SeaSpider: automated information gathering on vessel movements in support of marine intelligence, surveillance, and reconnaissance

Serhan Tatar[a], David M.F.Chapman[b]
[a]Dept. of Computer Engineering, Bilkent University, Ankara, TURKEY;
[b]Defence R&D Canada-Atlantic, P.O. Box 1012, Dartmouth, NS B2Y 3Z7, CANADA

ABSTRACT

SeaSpider is an R&D tool to investigate the development of a software agent that would aid an operator in gathering information about marine vessels from public sources on the Internet. This information would supplement sensor information used for Intelligence, Surveillance, and Reconnaissance (ISR) to enhance Maritime Domain Awareness (MDA) and to complete the Recognized Maritime Picture (RMP). Specifically, SeaSpider is fine-tuned to search for, extract, integrate, and display information about locations (ports), dates and times, and activities (arrival, in berth, departure). One module manages World Wide Web (WWW) searches and retrieves the web pages; another module extracts relevant ship activities, integrates them and populates a database; a third module retrieves information from the database in response to user-generated queries. In this paper, the SeaSpider concept is introduced, the design details of the prototype are presented, and performance is analyzed, with a view towards future research.

Keywords: Maritime Domain Awareness, Recognized Maritime Picture, Information Retrieval, Information Extraction.

1. INTRODUCTION

The information world is becoming smaller and more interdependent, closing the gap between international and domestic security. Today, security issues in one part of the world can affect the security of individuals who live in a different part of the world. In this environment of unpredictability and threat, information about intentions, capabilities, and activities of possible threats becomes important. Information superiority plays an important role in making decisions regarding the scope and design of security programs, the allocation of resources, and the deployment of assets [1].

The ability to distinguish important from irrelevant, or friend from foe is necessary in order to counter security threats. However, when the size of the area[1] to be kept under control is considered, it is seen that a certain amount of commitment is needed to acquire and maintain this ability [2].

With the aid of developing technology, many systems have been allocated to decision makers' use for increased awareness, which is among the major goals of the C4ISR (Command, Control, Communications, Computers, Intelligence, Surveillance & Reconnaissance). For example, Automatic Identification System (AIS), a self-reporting navigation and communications system for ships at sea, has been adopted as an information source for Maritime Domain Awareness (MDA). What kind of significance AIS holds for the construction of the Recognized Maritime Picture (RMP), how it might be used for Marine Intelligence, Surveillance, and Reconnaissance (MISR), and what research activities might be conducted in support of AIS for MISR were discussed in [3].

Besides the developed sensors, systems, and applications, public information available on the World Wide Web (WWW) carries some potential for MDA. Although the Internet is used by many individuals for information gathering, its potential for day-to-day MDA has not been completely explored, perhaps because of the heterogeneous nature of the Internet or because of poorly structured data. We believe that the utility of the WWW can be improved by providing software aids to operators.

1 It is reported in [2] that there are around 1700 ships in Canada's Atlantic, Pacific and Arctic areas of responsibility on a typical day (even more, if we consider the non-reported contacts further away from the major regulated ports or vessel traffic management systems).

Optics and Photonics in Global Homeland Security IV, edited by Craig S. Halvorson, Daniel Lehrfeld, Theodore T. Saito
Proc. of SPIE Vol. 6945, 694512, (2008) · 0277-786X/08/$18 · doi: 10.1117/12.778956

The SeaSpider project is an attempt to use information sources found on the Internet in order to improve MISR capability. How can we reduce ISR operators' overload? How can we make the RMP more meaningful? How can we use public information on the Internet more effectively? Our investigation addresses these questions, identifies some implementation challenges, and paves the way for other similar applications.

In this paper, the SeaSpider project is described. The structure of the paper is as follows. The next section describes the project details: objectives of the project, desired functions, challenges and the details of the solution model with the detailed explanation of each module. Section 3 outlines the experimental evaluation of the study. Finally, in the last section we conclude while suggesting paths towards future research.

2. SEASPIDER

The SeaSpider project seeks to enable automated gathering of public information about vessel movements to provide comprehensive assistance to ISR operators. Making the RMP more meaningful by supplying past and future information about vessels is another major goal of the project.

