Universidade do Minho
Escola de Engenharia

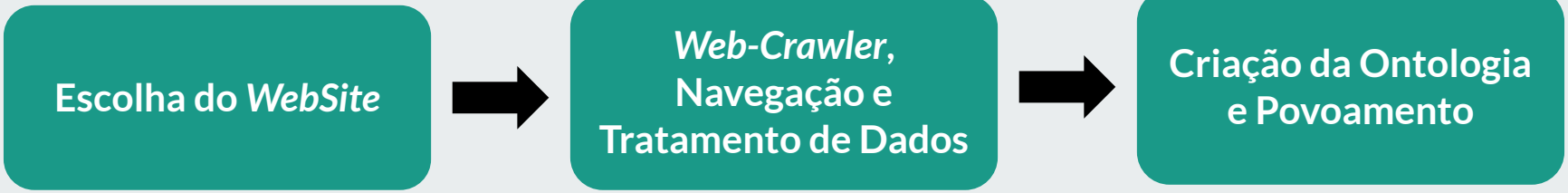
Web scraping + Ontologia

By Rafael Silva

Scripting no Processamento de Linguagem Natural

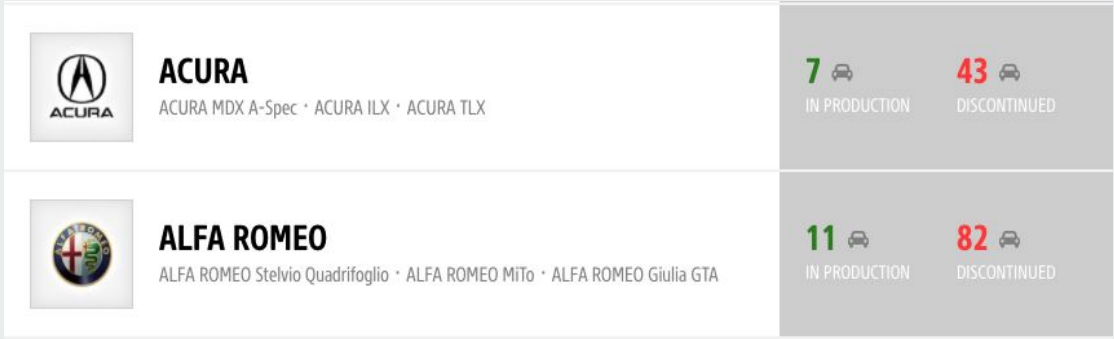
Universidade do Minho, Mestrado Integrado em Engenharia Informática,
4º Ano, 2º Semestre, Junho 2020

Fases do Projeto

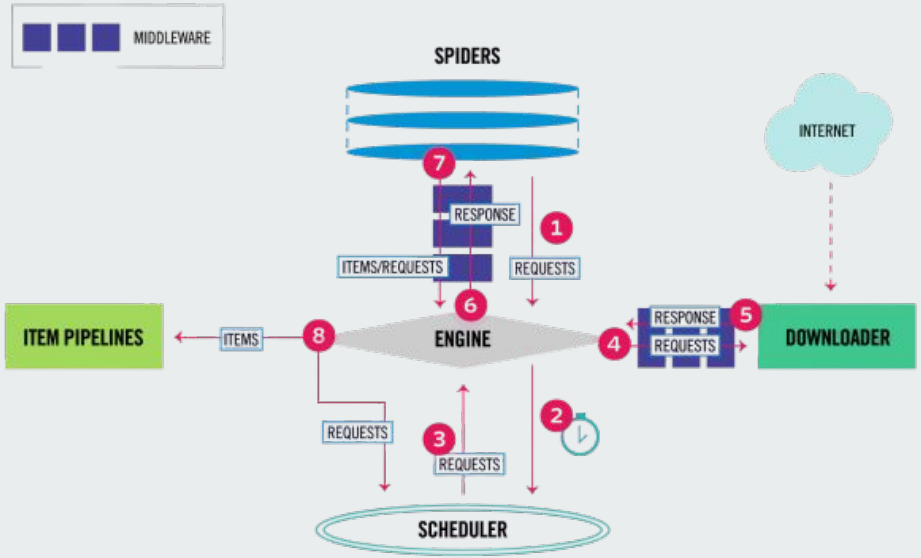


WebSite - AutoEvolution

- *Website* contém a informação de todas as marcas a nível mundial.
- Possui todos os modelos de todas as marcas.
- Versões de cada um desses modelos.
- Especificações de cada versão.

