­­­

Thywin – Design Document

Geschreven door: Bobby Bouwmann, Thomas Gerritsen, Thomas Kooi, Erwin Janssen, Imre Woudstra  
Datum: 14-5-2014  
Project begeleider: Alice de Groot  
Opdrachtgever: Joost Kraaijeveld

# Table of contents

[Table of contents 2](#_Toc389480642)

[Use Cases 3](#_Toc389480643)

[Use Case 1: Search 3](#_Toc389480644)

[Activity diagram 3](#_Toc389480645)

[Sequence diagrams 3](#_Toc389480646)

[Use Case 2: Crawler 4](#_Toc389480647)

[Activity diagram 4](#_Toc389480648)

[Sequence Diagram 5](#_Toc389480649)

[Use Case 3: Parsen 6](#_Toc389480650)

[Activity diagram 6](#_Toc389480651)

[Sequence diagrams 7](#_Toc389480652)

[Use Case 4: Master 8](#_Toc389480653)

[Get URI 8](#_Toc389480654)

[Put URI 8](#_Toc389480655)

[Get Document 9](#_Toc389480656)

[Put Document 9](#_Toc389480657)

[State diagram 10](#_Toc389480658)

[Domain Model 11](#_Toc389480659)

[Class Diagrams 12](#_Toc389480660)

[Master 12](#_Toc389480661)

[Parser 13](#_Toc389480662)

[Crawler 15](#_Toc389480663)

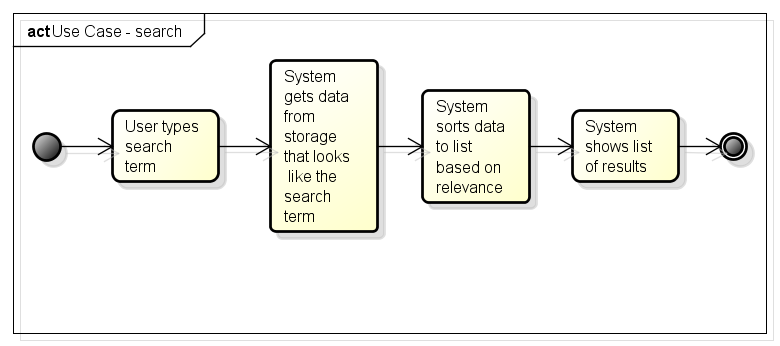
# Use Cases

## Use Case 1: Search

The user goes to <http://thywin.com>. The browser then shows a search page. The user types a search term in the search field. The server then shows all the results based on the search term.

|  |  |
| --- | --- |
| Primary Actor: User | |
| Stakeholders: Site owners | |
| Preconditions: The site is online and the database is available. | |
| Post conditions: The user gets results based on their searchterm. | |
| Main success scenario: | |
| 1. User goes to <http://thywin.com>. | 2. Webserver sends page back. |
| 3. User types search term in the search field. | 4. Webserver shows results. |
| Extensions: ( or Alternative flow) | |
|  | [If no results]  4a. Webserver shows message “0 results”. |

