# The Design of PoliDocs: a Web Information System for the Disclosure of Dutch Parliamentary Publications

Tim Gielissen and Maarten Marx

ISLA, University of Amsterdam, Kruislaan 403,
1098 SJ Amsterdam, the Netherlands
Tim.Gielissen@student.uva.nl
MaartenMarx@uva.nl

**Abstract.** The development of PoliDocs.nl, a Web Information System for the disclosure of Dutch parliamentary publications, is an effort to improve the disclosure of parliamentary publications in The Netherlands. The data is distributed over three sources and is available through different Web Information Systems that need improvement. This paper explains how we translated our knowledge about the current situation, about the requirements put forth by the collective weblog Sargasso, and about the wishes and needs of prospective users/professionals into a functional design. The functional design describes the back-end, which relies heavily on the Extract-Transfer-Load (ETL) process to harvest, integrate, transform and store the data. The functional design also describes the front-end website, which uses state-of-the-art techniques to meet the requirements. The functional design meets all the Sargasso requirements except one for which the information is not available.

**Keywords:** parliamentary data, functional design, faceted search, result aggregation/summarization

## 1 Introduction

A good information flow between a government and its people is a cornerstone of a well functioning democracy. This information flow has two directions, from the government to its people and vice versa. Each direction poses its challenges, mostly related to scale. Nowadays, the information flow from the government to its people runs primarily through news media. The advantage of this is that news media reach a lot of people. The disadvantage is that coverage by news media is always incomplete and edited, i.e., only information with news value is presented.

Luckily, there are alternatives. Most democratic countries document the actions of the members of their government and make these documents available to the public. This is also the case in the Netherlands, where all parliamentary papers from 1814 up until 'yesterday' are available.[1] These documents are impartial and complete. People

[1] At the time of writing, documents from 1974 onwards are available in digital form, everything before that time is available on paper and will be available in digital form in 2010.

who want to know more than the political events that are covered by the news media, can turn to these documents.

Currently, the parliamentary data in the Netherlands is distributed over three different Web Information Systems (WIS). This is because the data is partitioned in 1) preliminary, 2) definitive and digital (after 1995), and 3) scanned and OCRred data (1814-1995). This situation leaves much to be desired: first of all, the distribution of data over three sources is inconvenient. Furthermore, none of the websites offer search results ranked by relevance. All sites retrieve *documents* (one document is the meeting notes of one day, typically 80 pages 2 column PDF). Further search must be done in a PDF reader. Also, the most important site 'Parlando'[2], which offers the definitive and digital most recent data, lacks even the 'most basic functionality' (for example, linking to a document) according to the politically oriented, collective weblog 'Sargasso'[3]. In 2005, members of this weblog wrote a letter to the the Dutch lower house specifying 15 requirements that a new system/website should meet [1]. Now, more than three years later, the situation has not changed.

In other countries, the situation varies. An example of a good WIS for the disclosure of political data is the British TheyWorkForYou.com[4]. This WIS (probably unknowingly) meets most of the Sargasso requirements and reached 41st in a list of the 101 most useful websites, by the British newspaper 'The Daily Telegraph' [2]. However, this WIS only discloses recent publications that are originally available in an XML format. This differs from the Dutch situation where only PDF documents are available, including the legacy data with OCR errors.

Inspired by the actions of Sargasso and knowledge of the current situation and better initiatives abroad, a group of researchers and students of the University of Amsterdam started to develop a new WIS: 'PoliDocs'. This system transforms the PDF documents into an uniform XML format and makes them available in a new WIS. Over time, PoliDocs will meet the Sargasso requirements and should be as good as initiatives from abroad, or even better. In this way, it will improve the situation in the Netherlands. This paper is an experience report on the process of designing the WIS and shows how the requirements from Sargasso, the knowledge of the current situation and examples from abroad are translated into a functional design that incorporates state-of-the-art web technology.

The remainder of this paper is organized as follows: in section 2 the current situation is described in more detail. Section 3 covers the translation from requirements and foreign examples into a functional design. Section 4 describes the functional design. Section 5 links the functional design back to the requirements formulated by Sargasso. Finally, section 6 will conclude this paper with a short summary and conclusion.

