

MIR

A **M**edia **I**nformation **R**etrieval System

Marco Masetti
grubert65@gmail.com

Who am I ?

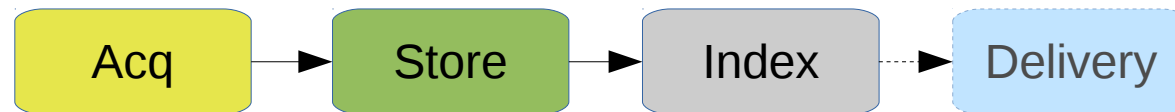
- Perl programmer since 2000
- Working in the research dptm of an Italian SME (> 200 employees)
- Both research and commercial projects
- Attended ACT conferences:
 - Italian Perl Workshop 2012
 - YAPC::Europe 2010
 - Italian Perl Workshop 2009
 - Italian Perl Workshop 2008
 - Italian Perl Workshop 2006
 - Italian Perl Workshop 2005
 - Italian Perl Workshop 2004
 - YAPC::Europe 2003

A gentle introduction to IR

- Information Retrieval (IR) is **finding material** (usually documents) of an **unstructured** nature (usually text) that satisfies an **information need** from within **large collections** (usually stored on computers) *(cit. C.D.Manning - "An introduction to Information Retrieval" - 2009)*
- MIR is a platform you can leverage to build distributed IR systems
- The MIR core comes as a single distribution

The MIR system

- MIR lets you configure and run **acquisition campaigns**
- A campaign is split into phases:

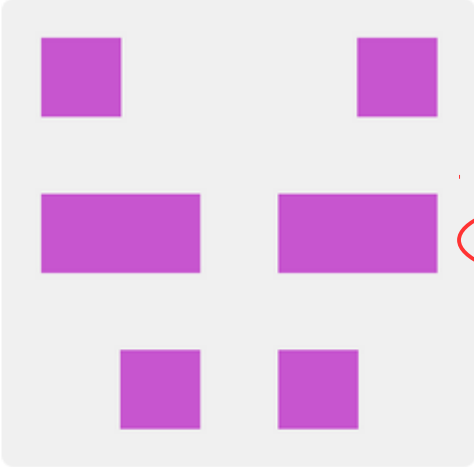


- Several campaigns can run in parallel
- MIR is domain-agnostic, highly configurable, can easily scale and is implemented in Perl

MIR main packages and scripts

- `Mir::Acq` : Acq processors (+ fetchers)
- `Mir::Store` : Metadata store
- `Mir::IR` : Indexing component
- `Mir::Config` : System configuration
- `Mir::Util` : Utility libraries
- `mir-acq-fetcher.pl` : to run a single fetcher
- `mir-acq-scheduler.pl` : to schedule the launch of an acq campaign
- `mir-acq-processor.pl` : to handle the running of campaign fetchers
- `mir-ir.pl` : indexing process (event-based)
- `mir-ir-polling.pl` : indexing process (polls for new docs to index)

MIR is on Github



Marco Masetti

grubert65

[Add a bio](#)

✉ grubert65@gmail.com

🕒 Joined on Sep 24, 2012

0

Followers


0

Starred

0

Following

Organizations



OverviewRepositoriesPublic activity

Edit profile

Popular repositories

Customize your pinned repositories

 **MooseX-Storage-IO-MongoDB**

Store and retrieve Moose objects to and from a MongoDB collection

1 ★

 **Dancer2-Plugin-REST**

0 ★

 **Mir**

Mir is a Media Information Retrieval platform...

0 ★

 **MooseX-Storage-IO-Redis**

The Redis driver of the Moosex::Storage::IO interface.