### Activity diagram



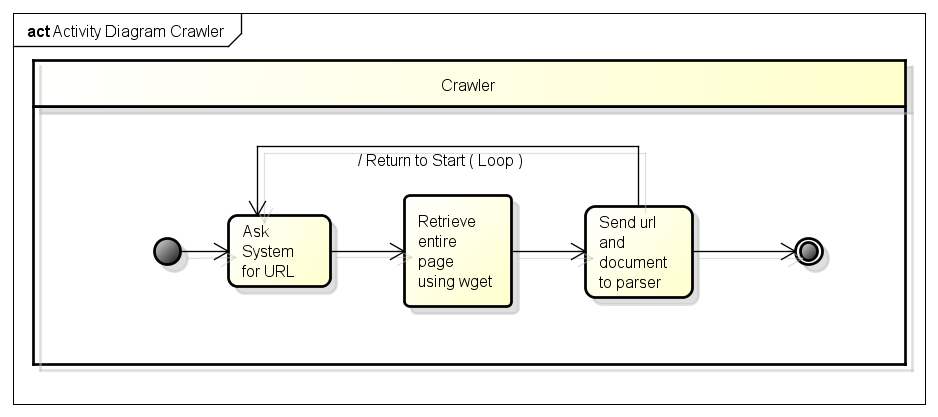
### Sequence diagrams

## Use Case 2: Crawler

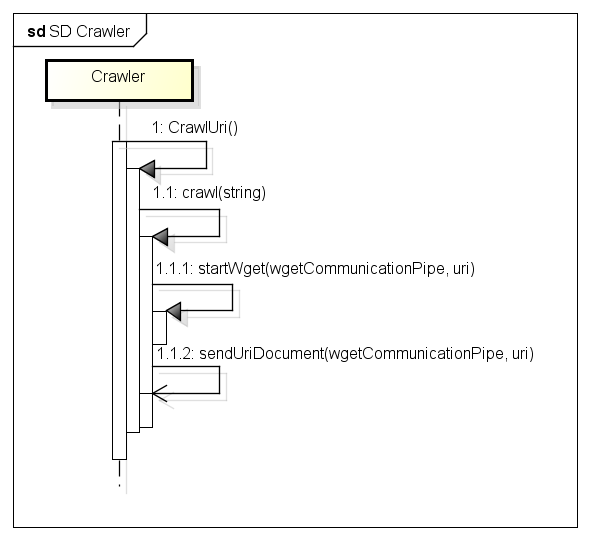
The crawler asks the master for an URI. The master then gets an URI from the URI queue en sends this to the crawler. The crawler then uses WGET to get the data from the URI. This data and the URI will be send back to the master.

|  |  |
| --- | --- |
| Primary actor: Crawler | |
| Stakeholders & interest: Scheduler, Parser | |
| Preconditions: Connection to the master, connection to the internet | |
| Main success Scenario | |
| 1. Crawler asks master for a URI to crawl. | 1. The master gets an URI from the URI queue. |
|  | 1. The master sends an URI to the crawler. |
| 1. Crawler gets the data from the URI using WGET. |  |
| 1. Crawler sends the URI and the crawldata to the master. |  |
| Extensions: ( or Alternative flow) | |
|  |  |