SeaSpider combines several research fields to collect information automatically from the Web. The key objective of the system is to be able to provide accurate information, with a good presentation, in a reasonable time. Supporting several functions, the system focuses on accuracy, speed, and usability. Accepting user-queries with different search criteria, collecting relevant information about vessels of interest (VOIs) from the Web, extracting relevant information from HTML pages, managing the queries and the extracted information, and presenting timeline of ship motion from the past into the future are among the various useful functions of the system.

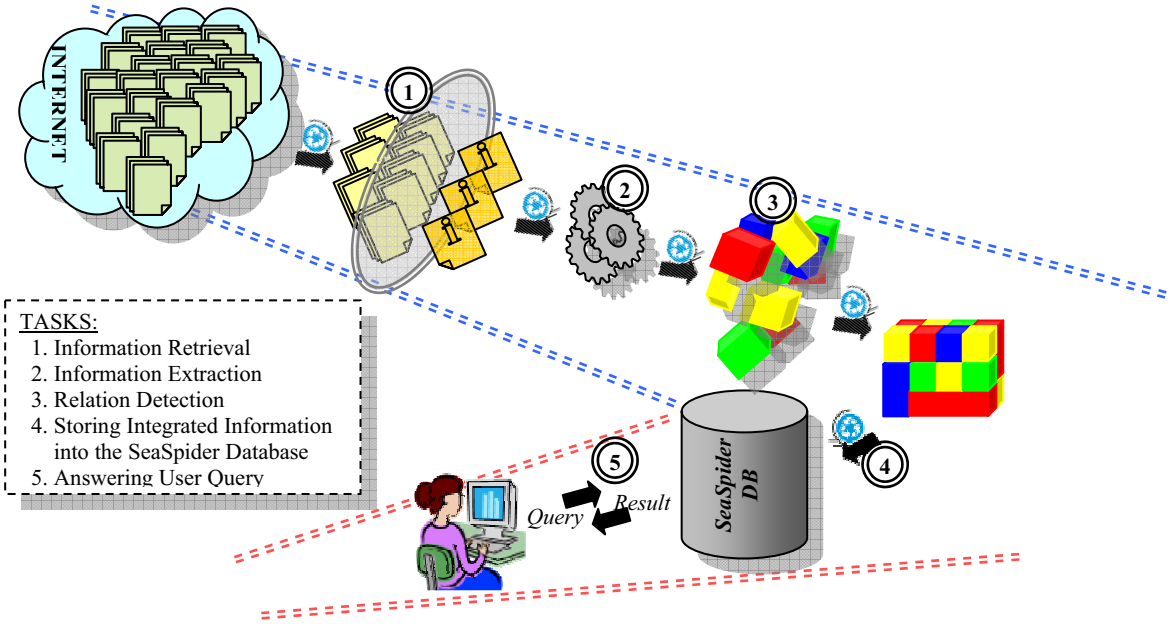

Fig. 1. The general flow of the SeaSpider.

SeaSpider concept contains several research challenges related to different research fields. The biggest challenges are related to information retrieval (IR) and information extraction (IE). Finding and extracting the relevant information from the WWW is the most necessary capability for the system. However, the problem is that there are few comprehensive information sources on the Web. In fact, there are a few commercial sources that include considerable information, but they are not open to the public and require a heavy license fee. The sites that are publicly available are large in number but contain a limited amount of information. Moreover, web pages are generally not structured in a way that makes them easily computer understandable. HTML, the language used for WWW documents, is useful when defining the format of the information, but is not sufficient for expressing the meaning of the information. Hence, it is essential to be able to interpret unstructured HTML documents written in natural language in order to extract information from them. Furthermore, ungrammatical sentences, non-unique ship names, variant spellings and misspellings are other challenges of automated information gathering. Yet another issue of information extraction is that the relevant

information found in Web pages is not always expressed in sentences. Instead, it can be found in tabular form. This makes applying classical natural language processing (NLP) methods more difficult. Even if the system performs the information extraction task with high accuracy, the extracted information needs additional processing. Furthermore, there are a multitude of date/time conventions and the existence of different time zones complicates matters. In addition, location identification and mapping locations into latitude/longitude coordinates are other important issues that need to be addressed.

The general flow of the SeaSpider is illustrated in Figure 1. The model consists of three main components that are responsible for performing five main tasks. The first component, the SeaSpider-Retrieval Engine, handles the information retrieval task. The second one, the SeaSpider-Extractor, is responsible for information extraction and storing integrated information into the SeaSpider Database. The last component, which is a user-tool for answering user queries, is called the SeaSpider-Client. In order to answer user queries in the most efficient way, the model uses continuous processes to store relevant vessel movement information into the database ahead of time. In other words, retrieved, extracted and processed information is inserted into the SeaSpider database before a user search has begun. The function of the client-side application, SeaSpider-Client, is sending user queries to the server and presenting the returned results to the user in an appropriate format. On the other side, the server continuously searches, extracts and stores relevant information on vessel movements.

