible to the human eye. Then, for a long period, the top performance of the recovery procedure (superior to 99%) co-existed with an important error ratio (around 8%). So, several new versions have been committed, with a goal of improving the basic transducer and reducing the number of recovery calls and test errors. The last important change took place in September 2005. All wrapper updates are realized in the evolutionary mode, by adding new queries and new annotated pages to the existing training set. Consequently, the number of tests grew from 26 in February 2004 to 39 in December 2005.

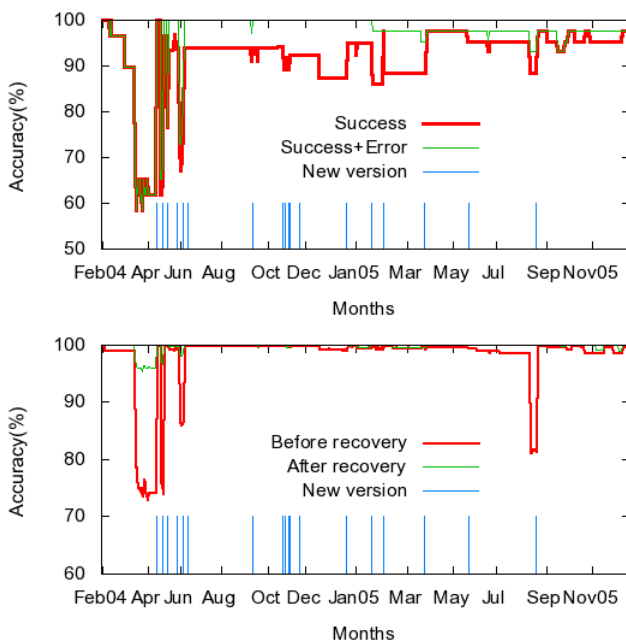


Figure 6: The daily performance of Google wrapper; a) successes, errors and failures; b) basic scan and recovery performance.

Figure 7 plots the day-to-day performance of 214 wrappers in ECI-4. The average failure ratio is about 3%; it remained stable during the testing period, with an exception of few starting months of migration from the previous version.

The error ratio varies around 7%. The fluctuation of values are characteristic to any open or noisy systems, with several important performance drops at the beginning of the testing period. In Figure 7.b, we show the accumulative scan and recovery performance, which particularly suffered during the migration period before achieving the optimal performance level.

5.4 Global recovery performance

We conclude the analysis by the global performance of the recovery algorithm. We estimate the impact the recovery has on the final wrapper accuracy. In Figure 8 we present the two-dimensional grid where x and y axes show five ranges of the wrapper accuracy before (x axis) and after the recovery (y axis). The value in a cell $[x,y]$ indicates the percentage of wrapper tests happened in range x before recovery and in range y after recovery.

Five selected ranges are $[0,90[$, $[90,95[$, $[95,98[$, $[98,100[$ and $[100]$. The largest value (71.2%) goes to the cell $([100],[100])$, where the basis transducer performs well and no recovery was called for. For 11.5% of parses, the extraction was less than 100% but