### Activity diagram



### Sequence Diagram

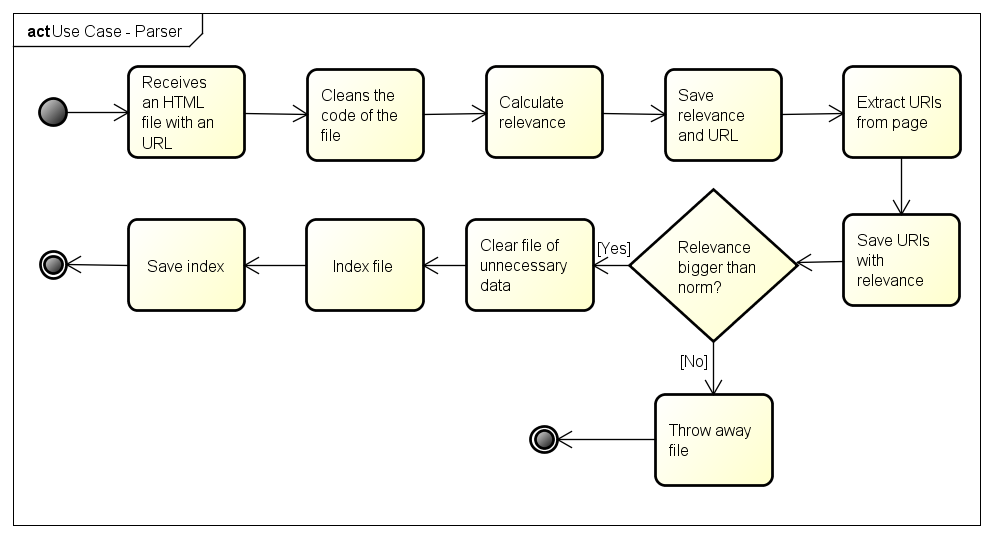


## Use Case 3: Parsen

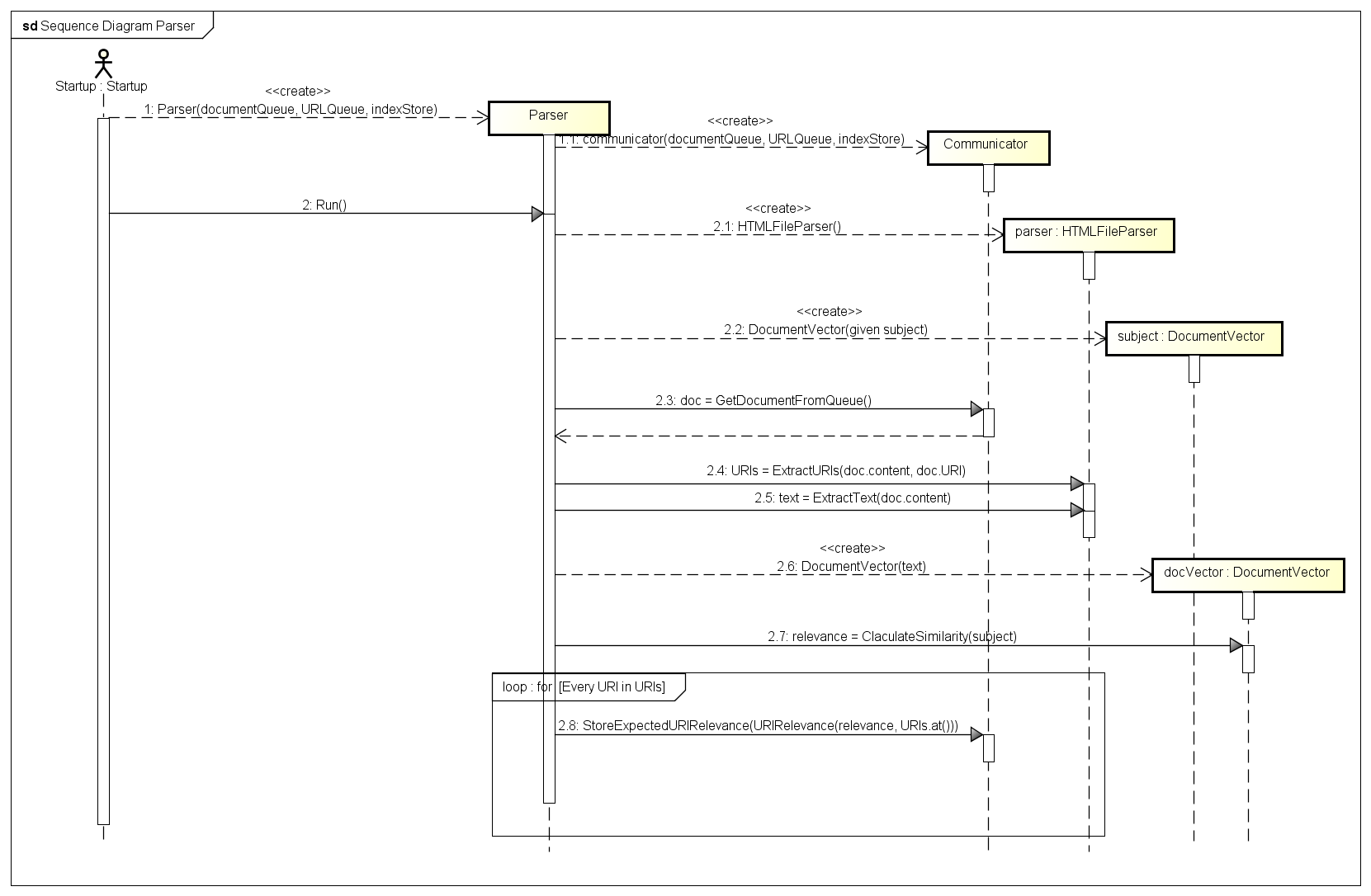
The parser requests an document and the related URI from the master. The parser then extracts all the URI’s from the document. The parser then removes the html tags. The parser is now ready to determine the relevance of the document. After this is done the parser sends the found URI’s to the master.