2.1 SeaSpider- Retrieval Engine

The Retrieval Engine is the module that is responsible for searching the WWW for relevant information sources, fetching relevant web pages and passing them to the Extractor module. In our design, the Retrieval Engine uses Google for searching the WWW for relevant information sources. It prepares queries that include keywords from a predefined set. Afterwards, the prepared queries are sent to the search engine using Google Search API [5]. Finally, the returned results serve as input to the Extractor. In this way, it is not needed to develop and maintain a crawler that automatically searches a large number of information sources. In addition to using Google for searching for relevant information sources, SeaSpider is able to scan a list of user specified information sources regularly.

Before giving details concerning the Retrieval Engine, it is useful to give definitions of the different structures in the system and the relations between them:

Keyword: SeaSpider has a predefined set of keywords, which are used to construct queries. One keyword can be used in many queries.

Query: Queries consists of one or more keywords and are used to search Google. In Google, keyword order in queries is significant. Changing the keyword order in any given query can produce different results.

Result: Results are the URLs returned by the Google search engine in response to queries. Results can contain different information in different times according to update period of the page. One query can return many results.

Result Snapshot: A view of an actual web page associated with a given result at a given instant is called a result snapshot. Accordingly, one result can have many result snapshots.

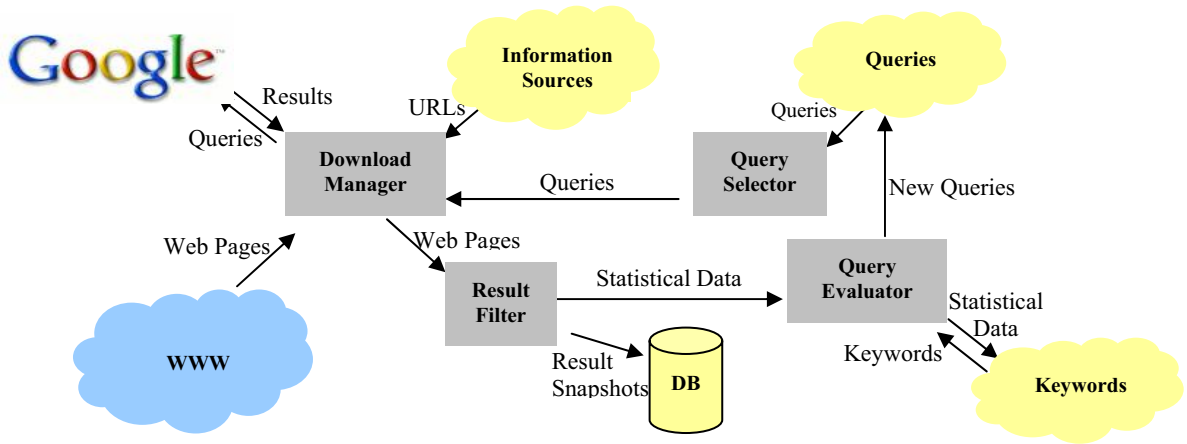

Fig. 2. The Retrieval Engine.

As illustrated in Figure 2, the Retrieval Engine has four main sub-modules: the Query Selector, the Download Manager, the Result Filter, and the Query Evaluator. The Query Selector is responsible for selecting the queries with high scores from the query pool and sending selected queries to the Download Manager. The Download Manager sends these queries to Google and gets result pages. After getting result pages, the Download Manager downloads both Google-returned results and user-specified information sources. Downloaded Web pages are sent to the Result Filter. The Result Filter processes downloaded pages and discriminates relevant pages from non-relevant ones according to their context. Result Snapshots belonging to relevant pages are stored into SeaSpider-Database by the Result Filter. After result filtering is finished, the Query Evaluator re-evaluates the queries and constructs new queries.

2.1.1 Query Selection & Query Evaluation

As shown in Figure 2, the Query selector selects a certain number of queries from a query pool. Each query is assigned a value, which indicates its relative relevancy factor. The Query selector simply selects the queries with high relevancy factors. The Query Evaluator is the sub-module, which assigns relevancy factors to the queries. Moreover, it creates new queries and adds them to the query pool by using statistical methods.

