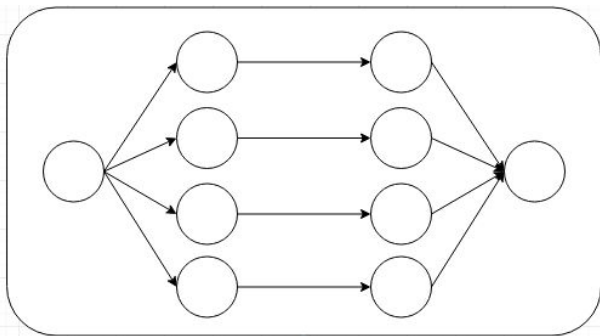


# Criando um pipeline no Portal I

Setup e conceitos básicos



Cristiano Singulani, Lucas Dal Piaz Nunes

# Outline

---

- Instâncias do portal
- Setup inicial
- Estudo do pipeline `example_objectscount`
- Versionamento de código: testing e production

# Instâncias do portal (Ambientes)

---

- testing - ambiente usado para dataset menores pois os recursos computacionais são reduzidos
  - Compartilha a infraestrutura com o portal dos desenvolvedores
  - Cluster com 4 nós de processamento
- production - ambiente usado para fazer ciência
  - loginix - 20 nós de processamento
  - rocks - 42 nós de processamento

# Setup inicial

---

- Tutorial disponível em:
  - <https://devel2.linea.gov.br/~lucas.nunes/site/>

## Pré-requisitos:

- acesso ao servidor de desenvolvimento
- conhecimento básico de linux
- git
- python