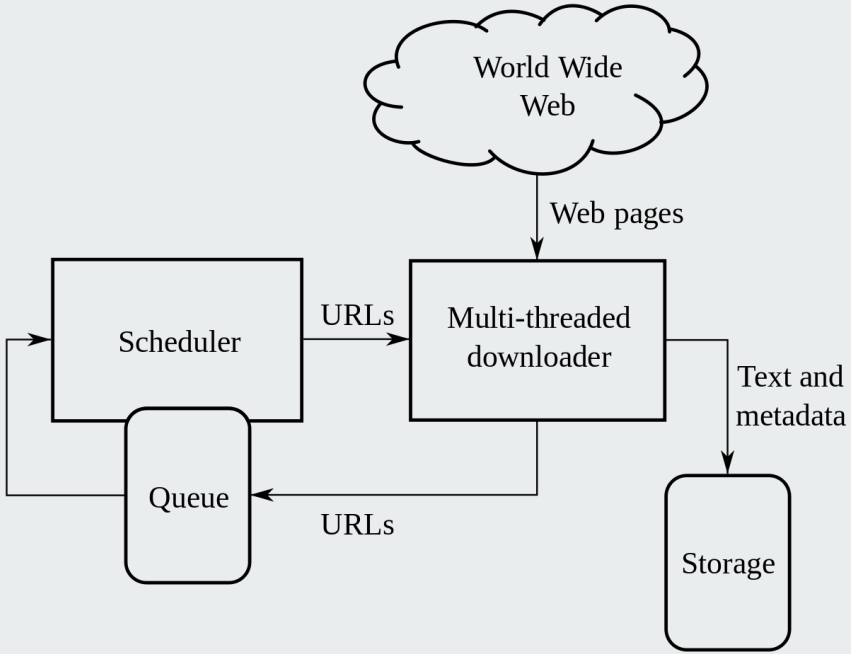


Scrapy



- Consiste numa *framework* de *Web-Crawling*.
- A arquitectura do **scrapy** é construída em torno de *"spiders"*, que são *crawlers* autônomos que recebem um conjunto de instruções.

Web-Crawler



- *Web-Crawler* é um programa que navega pela Internet de uma forma metódica e automatizada.
- O mesmo utiliza o código HTML da página fonte para fazer essa navegação.

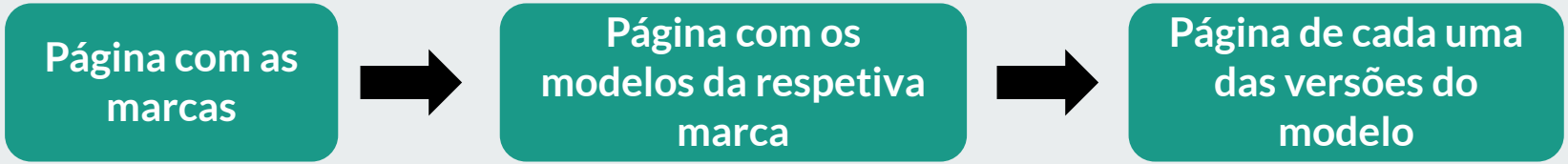
Desenvolvimento - *Script*

```
1 import scrapy
2 class AutoevolutionSpider(scrapy.Spider):
3     name = 'AutoEvolution'
4     allowed_domains = ['autoevolution.com']
5     start_urls = ['http://autoevolution.com/cars/']
6     def parse(self, response):
7         pass
```

- *Scrapy* configuração por default.
- array de *allowed_domains*
- array de *start_urls*

Navegação do *Web-Crawler*

- Primeiro o mesmo irá aceder ao *link* onde se encontram todas as marcas.
- De seguida irá recolher todas as informações.
- Visitar a *link da marca em questão* e recolher toda a informação necessária.
- Em seguida irá navegar para o *link* do correspondente a cada modelo e dentro deste irá também recolher dados.
- Por fim irá aceder ao *link* de cada versão do respetivo modelo.



Exemplo de um *parser*

```
...
def parseBrand(self, response):
    ...
    CAR_LIST_SELECTOR = '.carmod'
    for car in response.css(CAR_LIST_SELECTOR):
        ...
        URL_CAR = '.fl > a::attr(href)'
        url_car = car.css(URL_CAR).extract_first()
        model = {
            ...
            'url': url_car,
            ...
        }
        if url_car:
            request = scrapy.Request(url_car, callback=self.parseModelVersion, meta={
                'model': model, 'brand': brand})
            yield request
...
```