The query evaluator performs the query evaluation task in two steps: statistical data collection and query generation. In the first step, it assigns different relevancy factors to the structures described in section 2.1. According to the relevancy of their content, result snapshots are classified by the result filter as being either relevant or irrelevant. Ratio of relevant result snapshots to total result snapshots for any given result gives the relevancy factor of that result. In the same way, ratio of relevant result snapshots to total result snapshots for a query gives the relevancy factor of that query. Finally, ratio of relevant result snapshots to total result snapshots for a keyword gives the relevancy factor of that keyword.

The query evaluator assigns relevancy factors to the available queries in the system during statistical data collection. The other important task of the query evaluator in the system is query construction. SeaSpider uses bigram language model, a special case of N-gram which is used in various areas of statistical natural language processing, to construct new queries.

Statistical language models date back to Shannon's work on information theory [6]. One of the basic aims of the statistical language models is to predict the probability of the next word, given the previous word sequence: $P(w_n|w_1,...w_{n-1})$.

However, there is no easy way to compute the probability of a word given a long sequence of word history. It is not practical to keep the possibility of each word sequence, as this would require very large sample space. Moreover, when we consider that phrases or sentences have arbitrary lengths, it is not possible to observe most of the sequences during training of the language model. This will lead to data sparseness problem of over-fitting. For this reason, we group word sequences according to their last n-1 words to obtain an n-gram language model.

An n-gram of size 2 is called bigram. Given a bigram language model, it is straightforward to compute the probability of a word sequence as follows [7]:

$$P(w_1,w_2,w_3,...,w_n) = P(w_1)\,P(w_2|w_1)\,P(w_3|w_2) \ldots P(w_n|w_{n-1}) \quad (1)$$

For example, the probability of the query “Ship Arrival Departure” is computed as follows in our model:

$$P(\text{“Ship Arrival Departure”}) = P(\text{“Ship”})\,P(\text{“Arrival”} \mid \text{“Ship”})\,P(\text{“Departure”} \mid \text{“Arrival”}) \quad (2)$$

We approach the query construction problem as finding the most likely sequence of words. For finding the most likely keyword sequences, we use a dynamic programming technique: the Viterbi algorithm [8, 9]. Each keyword is treated as a state. Moreover, each state can only emit one value, which is the keyword value of that state. After applying the Viterbi algorithm to this model, a certain number of most likely keyword sequences are selected as new queries.

2.1.2 Page Downloading

A multithreaded download manager performs page downloading task. The Download Manager downloads result pages returned by the Google search engine in response to sent queries and information sources specified by the users. In order not to download pages that cannot be parsed by SeaSpider, the Download Manager simply eliminates non-HTML files by checking the file extensions and looking at the tag information.

2.1.3 Result Filtering

The objective of result filtering is to filter out irrelevant pages, which do not contain vessel movement information. Although the well-constructed queries return relevant pages most of the time, they can also return irrelevant ones. For this reason, the system has to process returned result pages and discriminate relevant pages from non-relevant ones.

Our observations showed that most of the vessel movement information is found in tabular form, usually embedded in itinerary tables that can be defined as an HTML table with a vessel column or date/time and port columns. Our approach is therefore to categorize documents into two groups: relevant documents which include an itinerary table and irrelevant documents which do not. Hence, the result filtering problem is reduced to itinerary table detection problem. Two sequential steps of table detection process, named entity recognition (NER) and column recognition, are the first two steps of the extraction process and the details of these tasks are given in sections 2.2.1 and 2.2.2.

2.2 SeaSpider- Extractor

After finding the relevant pages, the Retrieval Engine passes them to the Extractor, which is responsible for extracting relevant information from the filtered pages. The Extractor's task can consequently be seen as a slot filling problem, that is to say, filling the fields of a predefined target template. After the information is extracted and becomes ready for use, it is stored to the target database, a relational database with the tables of ships, ports, countries, cities and ship movements.