0 ★

 **perl5maven.com**

The source files of the Perl 5 Maven articles

0 ★

200 contributions in the last year

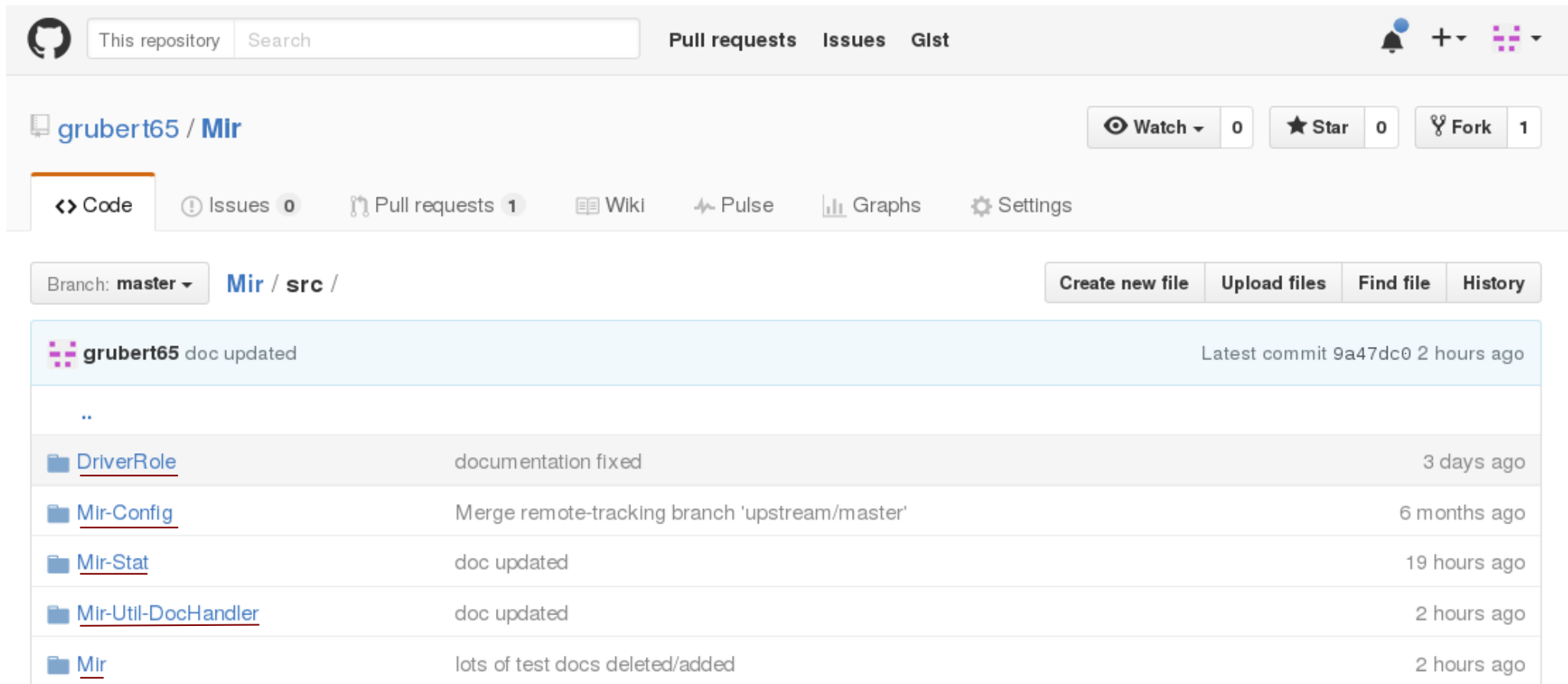
Contribution settings ▾

	Aug	Sep	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul
M												
W												
F												

Summary of pull requests, issues opened, and commits. [Learn how we count contributions.](#)

Less  More

The Mir repository...



This repository Search Pull requests Issues Gist

grubert65 / Mir Watch 0 Star 0 Fork 1

Code Issues 0 Pull requests 1 Wiki Pulse Graphs Settings

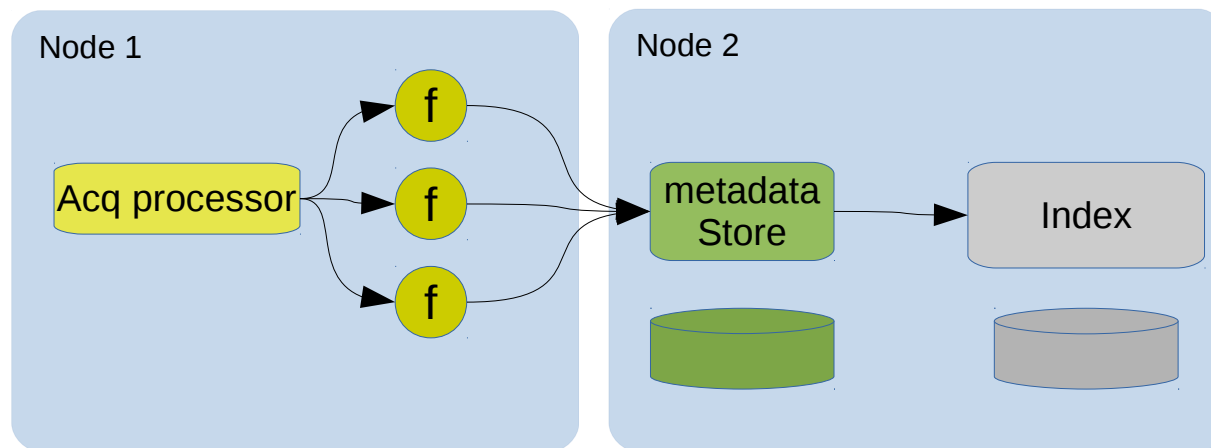
Branch: master Mir / src Create new file Upload files Find file History

grubert65 doc updated Latest commit 9a47dc0 2 hours ago

..		
DriverRole	documentation fixed	3 days ago
Mir-Config	Merge remote-tracking branch 'upstream/master'	6 months ago
Mir-Stat	doc updated	19 hours ago
Mir-Util-DocHandler	doc updated	2 hours ago
Mir	lots of test docs deleted/added	2 hours ago