|  |  |
| --- | --- |
| Primary Actor: Parser | |
| Stakeholders: Master, Crawler, Database | |
| Preconditions: Connection to the database and the crawler, file is HTML format | |
| Post conditions: File is indexed and the index, URIs and relevance is stored | |
| Main success scenario: | |
| 1. The parser requests a document from the master. | 1. The master send the URI and document to the parser. |
| 1. The parser extracts the URI’s from the document. |  |
| 1. The parser removes the html tags. |  |
| 1. The parser determines the relevance of the document. |  |
| 1. The parser sends the URI’s with the relevance to the master. |  |
| Extensions: ( or Alternative flow) | |
|  |  |

### Activity diagram



### Sequence diagrams



## Use Case 4: Master

### Get URI

|  |  |
| --- | --- |
| Primary Actor: Crawler | |
| Stakeholders: Crawler | |
| Preconditions: System is in waiting state. | |
| Post conditions: De Crawler has an URI. The URI queue has an element less. | |
| Main success scenario: | |
| 1. Crawler asks for an URI. | 1. System gets the first element of the URI queue. |
|  | 1. System sends the URI to the crawler. |
|  | 1. Systeem removes the URI from the URI queue. |
| Extensions (Alternative flow) | |
|  | [No element in queue]  2. System gets an URI of the default Queue (start point). |

### Put URI

|  |  |
| --- | --- |
| Primary Actor: Parser | |
| Stakeholders: Parser | |
| Preconditions: System is in waiting state. | |
| Post conditions: The URI queue has one more element. | |
| Main success scenario: | |
| 1. Parser sends an URI to the system. | 1. System checks if the URI exists in the queue. |
|  | 1. [URI is Unique]   System put URI in the queue. |
| Extensions (Alternative flow) | |
|  | [URI already exists in URI queue]  3a. System does nothing with the URI. |