[2] http://parlando.sdu.nl/cgi/login/anonymous (Dutch)
[3] http://sargasso.nl/ (Dutch)
[4] http://www.theyworkforyou.com

## 2 Background

The functionality of PoliDocs is based on, and inspired by multiple sources. The first is a set of user requirements set forth by the politically oriented, collective weblog Sargasso. The second source is a foreign example of a WIS to disclose parliamentary information: the British TheyWorkForYou.com. These sources are described in this section. Other sources include knowledge about the current situation and knowledge from conversations with prospective users/professionals. This conversations focussed more on content and quality standards than on functionality. Also, this knowledge is less formal and less structured and therefore not described here.

### 2.1 User Requirements by Sargasso

Table 1 contains a translation of the user requirements as formulated by Sargasso in 2005. We believe these requirements are relevant because they originate from end-users who are experienced with the current system, who are motivated to help improve the current system and who have diverse backgrounds.

**Table 1.** User Requirements by Sargasso

| # | Requirement |
|---|---|
| 1 | All 'kamerstukken' (parliamentary papers) should have a direct, stable link (URL); |
| 2 | Users should be able to retrieve all parliamentary papers that belong together - i.e., dossiers - in one request, including earlier versions of 'kamerstukken'; |
| 3 | Recordings of proceedings (audio/video) should be coupled with the parliamentary papers; |
| 4 | For all the motions/proposals the result of the voting should be clear, directly; |
| 5 | The voting behavior of all parties should be clear; |
| 6 | The voting behavior of all members of the parliament should be clear; |
| 7 | Users should be able to request which motions and written questions specific members of the parliament handed in or co-signed; |
| 8 | For every parliamentary paper it should be clear when it was discussed or when it will be discussed; |
| 9 | Users should be able to request the parliamentary papers directly from the parliamentary agenda; |
| 10 | Users should be able to switch between basic information and more elaborate information; |
| 11 | Users should be able to display the results of a keyword search on a timeline where parliamentary papers that belong together are grouped; |
| 12 | There should be an open API with which other parties can use the available information in different ways; |
| 13 | There should be an option to subscribe to RSS feeds of publications within specific dossiers; |
| 14 | All data should be presented in an open, standard format. There should not be a necessity to use a product from one single supplier to see/use to data; |
| 15 | The website that gives access to the information online should pass the Dutch 'Drempels Weg' test;[1] |

[1] 'Drempels Weg' is a Dutch foundation that tests if websites are user-friendly

### 2.2 TheyWorkForYou.com

TheyWorkForYou.com is a British website for the disclosure of parliamentary publications. When TheyWorkForYou.com is compared to the Dutch Parlando, the following differences arise that are interesting for the functional design of PoliDocs:

- All parliamentary papers have permanent links (permalinks). It is even possible to link to a specific statement in a debate.
- The parliamentary papers are available in HTML and XML, as opposed to the PDF that Parlando offers.
- TheyWorkForYou.com offers many ways to search. Users can search for persons, keywords, keywords with extra options (advanced search) and for locations. Furthermore, users can see recent parliamentary papers and subscribe to different RSS feeds.
- (Most) results can be sorted according to date, person or relevance.
- When searching in the proceedings, users get statements as results, as opposed to entire documents that Parlando returns.
- TheyWorkForYou.com offers an API so that professional users can use the system.
- Users can contribute to the website by adding comments or answering polls.
- The video is divided into "chapters" based on speaker-changes. These chapters are synced with the proceedings.

Because of these characteristics, TheyWorkForYou.com (probably unknowingly) meets quite a few of user requirements formulated by Sargasso. They meet all requirements except: 2 (retrieve documents by dossiers), 9 (no agenda's available), 11 (display results on timeline), and 15 (it is not clear if the website passed a usability test).