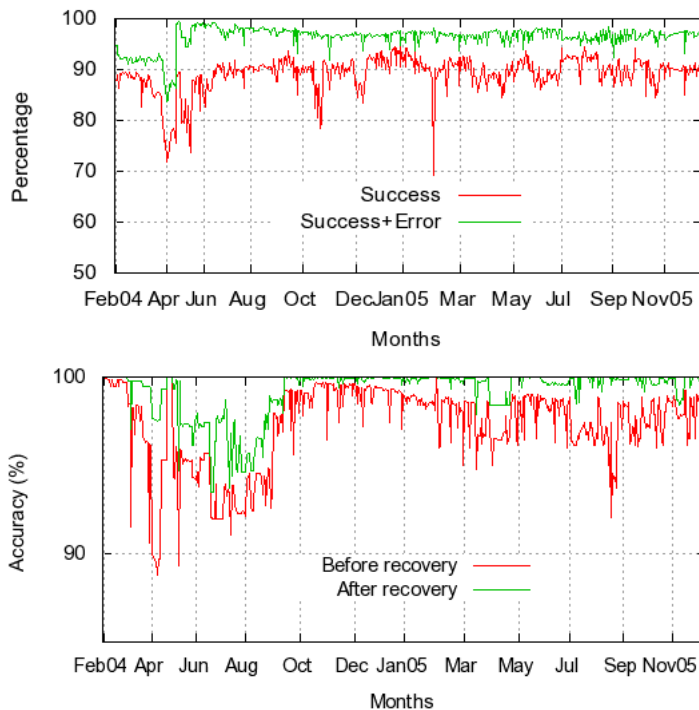


Figure 7: Daily wrapper and recovery performance for ECI-4.

more than 98%, and 93.9% of them have been recovered to 100%.

A similar situation takes place for other ranges. In general, 28.8% parses failed to achieve 100%, and the recovery managed to achieve 100% result for 77.8% of them. Similarly, out of 17.3% of tests that failed to reach the 98% level, the recovery helped 82.6% to achieve 98%, and 67.1% to achieve 100%. Only 4.0% of tests remained under 98% after the recovery.

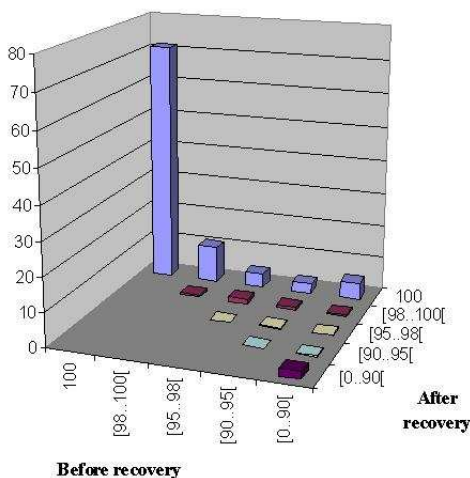


Figure 8: Recovery performance for ECI-4, by ranges.

5.5 False positives and false negatives

A *false negative* (or false alarm) occurs when a wrapper test reports a problem whereas none exists in reality. With an exception of internal causes (see Section 4), a majority of failures are caused by

provider access and network traffic problems; they are of a volatile nature and are therefore *false failures*. A false error often occurs due to insufficient wrapper training or a really complex Web site. An unexpected variation in the page structure would run the transducers in error and the content classifier would be unable to recover the extraction process.

A *false positive* occurs when a wrapper test reports a success, whereas the wrapper extracts data incorrectly. The wrapper test fails to detect an important shift and does not alert the designer. This may happen when a context of a token confuses the transducer and triggers a wrong rule which then fits the validation threshold. Luckily, such a combination happens rarely and was never observed in ECI-4. The second cause is due to incorrectly annotated sample pages used for wrapper induction. If, for some reasons, a badly annotated sample is included in the training set, it can generate a rule that extracts wrong results from new pages. This appears particularly dangerous in the automatic wrapper repairing, where the recovery manages to complete the extraction from test pages and certifies the newly annotated pages as a new training set for inferring a new wrapper version.

A reasonable trade-off between false positives and false negatives can be achieved by tuning the recovery algorithm and alarm criteria. To ensure the wrapper quality, the ECI deployment policy is made more restrictive with respect to false positives, however at the risk of introducing more false alarms. The ECI deployment policy includes the following rules of thumbs:

Recovery parameters. The basic transducers appear to be very accurate with few or none false positives. Instead, the predictive power of content classifiers vary considerably over the entire set of wrappers. In order to meet the scalability requirement and keep one global value *thVal*, the decision was made to set it lower that was initially recommended in [4]. Similarly, after multiple iterations, the recovery threshold value *thRec* was set to 0.86.