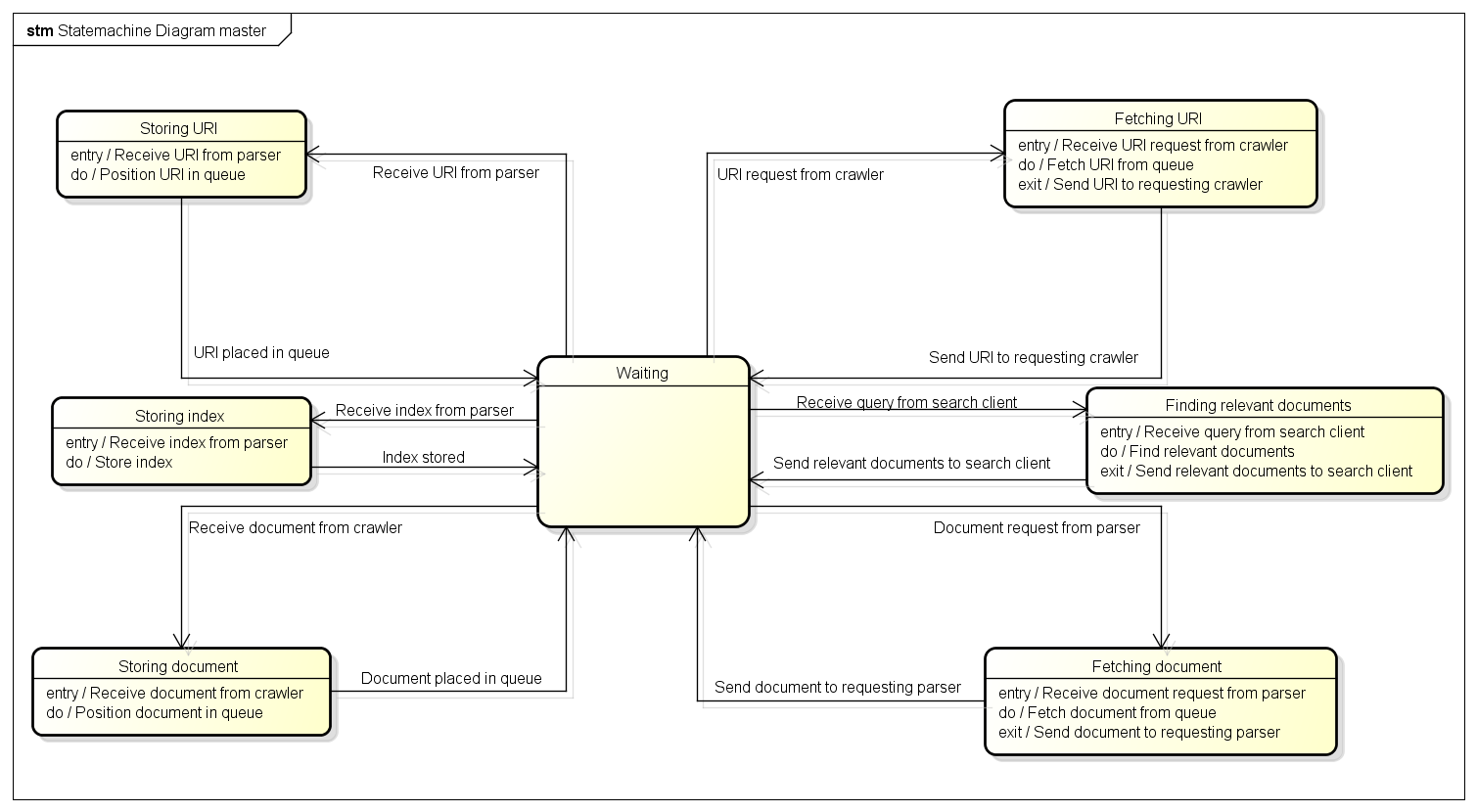
### Get Document

|  |  |
| --- | --- |
| Primary Actor: Parser | |
| Stakeholders: Parser | |
| Preconditions: System is in waiting state. | |
| Post conditions: The parser has a document from the document queue. The document queue has one element less. | |
| Main success scenario: | |
| 1. Parser asks for a document | 1. System gets the first document from the document queue. |
|  | 1. System sends the document to the parser. |
|  | 1. System removes the document from the document queue. |
| Extensions (Alternative flow) | |
|  | [No document in document queue]  2a. System waits until the document queue has a new document. |