## 3 Methodology

The work for the PoliDocs project is divided in four phases: Requirements analysis, Design, Implementation and Maintenance. Each phase has moments for evaluation. There are different perspectives on the relation between these phases [3]. In the PoliDocs project, the Sashimi model was used [4]. In the Sashimi life cycle of software development, the different phases overlap, but do not get mixed up. In our experience, it is not desirable to let the different phases linger throughout the project, because of the implications for the following phases.

The first two phases, requirements analysis and design, are the topic of this paper. The requirements analysis consists of identifying user needs and specifying goals of the project. This meant comparing the different Dutch Web Information Systems and those from abroad along the same dimensions. Furthermore, we made contact with potential users and professionals, for example the ICT-department of the parliament, members of the Dutch Royal Library, political scientists and other scientists. They explained what they believe is important, for example retaining the high quality of the data during the transformation from PDF to XML. Finally, the Sargasso requirements

were included in the requirement analysis as well. On the basis of these sources, we formulated requirements and goals of the system.

After the requirement analysis, the design phase for a new Web Information System began. In this phase, the question of how to meet the requirements is answered. This meant comparing (and testing) different technical solutions to meet the requirements. We incorporated the latest technical insights in our system that we gained from visiting conferences, reading literature, and from fellow scientists.

The first two phases of the software development cycle result in a functional design. The functional design is the answer to the question how to meet the requirements. The functional design is described in the following section. It shows how the requirements, distilled from the user requirement specified by Sargasso, knowledge of the current situation, and contact with users/professionals, are translated into a design. Section 5 will evaluate the functional design by linking it back to the Sargasso requirements.

## 4 Functional Design

The PoliDocs Web Information System is composed of two parts: the back-end system and the front-end website. The functional design of both will be covered in this section.

### 4.1 PoliDocs back-end system

The Dutch parliamentary data is only available in PDF. This is not a desirable format to work with because we want *information* retrieval and not *document* retrieval. Therefore, we decided to develop an XML-based system to ensure easy navigation within the data corpus. TheyWorkForYou.com shows some of the advantages of using XML, for example the ability to point users to exactly the right spot in a document: 'entry point retrieval' [5]. Also, PDF documents are flat and meaningless for the machines. XML allows us to add semantic information to the documents (e.g., this is a person or this is a party) and to make implicit structure explicit digitally.

So the task at hand is to harvest the PDF documents from different sources, integrate this data, convert them to a uniform XML format and to store them in an XML database system. We structure this using the Extract-Transfer-Load (ETL) process [6]. In the extraction phase, we harvest the data from different sources and map the data structure. In the transformation phase, we transform the PDF documents into XML documents according to our DTD. In the loading phase, the database is created and filled.

The transformation part of this process is the most challenging. The transformation of the PDF documents from different sources into uniform XML documents involves a number of steps: First the text layer needs to be extracted from the PDF document. This text layer needs to be cleaned up, especially when it contains OCR errors. In this text, we will place markers to mark syntactic (e.g., white lines, column start) and semantic elements (e.g., this is a person, this is the start of a topic). These markers will be placed using layout information, like the position text elements on the page,

and using regular expressions that match the text itself. This approach has proved to work when building our prototype because of the highly standardized, uniform format of the parliamentary publications. When the markers are set, we use them to create an XML document with a deep structure using the XSLT 2.0 command: xsl:for-each-group. This way, we nest the different elements in a document according to our DTD. More about the DTD and the technical approach in [7].

### 4.2 PoliDocs front-end website

A user of PoliDocs.nl should be able to find information in the Parliamentary data easy and fast. To accomplish this, the website should meet the following requirements:

- Parliamentary data from different sources has to be combined seamlessly so it can be presented in a uniform way;
- All the information that is available in the data should be available and easily accessible on the website;
- Users should be provided with different methods to search in the data;
- The focus should be on information retrieval and not document retrieval;
- Everything should be presented in a clear manner;
- Professional users should be able to use the system without unnecessary limitations;
- The website has to be up-to-date and users should be able to request to be updated about new parliamentary information on the website (using RSS feeds);

To meet these requirements, the website is composed of three parts: faceted search, results aggregation/summarization, and a professional user interface. These parts will incorporate state-of-the-art web technologies.