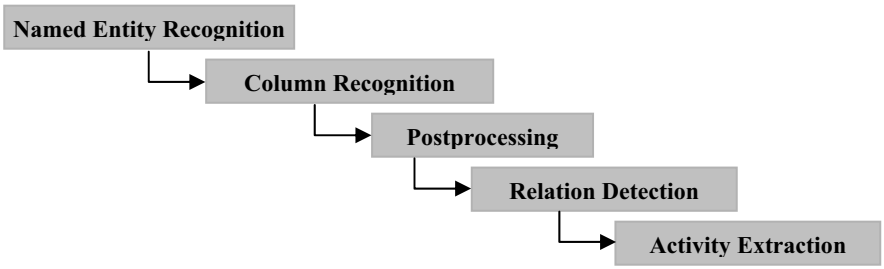

Fig. 3. General flow of the tasks performed by the Extractor.

The relevant information that SeaSpider needs is generally found in tabular form on the Internet. The problem with tabular representation is the lack of sentence structure. Therefore, some classical analysis methods used for natural language understanding are not very helpful in the case of tabular data. On the other hand, we can benefit from features specific to tabular data. HTML tags in source WEB pages can be utilized as clues to find hidden information, because they are more evident in tabular data given specific syntax features of HTML table structures.

Processing at this stage includes named entity recognition, column recognition, relation detection, date/time conversions, entity identification, and integration of the chunks of information coming from different sources. General flow of the tasks performed by the Extractor is shown in Figure 3.

2.2.1 Named Entity Recognition (NER)

SeaSpider starts the extraction process by recognizing named entities. Named entities are predefined entity categories such as the names of persons, organizations, locations and so on. The Named Entity Recognizer takes as input web documents in HTML and outputs identified named entities stored in an HTML element tree.

We used open-source GATE (General Architecture for Text Engineering) software [4] for NER. The extraction rules for the common named entities (e.g. person or location) are defined in the system. For this sub-module, we extended the rules and gazetteer lists defined in GATE to recognize fourteen extra named entity classes (ship, port and so on) specific to maritime domain.

SeaSpider also makes use of HTML tag hierarchy embedded in the document to extract target information. Each HTML element has a matching class in SeaSpider. The output of the named entity recognizer is a special tree structure which stores named entity information and the matched HTML tags while maintaining the HTML tag hierarchy. Although the NER phase is significant in the extraction process, it is not the decisive phase for labeling named entities, because a word or phrase in the document can correspond to more than one named entity classes. For instance, the word "Halifax" can be both a port name and a ship name. The Named Entity Recognizer does not try to resolve such kind of ambiguities at this level. Final class labeling is performed during post-processing.

2.2.2 Column Recognition & Post-Processing

It is essential to recognize relevant itinerary tables both for extracting vessel activity information from those tables and filtering out the irrelevant result pages. However, most tables are designed for human eyes and their layout and semantic meanings are not always well defined. A developer can design very complex tables that are made to be read by humans but not necessarily by computers. SeaSpider should therefore also have the capacity of reading tables of varying complexity.

SeaSpider performs table recognition process in two steps. The first step is called column recognition. The column recognizer, the sub-module responsible for column recognition, first reduces table complexity by converting complex tables to simple ones. After table complexity reduction, the system recognizes the column types of the table. We use majority voting for column type recognition task. Moreover, to label a column with a specific column type at least one header cell suitable to that specific column type must be found in the column. Headers and some unnecessary cell values (text elements labeled as "IgnoreElement" during NER) are not included in the voting process. An example of an itinerary table is shown in Figure 4. The first column is found to be of the type "ITINERARY_DAY". The second column is of the type "GENERAL_PORT". While analyzing the second column, the column recognizer does not take the cell values "At Sea" into account. Because such values were labeled as "IgnoreElement" during NER, these cells are not included in voting. In the same way, the cell values "---" in the third and fourth column are also not included in voting. The third column is labeled as an "ARRIVAL_TIME" column. The last column is of type "DEPARTURE_TIME".

DAY	PORT	ARRIVAL	DEPARTURE
1	Port Canaveral, Florida	---	4:00 p.m.
2	At Sea	---	---
3	Key West, Florida	7:00 a.m.	1:00 p.m.
4	Cozumel, Mexico	11:00 a.m.	7:00 p.m.
5	At Sea	---	---
6	Port Canaveral, Florida	7:00 a.m.	---

Fig. 4. An example itinerary table.