### Put Document

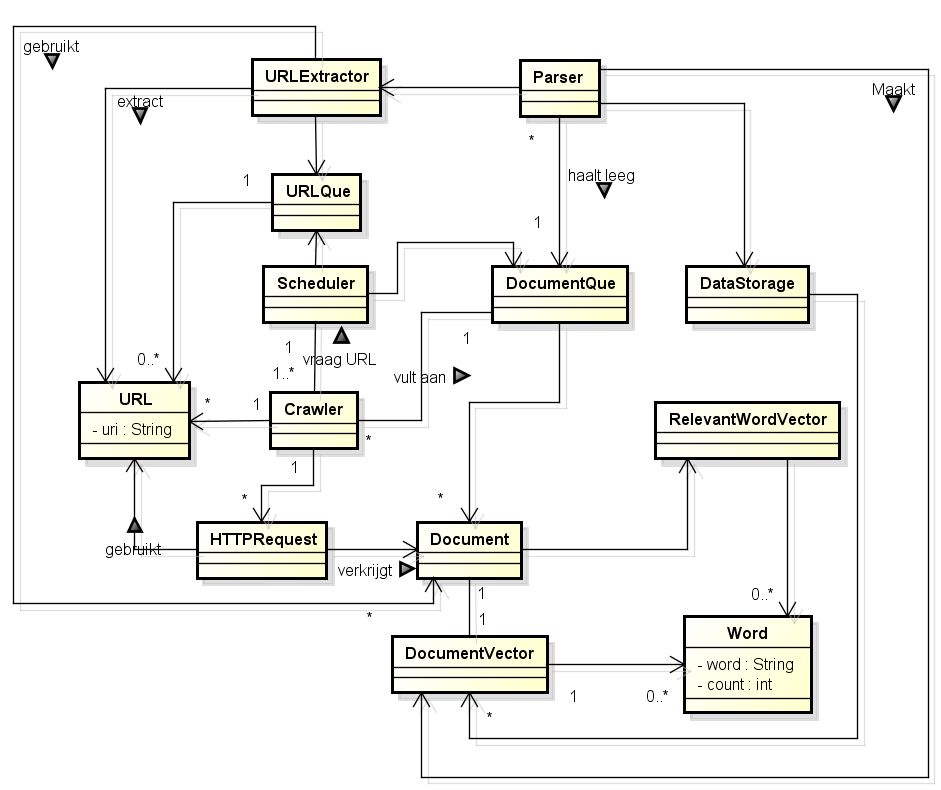
|  |  |
| --- | --- |
| Primary Actor: Crawler | |
| Stakeholders: Crawler | |
| Preconditions: System is in waiting state. | |
| Post conditions: The document queue has one more element. | |
| Main success scenario: | |
| 1. Crawler send a document to the system. | 1. System checks if the document exists in the document queue. |
|  | 1. [document is unique]   Systeem plaatst Document in de queue. |
| Extensions: ( or Alternative flow) | |
|  | [document already exists in document queue]  3a. System does nothing with the document. |