Automatic wrapper repairing. An important decision was made of disallowing the automatic wrapper repairing at the central repository, because of a high risk of false positives. The automatic repairing remains activated at client repositories. If an automatically repaired version fails to correctly accomplish the extraction task, the client can retrieve a better controlled version from the central repository.

6. CONCLUSION

We have presented an analysis of ECI wrapper technology deployed in Documentum commercial installations. ECI wrappers deploy a multi-view approach to data extraction from Web pages published by content providers. A set of about 1100 daily wrapper reports collected over 31 months has been used to evaluate the performance of ECI wrappers. In our analysis, we address the issue of wrapper robustness under concept shifts and we pay a particular attention to the maintenance effort reduction. Additionally, we discuss requirements imposed by the large-scale deployment context, such as scalability and easy quality control, and how they influence the technology tuning and customization.

7. REFERENCES

- [1] A. Laender and B. Ribeiro-Neto and A. da Silva and J. Teixeira. A Brief Survey of Web Data Extraction Tools. *SIGMOD Record*, 31(2), 2002.
- [2] D. Bredelet and B. Roustant. Java IWrap: Wrapper Induction by Grammar Learning. Master's thesis, ENSIMAG, Grenoble, France, 2000.

- [3] B. Chidlovskii. Wrapping web information providers by transducer induction. In *European Conference on Machine Learning, Lecture Notes in Computer Science*, volume 2167, pages 61–73, 2001.
- [4] B. Chidlovskii. Automatic Repairing of Web Wrappers by Combining Redundant Views. In *Proc. of 14th IEEE Intern. Conference On Tools with Artificial Intelligence, Washington DC, USA, Nov. 4-6, 2002*, pages 399–406, 2002.
- [5] V. Crescenzi and G. Mecca. Automatic information extraction from large websites. *J. ACM*, 51(5):731–779, 2004.
- [6] T. G. Dietterich. Machine learning for sequential data: A review. In T. Caelli, editor, *Lecture Notes in Computer Science*. Springer-Verlag, 2002.
- [7] T.G. Dietterich. Ensemble methods in machine learning. In *Multiple Classifier Systems, First International Workshop*, pages 1–15. Springer Verlag, 2000.
- [8] Documentum Services ECI Adapter Library. <http://www.documentum.com/products/glossary/al.htm>.
- [9] Documentum Enterprise Content Integration. <http://www.documentum.com/solutions/eci>.
- [10] G. Gottlob, Ch. Koch, R. Baumgartner, M. Herzog, and S. Flesca. The lixto data extraction project: back and forth between theory and practice. In *Proc. 23rd ACM PODS*, pages 1–12, New York, NY, USA, 2004.
- [11] N. Ireson, F. Ciravegna, M.-E. Califf, A. Lavelli, D. Freitag, and N. Kushmerick. Evaluating machine learning for information extraction. In *Proc. Int. Conf. Machine Learning*, 2005.
- [12] N. Kushmerick. Wrapper Induction: Efficiency and Expressiveness. *Artificial Intelligence*, 118:15–68, 2000.
- [13] K. Lerman, S. Minton, and C. A. Knoblock. Wrapper maintenance: A machine learning approach. *Journal of Artif. Intell. Research (JAIR)*, 18:149–181, 2003.
- [14] I. Muslea, S. Minton, and C. Knoblock. A Hierarchical Approach to Wrapper Induction. In *Proc. 3rd Intern. Conf. on Autonomous Agents Conf.*, pages 190–197, 1999.
- [15] J. Oncina, P. Garcia, and E. Vidal. Learning subsequential transducers for pattern recognition interpretation. *IEEE Trans. on Pattern Analysis*, 15:448–458, 1993.
- [16] Ian H. Witten and Eibe Frank. *Data Mining: Practical machine learning tools and techniques, 2nd Edition*. Morgan Kaufmann, San Francisco, 2005.