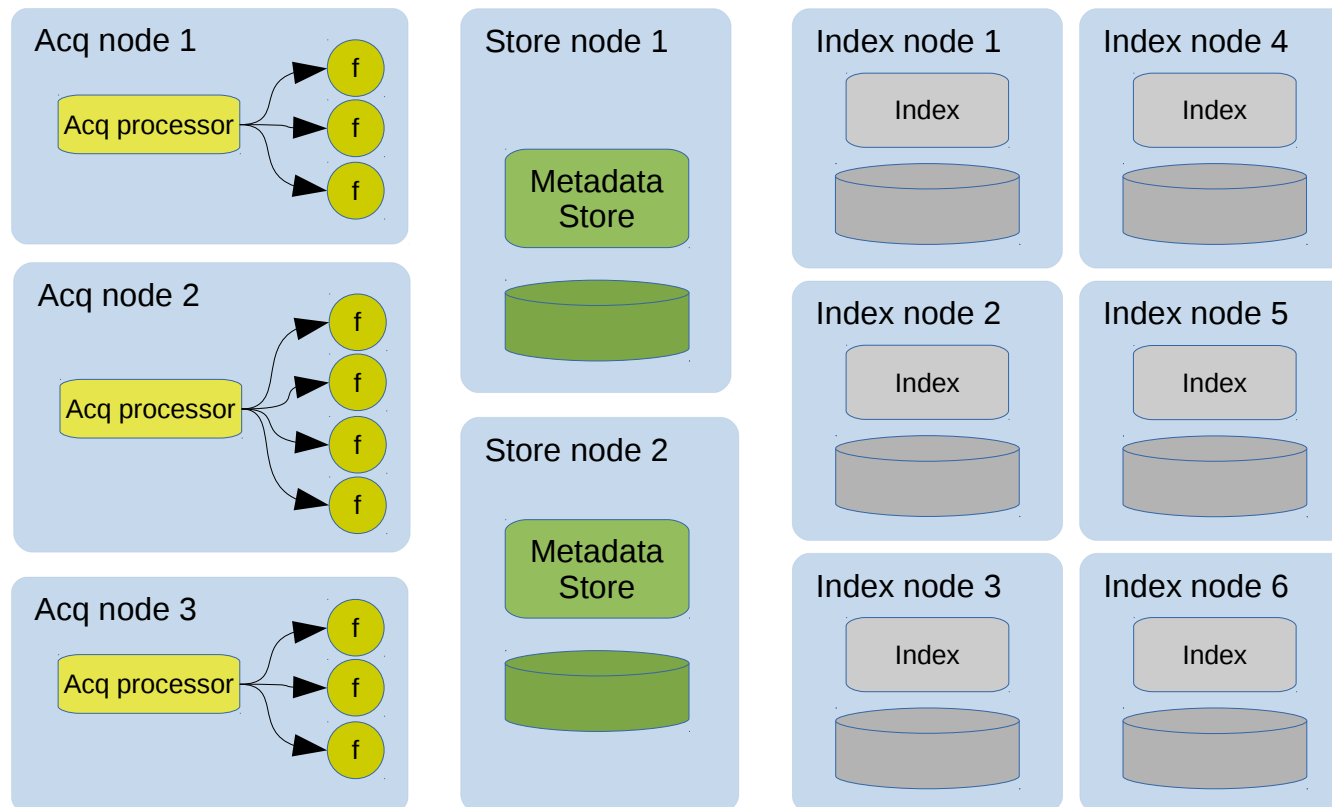
MIR system deployments (1)

- From simple configurations...



MIR system deployments (2)

- ...to complex ones...



Configuring a MIR system (1)

- Configuration is split into **sections**. Sections can be used to configure other sub-systems MIR interacts with.
- Configuration sections can be centralized in a single node or distributed
- By default a MIR system configuration section is stored in the **system** collection of the **MIR** database of the local **Mongo** server.
- Configuration sections can also be stored as distinct **JSON** files.
- MIR lets you configure acquisition campaigns. A campaign can be identified by a process chain from data acquisition, to metadata extraction down to information delivery

Configuring a MIR system (2)

- The system section usually holds:
 - one configuration item for each **acquisition campaign**
 - a configuration item for the **ElasticSearch** engine (not mandatory).