### State diagram



# Domain Model

*Versie 1.0:*



# Class Diagrams

Dit hoofdstuk beschrijft de class diagram ontwerpen voor de verschillende onderdelen van het Thywin Project. Wij hebben er voor gekozen om de verschillende taken binnen het algehele proces op te splitsen en te verdelen over andere processen die schaalbaar zijn.

Deze processen zijn:

* Master Proces
* Parser Proces
* Crawler Proces
* Search Engine Proces

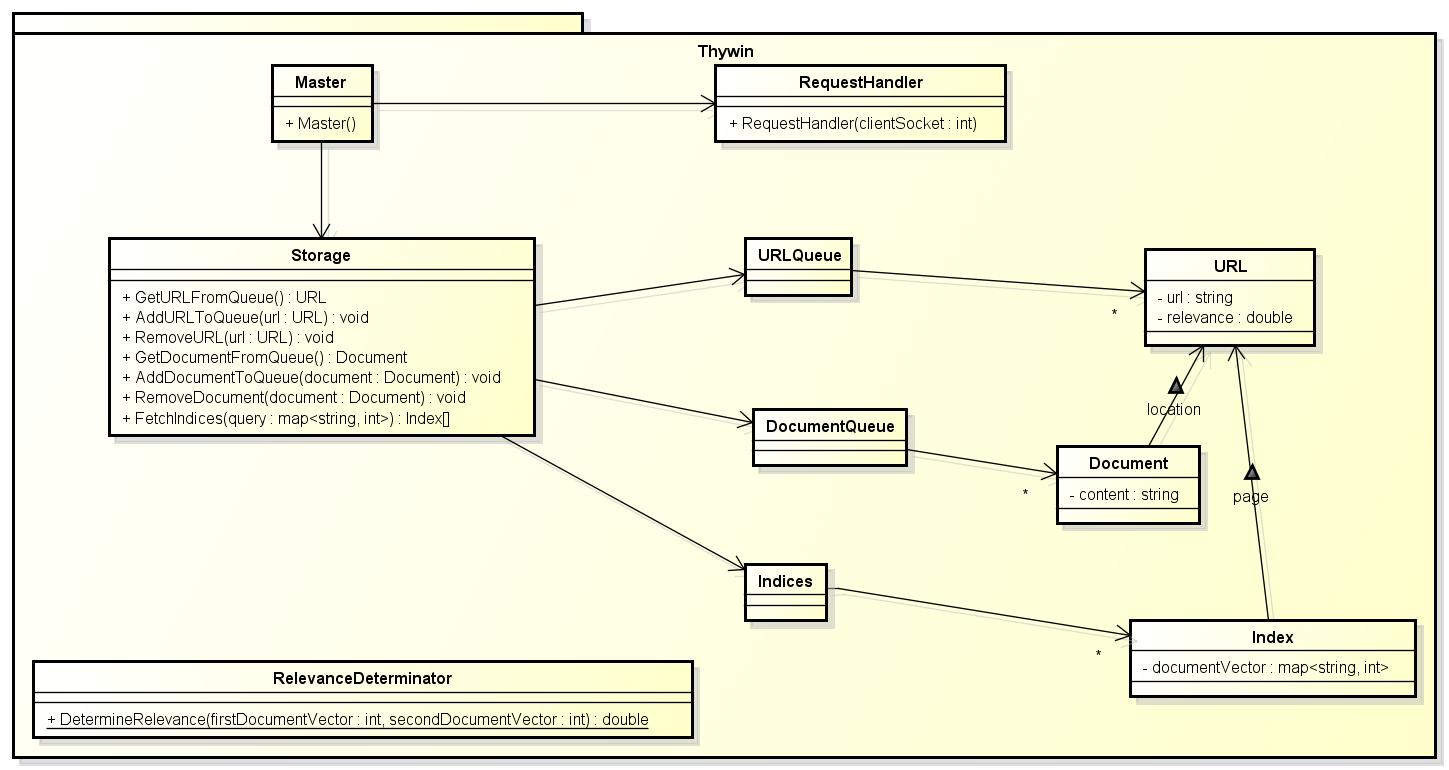
## Master

Het Master process heft als taak het bij houden van de queues en opslaan van informatie. Het zorgt er voor dat de juiste URIs worden door gegeven en dat er indexen worden opgeslagen in de database.