The main purpose of postprocessing, the second step in the table recognition process, is to tune up the column recognition task. In the design of the system, the output of a component is input to the next component in line. From this point of view, an incorrect recognition of a named entity during NER phase negatively affects the success of activity extraction task negatively. Postprocessing is the second and last chance to recover from the errors made at lower levels. For instance, the Named Entity Recognizer sub-module can sometimes fail to find all port names in the document. In such a situation, if the column recognizer marks a column as "GENERAL_PORT" and there is a cell value in the column whose named entity category is neither "PortName" nor "IgnoreElement", then that cell value is marked as "PortName" during postprocessing. Moreover, long sentences in itinerary tables (descriptions, notes etc.) are eliminated during postprocessing. After this, header cells and ignored cells are relabeled. The last step in the postprocessing task is to reassess the relevancy of the table. According to newly assigned labels, itinerary table candidates are reevaluated and tables marked as irrelevant are eliminated.

2.2.3 Relation Detection

Itinerary tables detected by the system do not always contain the necessary information to fill in the activity templates. For example, the itinerary table shown in Figure 4 does not have a vessel name column. Moreover, the first column in the table has to be mapped to date values. Information distributed to multiple columns is another issue to be addressed. The Relation Detector where the raw information chunks are mapped and integrated handles such issues. Furthermore, the processing performed in this module includes date/time conversions, and the integration of chunks of information from different sources.

Relation detection is the most crucial phase in the extraction process as it shapes the future course of the process. The Relation Detector takes as input itinerary tables with column types labeled and outputs field sets. As stated above, we treat vessel activity extraction problem as a slot filling problem; filling the fields of a predefined target template. The actual field filling process is performed at this level in which field sets are constructed. What is done at higher levels is the classification of the field sets and extraction of the activities.

Missing field search is the most important activity performed by the Relation Detector. For further processing a field set must contain three compulsory fields: a date field, a port field, and a ship field. If the system detects that any of these

fields are missing, it performs a missing field search. The Relation Detector also handles itinerary day mapping that is the mapping of date references to specific date values. We used different approaches for different tasks performed during the relation detection phase. We followed a rule-based approach for itinerary day mapping. Moreover, for searching missing date and port fields, best results are obtained by using rule-based fields. On the other hand, we used a statistical approach to search for missing ship names. Probability of being a ship name, distance from a ship header, and distance from the itinerary table are the features used while searching for missing ship names. We defined column combination rules to find related fields and combine them to form new columns. The last and the decisive step in the relation detection process is the rule-based conversion operation where columns are converted to the fields according to the conversion rules defined in the system.

2.2.4 Activity Extraction

During the activity extraction, the last phase of the extraction process, the input field sets are processed and converted to three types of vessel activities: arrival, departure, and uncategorized. For rule-based conversion, input field sets must contain a Ship field, a Port field and a DateTime field. As a result of applying activity extraction rules, the Activity Extractor sub-module outputs classified activities and inserts extracted activities into the SeaSpider database.

2.3 SeaSpider- Client

SeaSpider-Client is a client-side application, which sends user queries to the database and returns results to the user in an appropriate format, as illustrated in Figure 1. The principal function of the application is to query the SeaSpider database for vessel activities. Moreover, data can be exported into Google Earth thus enabling users to explore vessel activities with its help. Another function supported in the client application is to access the result snapshots stored in the SeaSpider database, the ability of accessing the source of each individual vessel.