Metadata handling and storage

- A metadata should be a Moose class that should extends the `Mir::Doc` class or at least consume the `Mir::R::Doc` role.
- The `Mir::Doc` class consumes the `DriverRole` and the `Mir::R::Doc` roles
- The `DriverRole` implements the driver/interface pattern and lets you create a new metadata in this way:

```
my $doc = Mir::Doc->create(driver => 'Tweet');
```
- The `Mir::R::Doc` adds some common attributes (unique id, creation date,...) and consumes the `MooseX::Storage` role. The `MooseX::Storage` role provides persistency to Moose objects.
- A metadata obj is usually created by an acq fetcher and pass throw all process chain, usually being enriched while processing proceed.

Fetcher design

- A fetcher is a tiny bit of logic that gathers data from a given source and creates metadata. Fetchers come as perl distributions. A fetcher class should consume the **Mir::R::Acq::Fetch** role and belong to the **Mir::Acq::Fetcher** namespace.

```
package Mir::Doc::Tweet;
use Moose;
with 'Mir::R::Doc';

has 'lang' => (is => 'rw', isa => 'Str');
has 'place' => (is => 'rw', isa => 'Str');
has 'text' => (is => 'rw', isa => 'Str');
...
```

1) First define your metadata...

```
package Mir::Acq::Fetcher::Twitter;
use Moose;
with 'Mir::R::Acq::Fetch';

sub get_docs {
    # all the logic to interact with Twitter...
    my $tweet = Mir::Doc::Tweet->new(
        lang      => 'en',
        place     => 'Cluj-Napoca',
        text      => 'Following this weird pres...',
        ...
    );
    $tweet->store();
}
```

2) then implement the fetcher...

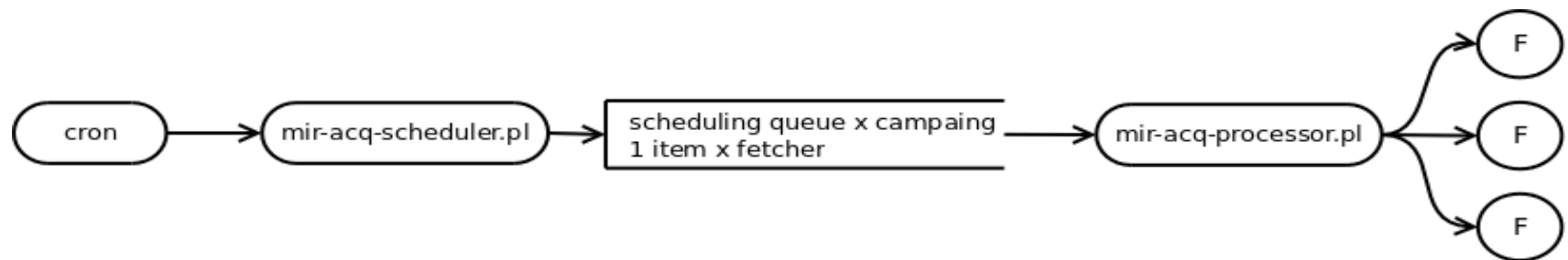
Fetchers implemented

Implemented by my team @ my company for commercial projects (not freely available...), the aim is to use CPAN to build a collection of freely available fetchers.

Mir::Acq::Fetcher::FS	Traverse (in many ways) a local or remote file system. Create a metadata for each new file found.
Mir::Acq::Fetcher::Instagram	Retrives published media according to configured criteria (es: lat,lon, radius)
Mir::Acq::Fetcher::Netatmo	Weather data from the Netatmo network of weather stations
Mir::Acq::Fetcher::OpenWeather	Weather data using OpenWeather API
Mir::Acq::Fetcher::WU	Weather data using WeatherUndergroud API
Mir::Acq::Fetcher::RSS	News feeds from an rss channel
Mir::Acq::Fetcher::Twitter	Use Twitter REST API, retrives tweets according to configured criteria
Mir::Acq::Fetcher::TwitterStream	Use Twitter Streaming API to retrieve tweets in stream mode

How fetchers are organized...