**Versie 1.1 (huidig):**

TBC

*Versie 1.0:*

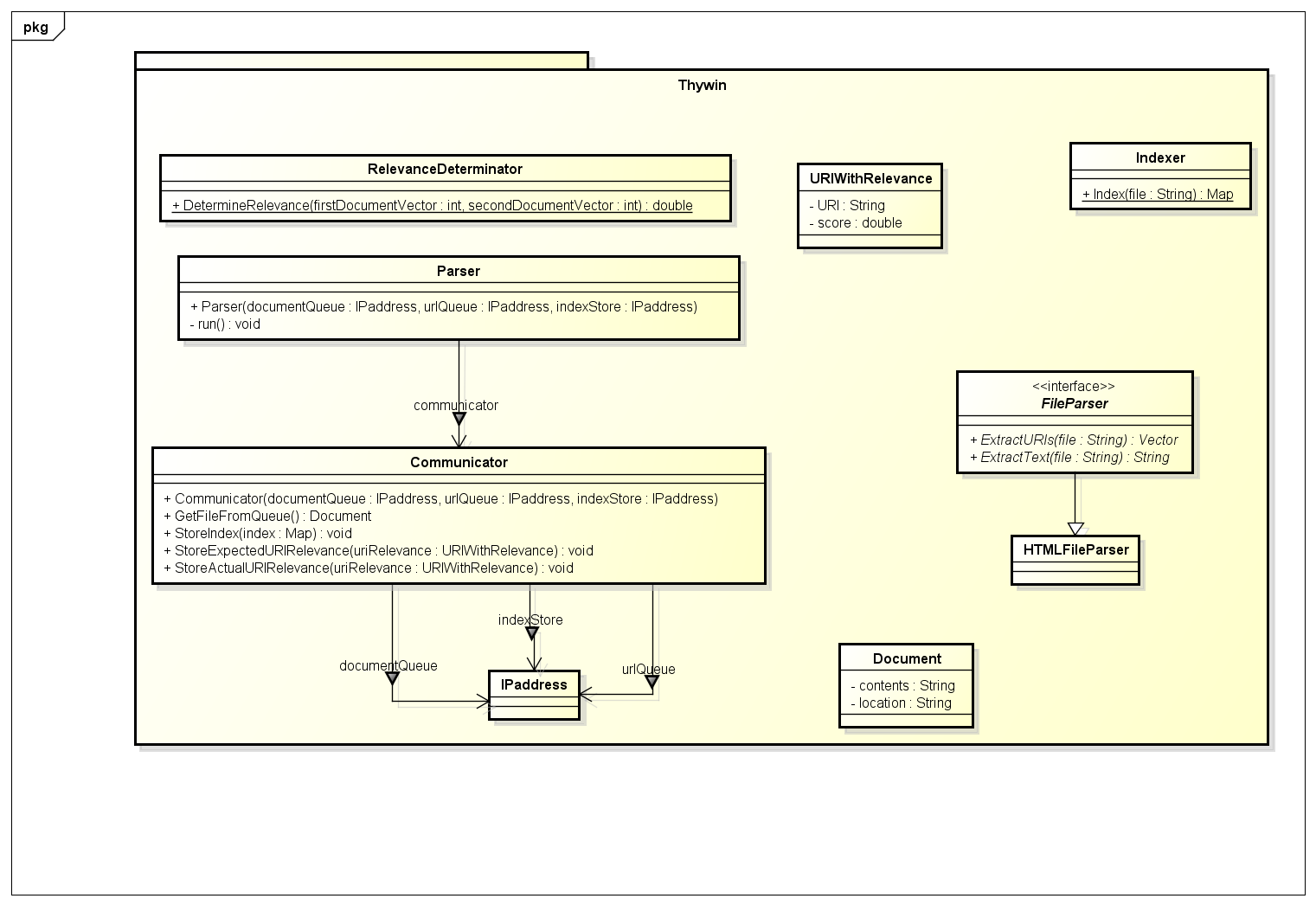


## Parser

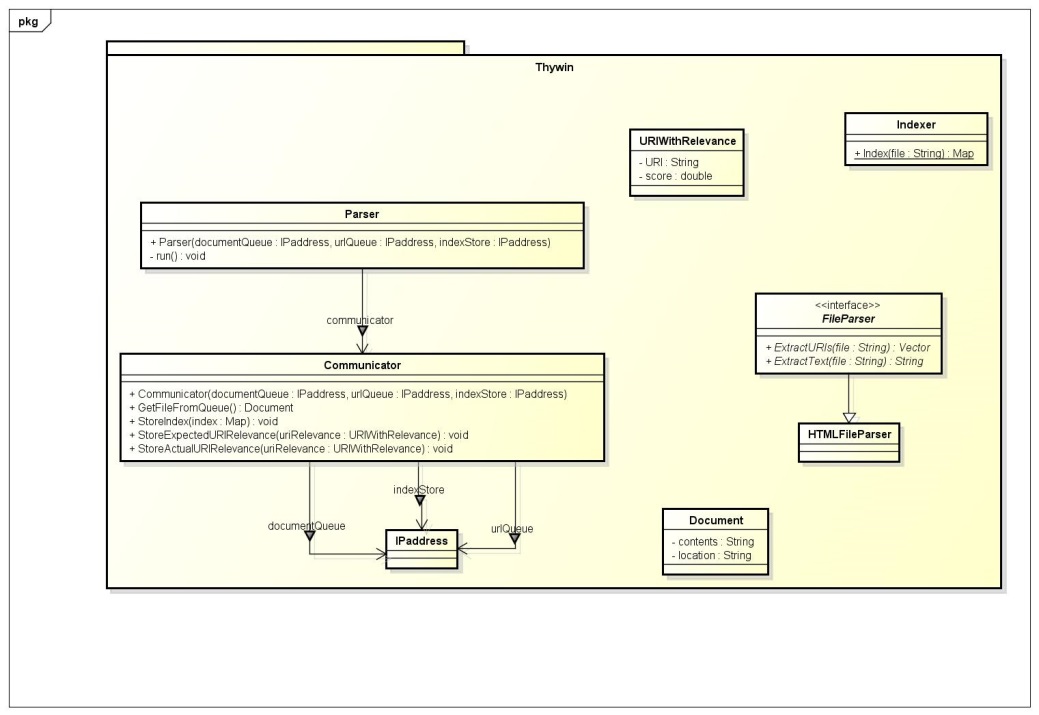
De Parser heeft als taak het verwerken van een opgehaald document. Dit betekend: Het tellen van unieke worden en het verzamelen van URIs van een document en deze door geven aan het Master proces voor opslag.

Versie 1.2 (huidig):

*Versie 1.1:*



*Versie 1.0:*

**

## Crawler

De Crawler zorgt er voor dat het document bijbehorend bij een URI opgehaald wordt van het web.

Versie 1.1 (huidig):

*Versie 1.0:*