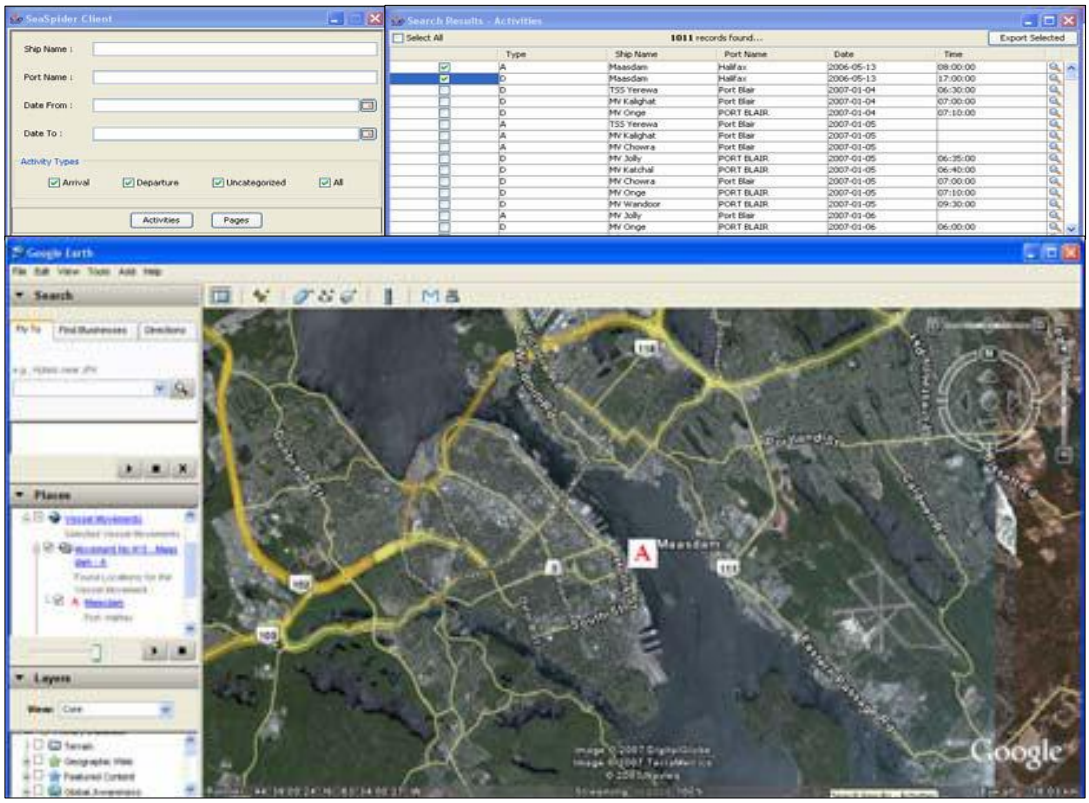

Fig. 5. A snapshot of SeaSpider-Client.

A snapshot of the SeaSpider-Client is shown in Figure 5. The query form appears in the top-left corner of the figure. A user can restrict his/her search by entering/changing the values of the fields found in the form. The search result window, where search results appropriate to the user query are listed, is located on the top-right corner of Figure 5. Vessel activity results can then be exported into Google Earth, the result of which is illustrated in the picture portion of Figure 5.

3. EXPERIMENTAL EVALUATION

3.1 Example Corpus

In this section, we present how our model performs different tasks in terms of precision and recall. We begin this section by explaining the corpus used in our experiments. In order to conduct experiments, we built a corpus containing 496 randomly selected documents. 133 documents in the corpus are relevant and the other 383 documents are irrelevant. The total number of tables in the documents is 2097. 1860 of these tables are irrelevant and 237 tables contain ship activity information. There are 384 missing fields (port names, ship names, date/time values) to find in the document set. Out of 7388 activities in the corpus, 3588 are classified as arrival, 3243 are classified as departure and 557 are classified as uncategorized.

3.2 Methodology

Several experiments were performed to analyze the success of the SeaSpider in performing different tasks: result filtering, table recognition, missing field search and activity extraction. During the experiments, the inputs to the system were the individual documents of the constructed corpus. At each processing step, the performance of the system was measured and reported.

We measured precision, recall, and F-measure, as is commonly done in the Message Understanding Conference (MUC) evaluations [10, 11, 12, 13, 14]. Precision is the fraction of correct outcomes divided by the number of all outcomes. For instance, the precision value for the activity extraction task is the percentage of extracted activities that are correct. On the other hand, recall is analogous to sensitivity in binary classification. Recall can be defined as the fraction of correct outcomes divided by the total number of possible correct answers. The F-measure, harmonic mean of precision and recall, provides a method for combining precision and recall scores into a single value.

For the two tasks related to information extraction, missing field search and activity extraction, we used the most conservative approach to determine the truth-value of the matching in which target sequences should be exactly matched. For example if ship name is “Seven Seas Mariner” and the extracted value is “Seas Mariner”, we do not count this extraction as a correct match.

3.3 Results

In this section, the quantitative results of the experiments are reported. The calculated precision and recall scores for the tasks show how successful our approach has been. In Table.1, the first row shows how our result filter classifies relevant and irrelevant web documents retrieved by the download manager. In the second row, the combined results of named entity recognition and column recognition tasks are presented. As has been previously explained, after these two serial processes are completed, the system decides whether or not a table is an itinerary table. Precision and recall values for missing field search task are listed in the third row. These results show the performance of the system in finding different fields that are located outside itinerary tables (date expressions, port names, vessel names and so on). Finally, results of the activity extraction task show the general extraction performance of the SeaSpider system.

Table. 1. Precision/Recall Values for different tasks performed by SeaSpider.