- This is quite complex, but complexity provides you freedom in metadata acquisition
- Basically:
 - Either run a single fetcher via **mir-acq-fetcher.pl**
 - Or configure an acq campaign using:
 - **Mir-acq-scheduler.pl** to schedule which campaign should start when (using for example the cron daemon)
 - **Mir-acq-processor.pl** to define how many fetchers for each campaign should be launched in parallel
 - The scheduler and the processor talk using a network queue



A real example of acquisition handling

```
# Mir Acq campaigns scheduling on node 1
# m h dom mon dow  command
30 19 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_commerciale
30 20 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_eccairs
0 */6 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_progetti
30 4 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_cv
30 5 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_segreteria_pub
30 6 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_segreteria_priv
30 7 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_segreteria_ris
30 8 * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign DocIndex_segreteria_usr
*/20 * * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign eccairs-SRIS
*/15 * * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign eccairs-AVIATION
0 * * * * /home/mir/perl5/perlbrew/perls/perl-5.22.0-threaded/bin/mir-acq-scheduler.pl --campaign rss-news
```

```
# Mir Acq processing on node 1
mir@mir:~$ ps -ef |grep perl
... perl mir-acq-processor.pl --campaign DocIndex_commerciale --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_eccairs --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_cv --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_segreteria_pub --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_segreteria_priv --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_segreteria_ris --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_segreteria_usr --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_progetti --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign DocIndex_segreteria_usr --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign bilanci --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign visure --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign eccairs-SRIS --log_config_params mir-acq-processor-log.conf
... perl mir-acq-processor.pl --campaign eccairs-AVIATION --log_config_params mir-acq-processor-log.conf
```


MIR-IR – text extraction

- Metadata can easily refer to files of some format
- **Mir::Util::DocHandler** is a class to extract text from a good range of file types, including: PDF, MS Office, ODF, images,...
- Drivers are implemented for each supported file suffix (Office, doc, docx, html, pdf, pdf2, pdf3, rtf, xls, ppt, txt, image), you get a driver for the proper format in the usual way:

```
my $dh = Mir::Util::DocHandler->create( driver => $suffix );  
$dh->open_doc("/path/to/doc.$suffix");  
my $num_pages = $dh->pages();  
my ( $text, $confidence ) = $dh->page_text($_) foreach (1..$num_pages);
```

- LUTs can be configured to use drivers for other suffix (es: 'htm' => 'html')
- The **mir-doc-handler.pl** script can be used to test how text extraction works on a given doc

MIR-IR – text indexing

- The text indexing process deals with text extraction and indexing into Elastic
- A text indexing process active for each campaign
- Text indexing can be driven in 2 ways:
 - The indexing process gets metadata from an indexing

```
# Mir indexing processes on node 2
mir@mir2:~$ ps -ef |grep perl
perl mir-ir-polling.pl --campaign DocIndex_commerciale --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_eccairs --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_cv --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_progetti --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_segreteria_pub --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_segreteria_priv --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_segreteria_ris --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign DocIndex_segreteria_usr --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign eccairs-SRIS --config_driver Mongo --config_params {"host":"192.168.56.21"}
perl mir-ir-polling.pl --campaign eccairs-AVIATION --config_driver Mongo --config_params {"host":"192.168.56.21"}
```

Success stories

perl elastic

Cerca

Ricerca avanzata

Risultati della ricerca

Trovati 528 documenti

File	Correa_Vito_ricv.docx
Ultima modifica	2016-07-20 06:23:14
Cartella	srvcluster1/segreteria_pub/DATI_PERSONALI/CV/CV_ESTERNI/EM_01_2016
File	Giuliano_Stefano_cv.pdf
Ultima modifica	2012-03-26 07:23:36
Cartella	srvcluster1/segreteria_pub/DATI_PERSONALI/CV/CV_ESTERNI/GENOVA/2012
File	Palazzo_Olivero_ricv.docx
Ultima modifica	2014-01-10 10:15:56
Cartella	srvcluster1/segreteria_pub/DATI_PERSONALI/CV/CV_ESTERNI/INSERZIONI_2013/OS_GE_01_13
File	Masetti_Marco_cur.doc
Ultima modifica	2016-05-10 06:55:52
Cartella	srvcluster1/segreteria_pub/DATI_PERSONALI/CV/CV_INTERNALI/CV TECNICI/2016/nuovo_template
File	Vasilyev_Serhiy_cv.doc
Ultima modifica	2010-02-15 07:30:50
Cartella	srvcluster1/segreteria_pub/DATI_PERSONALI/CV/CV_ESTERNI/GENOVA/2010

What next...

- Improve documentation (tutorials,...)
- MIR system Monitoring GUI
- Use Vagrant or similar to ship appliances

Thank YOU!!!

Enjoy your **YAPC!!!**