#### 4.2.1 Faceted search

First of all, users must be able to search in the data. We want to use faceted navigation because of its power to support exploration and discovery within an information collection [8]. When using faceted search, all information objects can be classified in different categories. This can add to the power of a keyword search functionality. It allows for autocompletion for example. Thus, users of PoliDocs should be able to:

- Search using keywords;
- Put extra restriction on the available categories;
- Get suggestions for completions when typing a query (auto-completion);

But faceted search does not end after the query is executed. The results can be displayed in a list, but information about the categories of the result can be shown as well. This allows for a fluid switching between refining and expanding the data. Furthermore, the keyword search terms should affect the facet label ordering [8]. This turns the facets into a simple analysis tool and summarizes the data. The current prototype of PoliDocs contains this kind of faceted search, as illustrated in figure 1.


**Figure 1.** Faceted search in PoliDocs

Figure 1 shows our 'top 5 lists' after a search query for 'euthanasie' (euthanasia). From left to right, the five most occurring persons, parties and years are displayed. Note that the keyword search term influences the order of the labels, as they are ordered by quantity. The 'top 5 lists' can also be used for navigational purposes.

For the final version of PoliDocs, we want this faceted search to be expanded. Especially, we want to use more categories. For example, we could also show top 5 lists of most occurring head words. The most important category that has to be added is the use of 'dossiers'. Euthanasia has its own dossier, but if you search for 'airport' for example, there are results from all kinds of dossiers. This is also part of the Sargasso requirements (specifically requirements 2 and 11).

#### 4.2.2 Result aggregation/summarization

Faceted search is great if you want to find something rather specific in the data. If you just want to explore or discover the data, result aggregation/summarization is more suited [9]. Where faceted search takes users into the depth of the XML data, result aggregation/summarization shows the breadth of the data by aggregating data or by summarizing it.

We want to incorporate three groups of results aggregation/summarization: per person, per party and per time period. When searching for a person, party or period, the system should return a sort of 'dynamic biography'. For a person for example, the most recent parliamentary papers should be displayed and more static information like their name and picture. The aggregation and summarization will manifest in tables, graphs and tagclouds. A few examples: tagclouds could show linguistic usage of a person. Graphs could show the number of written questions on a timeline. Figures could show who 'attacks' who in a debate. For more examples, see [7].

PoliDocs should include result aggregation/summarization per person, party or time period. All tagcloud, graphs and tables have to be interactive, so users can jump to certain documents by clicking in a tagcloud or graph. Result aggregation/summarization can be done relatively easy with a XML collection.

#### 4.2.3 Professional user interface

PoliDocs should offer an interface for professional users so they can query the database without restrictions. Here, professional users can query the PoliDocs database directly using the powerful XQuery and Narrowed Extended XPath I (NEXI) [5].

## 5 Evaluation

In Table 2 we list the user requirements specified by Sargasso in abbreviated form and indicate how they are met by our functional design. Note that some requirements are rather specific, while we did not discuss the specifics of our functional design here.

**Table 2.** Sargasso requirements and PoliDocs

| # | Requirement in short | How the demand will be met |
|---|---|---|
| 1 | Permalinks for all parliamentary papers | All PDF and XML documents, and all webpages will have permalinks. Also, every statement will have a permalink |
| 2 | Retrieve dossiers in one request (including earlier versions) | Users will be able to search for dossiers and these will include older documents when available |
| 3 | Audio/Video coupled to proceedings | Another parallel project will make this possible for us (the project is called "OpenKamer") |
| 4 | Voting per proposal | This information will be displayed along with other information about proposals |
| 5 | Voting per party | This information will be displayed on the party overview page |
| 6 | Voting per person | This information is **not** available, so we cannot meet this demand |
| 7 | Find proposal/question by person who handed it in or co-signed | This will be possible using advanced search |
| 8 | It should be clear when parliamentary paper were/will be discussed | The date on which parliamentary papers were discussed will be displayed along with the other information and will also be included in result lists. When parliamentary papers will be discussed will be visible in the agenda |
| 9 | Request documents from agenda | The user will be able to request document from the agenda's |
| 10 | Switch between basic/advanced information | There will be multiple switching options. For instance, the difference between basic/advanced search, or between PoliDocs.nl and the professional user interface |
| 11 | Results displayed on a timeline | The results will be displayed on a timeline when requested, also the result aggregation/summarization contains multiple timelines |
| 12 | API for other parties | We will offer the professional user interface with which users can query the database with a powerful query language |
| 13 | RSS | PoliDocs will offers multiple RSS to subscribe for, including RSS per dossier |
| 14 | All data in open standard format | PoliDocs will use a custom XML format and will conform to web standards |
| 15 | Pass usability test | PoliDocs will be put through a usability test when it is done and will be adjusted if necessary |