Task	Precision	Recall	F-value
Result Filtering	92.30%	95.57%	93.90%
Table Recognition	99.56%	96.62%	98.06%
Missing Field Search	70.18%	60.67%	65.08%
Activity Extraction	94.69%	66.52%	78.15%

3.4 Discussion

As can be seen from Table.1, the success rates of the result filtering and the table recognition tasks are quite satisfactory. On the other hand, it is obvious that the missing field search algorithms need improvement, though satisfying results have been obtained for many of the implemented methods. Another observation arising from the results is that the general flow of the system makes components interdependent. Since the output of a component is input to the next component, the success of one has an impact on the success of the next. This also means that errors can propagate from one module to another. This is in fact the main reason why performance of the activity extraction task was low compared to those of result filtering and table recognition tasks.

4. CONCLUSION & FUTURE WORK

This paper introduces the SeaSpider project, which aims to enable automated gathering of public information about vessel movements. The SeaSpider project has identified a successful methodology for automated search of unstructured information on web pages; this methodology could be applied to other domains of interest. SeaSpider is intended primarily as an operator aid to assist conventional systems and applications. Furthermore, the project is an attempt to use information sources found on the Internet in order to improve MISR capability. The study shows that public information located on the WWW carries significant potential for Maritime Domain Awareness. SeaSpider is thus an important landmark, being one of the first projects of its kind in the field.

In the frame of this study, a prototype system was implemented. Different rule-based and statistical methods were explored and satisfying results obtained for many of the implemented methods. Although SeaSpider combines several research fields to collect information automatically from the Web, it is mainly focused on information extraction. While the existing application areas for IE are broad and varied, our belief is that SeaSpider is an addition to the field.

Future work would include further refinements of the hand-crafted rules for expanding named entity boundaries and improvements of the generalization capability. Moreover, the system needs a rule engine to ease the task of rule definition and interpretation. Overall performance achieved for the system is quite satisfying. However, results also show that a better performance could be achieved by improving missing field search algorithms. The system uses keywords from a predefined set of keywords to construct new queries. The keyword set is however static in the current design. We believe that better queries could be obtained during the retrieval process by addition of a keyword extraction facility, which can update the keyword set according to result filter statistics, to the system.

REFERENCES

[1] Government of Canada Privy Council Office, “Securing an Open Society: Canada's National Security Policy”, http://www.pco-bcp.gc.ca/docs/InformationResources/Publications/NatSecurnat/natsecurnat_e.pdf (2004).

[2] Directorate of Maritime Strategy-Canada, “Securing Canada’s Ocean Frontiers: Charting the Course from Leadmark”, http://www.navy.forces.gc.ca/mspa_video-media/Securing_Canada_E.pdf (2005).

[3] Hammond, T.R. and Kessel, R.T., “The implications of the universal shipborne automatic identification system (AIS) for maritime intelligence, surveillance and reconnaissance”, DRDC Atlantic TM 2003-143, Defence R&D Canada – Atlantic (2003).

[4] Cunningham H., Maynard D., Bontcheva K., Tablan V., “GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications”, Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02), 168-175 (2002).

[5] Google SOAP Search API, http://code.google.com/apis/soapsearch.

[6] Shannon, C.E., “A Mathematical Theory of Communication”, Bell System Technical Journal, 27, 379-423 and 623-656. (1948)

[7] Jurafsky, D. and Martin, J.H., [Speech and Language Processing], Prentice Hall, Upper Saddle River, New Jersey, 191-235 (2000).

[8] Viterbi A.J., “Error bounds for convolutional codes and an asymptotically optimum decoding algorithm”, IEEE Transactions on Information Theory 13(2), 260–269 (1967).

[9] Rabiner L. R., “A tutorial on hidden Markov models and selected applications in speech recognition”, Proceedings of the IEEE 77(2), 257–286 (1989).

[10] Proceedings of the Third Message Understanding Conference (MUC-3), Morgan Kaufmann, (1991).

[11] Proceedings of the Fourth Message Understanding Conference (MUC-4), Morgan Kaufmann, (1992).

[12] Proceedings of the Fifth Message Understanding Conference (MUC-5), Morgan Kaufmann, (1993).

[13] Proceedings of the Sixth Message Understanding Conference (MUC-6), Morgan Kaufmann, (1995).

[14] Proceedings of the Seventh Message Understanding Conference (MUC-7), (1998).