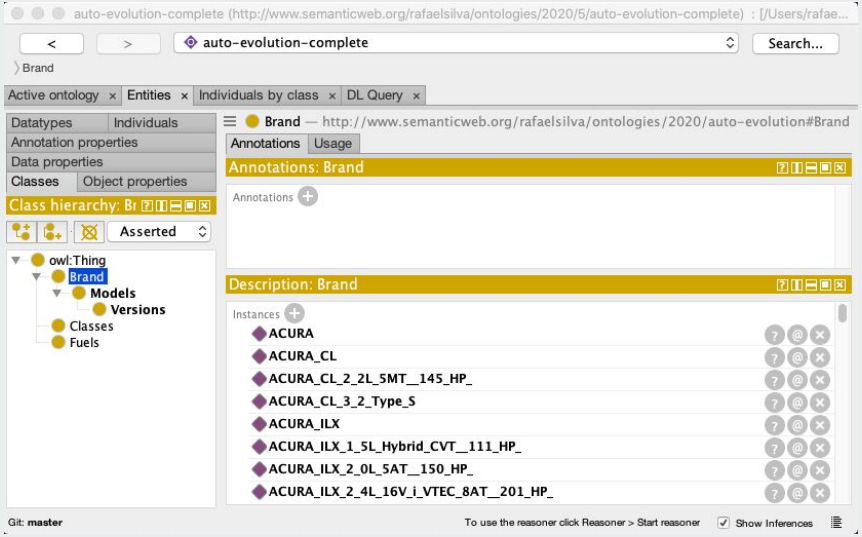
Tratamento de Dados

```
...
class AutoevolutionScraperItem(scrapy.Item):
    brand = scrapy.Field()
    model = scrapy.Field()
```

```
import json
class AutoevolutionScraperPipeline:
    def process_item(self, item, spider):
        line = json.dumps(dict(item), indent=2) + ',\n'
        self.file.write(line)
        return item
    def open_spider(self, spider):
        self.file = open('autoevolution.txt', 'w')
        line = '[\n'
        self.file.write(line)
    def close_spider(self, spider):
        line = ']\n'
        self.file.write(line)
        self.file.close()
```

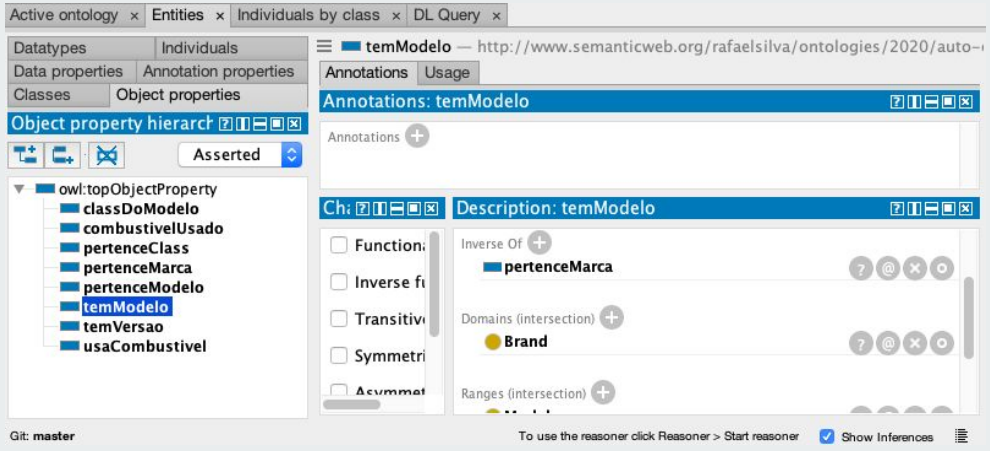
- Usar um objeto em *JSON*.
- Por cada *parser* guardam-se dados.
- Passar a informação de *parser* para *parser*.
- Completar o objeto final no último *parser*.
- Escrever esse objeto num ficheiro utilizando as *pipelines* do scrapy

Ontologia



- Possui quatro classes e duas subclasses.
- **Fuels**, **Brand** e **Classes** são as principais.
- **Models** e **Versions** são subclasses, a primeira de Brand e a segunda de Models.

Ontologia - Relações



- Possui oito relações.
- Quatro dessas relações são o inverso da correspondente.
- **pertenceClass** - relaciona **Models** e **Classes,** o seu inverso é **classDoModelo**.
- **temModelo** - relaciona **Brands** e **Models,** o seu inverso é **pertenceMarca**.
- **temVersao** - relaciona **Models** e **Versions,** o seu inverso é **pertenceModelo**.
- **usaCombustivel** - relaciona **Models** e **Fuels** o seu inverso é **combustivelUsado**.

Povoamento - Ontologia (Exemplo)

```
def populate_model(data):
    with open("auto_evolution.ttl", "a") as auto_evolution:
        for model in data:
            brand_name = model['nameBrand']
            ...
            model_aux = '''
###  http://www.semanticweb.org/rafaelsilva/ontologies/2020/auto-evolution#{0}
:{0} rdf:type owl:NamedIndividual ,
            :Models ;
    :pertenceClass :{1} ;
    :pertenceMarca :{2} ;
    :usaCombustivel {3}
    :imgModel "{4}"^^xsd:string ;
    :modelYears "{5}"^^xsd:string ;
    :nameModel "{0}"^^xsd:string ;
    :nrGenerationsModel "{6}"^^xsd:string ;
    :urlModel "{7}"^^xsd:string .
'''.format(model_name,model_class,brand_name,fuels_aux,model_img,modelYears,
    model_nrGenerations,model_url)
        ...
```

GraphDB

Class	Links	
:Models	60K	⇄
:Brand	29K	⇄
:Fuels	11K	⇄
:Classes	8K	⇄
:Versions	6K	←

Universidade do Minho
Escola de Engenharia

Web-Scraping + Ontologia

By Rafael Silva

Scripting no Processamento de Linguagem Natural

Universidade do Minho, Mestrado Integrado em Engenharia Informática,
4º Ano, 2º Semestre, Junho 2020