Table 2 shows that the final version of PoliDocs will meet all Sargasso requirements except providing information on the voting results per person, because the information is not available. Although it is not our main goal to meet the user requirements by Sargasso, it serves as a good checklist.

Furthermore, we evaluated our functional design and its approach using rapid prototyping. Three small groups of Bachelor Information Science students tried the approach on three different sets of parliamentary data: the German Bundestag, the Flemish Parliament, and the Belgium federal parliament. Within 3 weeks, two of the three groups succeeded very well in automatically transforming PDF documents in uniform XML documents and loading them into a XML database.

## 6 Conclusion

This paper described the situation of the disclosure of parliamentary publications in the Netherlands. The data is distributed over different Web Information Systems that do not compare to some similar systems from abroad. With the development of PoliDocs, we try to improve this situation. This paper explained how we translated our knowledge about the current situation, about the user requirements specified by Sargasso, about examples from abroad and about the wishes and needs of prospective users/professionals into a functional design. The functional design describes the back-end, which relies heavily on the ETL process to harvest, integrate, transform and store the data. The functional design also describes the front-end website, which uses state-of-the-art techniques to meet the requirements. These techniques include faceted search with autocompletion and adaptive facet labels, result aggregation/summarization using interactive tables, tagclouds and graphs, and a professional user interface using XQuery and NEXI. The functional design meets all the Sargasso requirements except one for which the information is not available.

The functional design will guide the implementation of PoliDocs.nl. The website will be finished in June 2009, but for now a prototype is available at:

http://www.polidocs.nl.

We envision that a new, well-functioning and fun Web Information System for the disclosure of parliamentary data will improve the information flow between the Dutch government and its people.

## References

1. Actie Open Democratie 1.0 - De brief. Retrieved June 15th, 2008, from Sargasso.nl: http://sargasso.nl/archief/2005/11/17/actie-open-democratie-10-de-brief/ (2005)
2. The 101 Most Useful Websites. Retrieved February 2nd, 2009, from Telegraph.co.uk: http://www.telegraph.co.uk/scienceandtechnology/3356874/The-101-most-useful-websites.html (2008)
3. Raccoon, L.B.S.: The Chaos Model and the Chaos Life Cycle. Software Engineering Notes, 20(1):55-66 (1995)
4. DeGrace, P., Stahl, L. H.: Wicked Problems, Righteous Solutions: A Catalogue of Modern Software Engineering Paradigms. Englewood Cliffs, NJ: Yourdon Press (1990)

5. Sigurbjörnsson, B.: Focused Information Access using XML Element Retrieval. PhD thesis, University of Amsterdam (2006)
6. Rahm, E., Do, H. H.: Data Cleansing: Problems and Current Approaches. IEEE Data Engineering Bulletin, 23(4):3-13 (2000)
7. Gielissen, T., Marx, M.: Exemelification of Parliamentary Debates. In: the 9th Dutch-Belgian Information Retrieval Workshop, pp. 19-25. Centre for Telematics and Information Technology, Enschede (2009)
8. Hearst, M. A.: UIs for Faceted Navigation: Recent Advances and Remaining Open Problems. In: Workshop on Computer Interaction and Information Retrieval, pp. 13-17. Microsoft (2008)
9. Murdock, V., Lalmas, M.: Workshop on Aggregated Search. SIGIR Forum December 2008, 42(2):80-83 (2008)