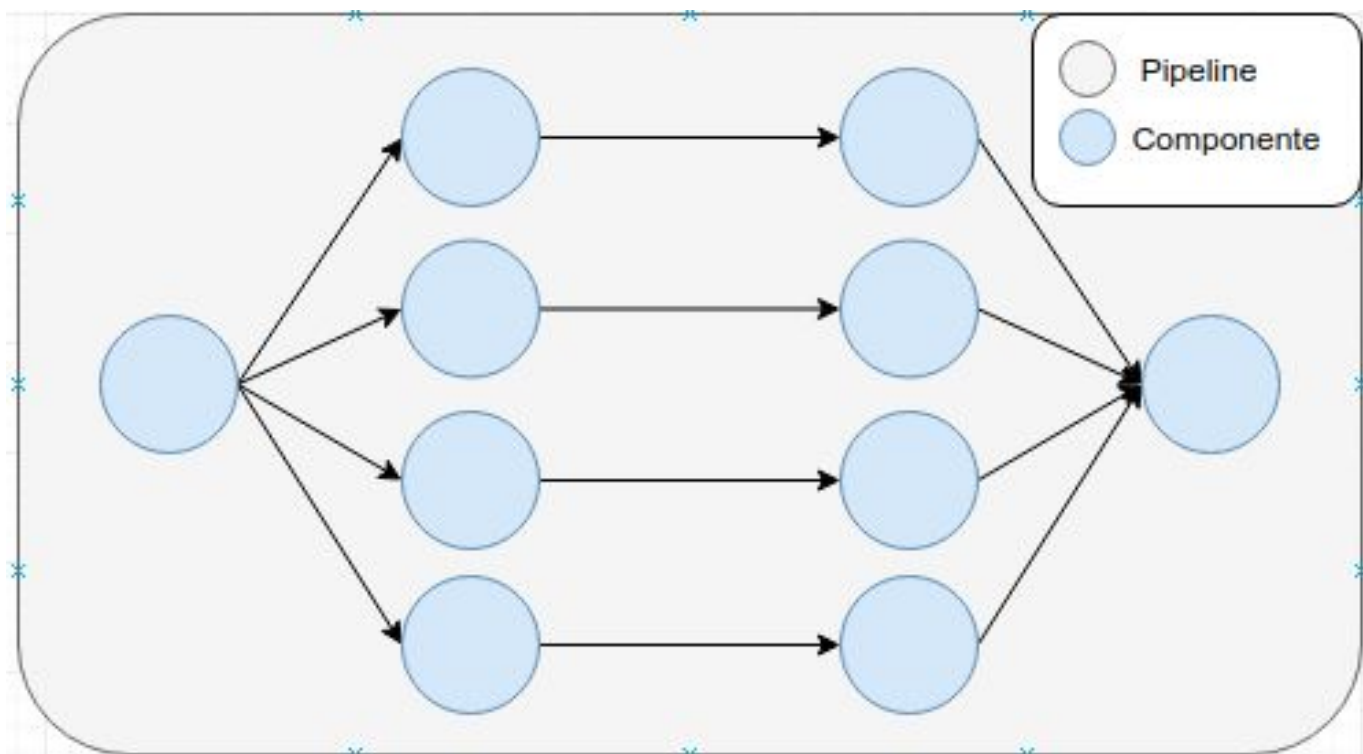
**COLA**

**Scrip para fazer automaticamente**

`/home/aguena/bin_share/set_up_portal`

# Repositório de códigos

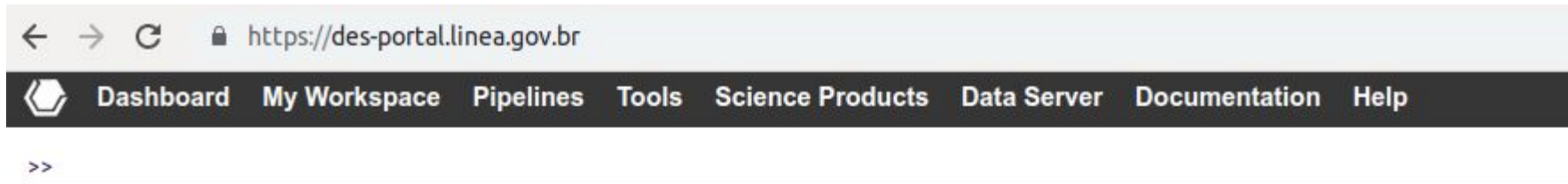
- `$PORTAL_ROOT/des/pipelines/`
- `$PORTAL_ROOT/des/components/`



# Repositório de códigos

---

- \$PORTAL\_ROOT/pypeline/
- \$PORTAL\_ROOT/scienceportal/



## DES Science Portal: Workflows

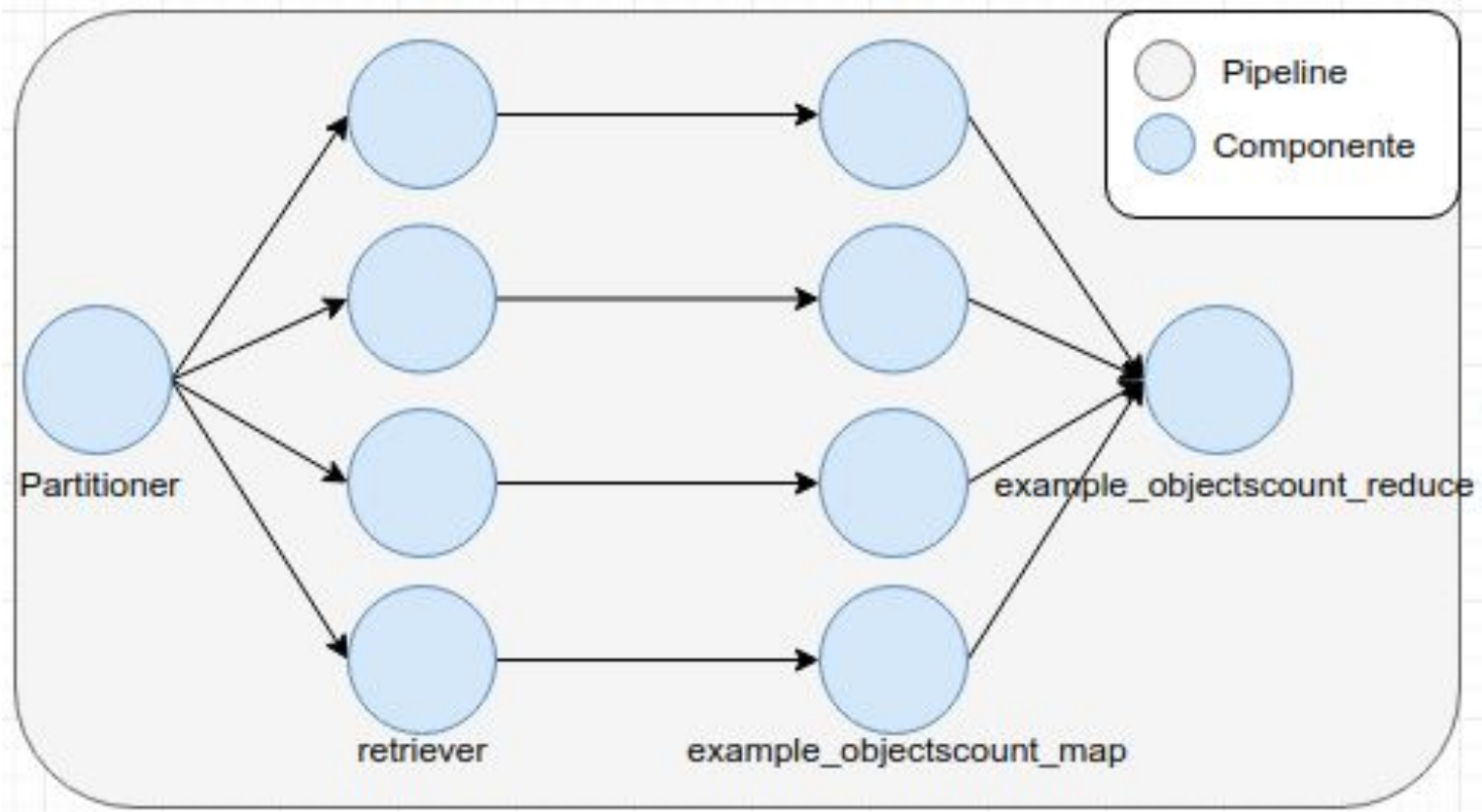
The Science Portal has two instances:

- **Workflows:** hosts workflows for Quality Assessment (QA), for the creation of Value-Added Catalogs (VACs) and for Science Analysis.
- **Data Server:** provide access to the Catalog Server and published results

The system is designed to be self-evident, use the help icon "(?)" available on each page.

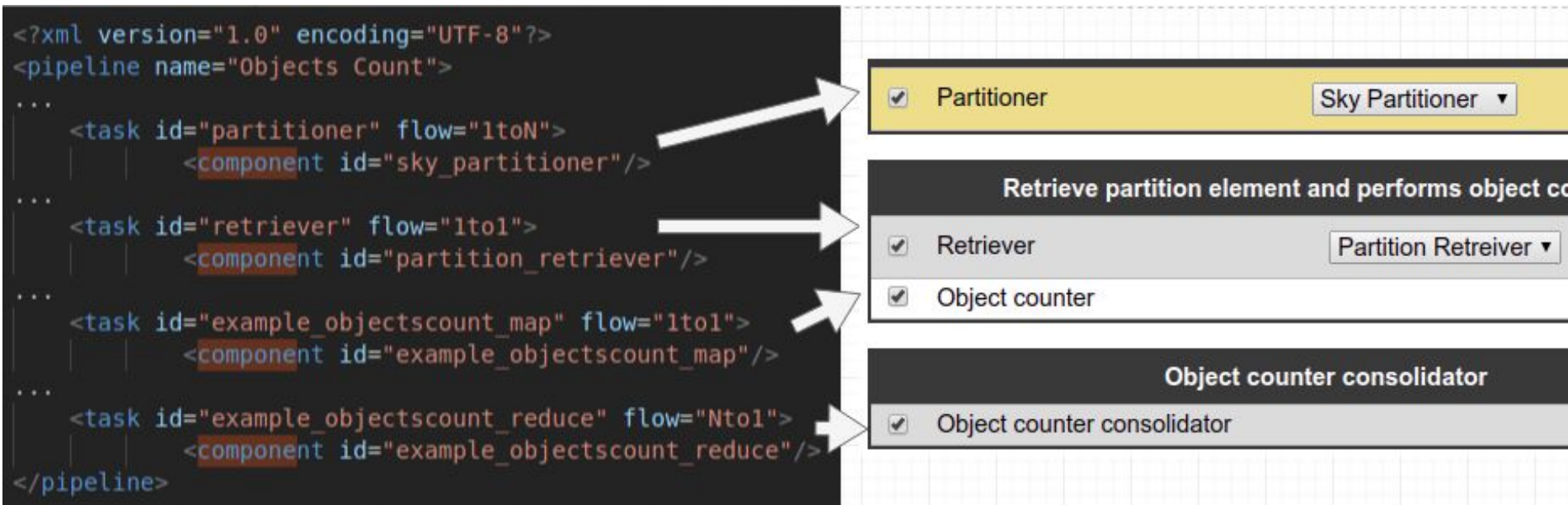
# Estudo do pipeline example\_objectscount

- Motivação



# Estudo do pipeline example\_objectscount

- Pipeline.xml
  - Fluxo de execução dos componentes
  - Dependência entre componentes

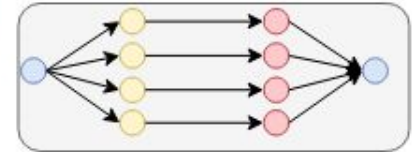




# Estudo do pipeline example\_objectscount

- Comunicação entre componentes

```
self.put_file_1_1(  
    path=fits_file,  
    file_class="science",  
    mime_type="application/fits-table",  
    local_scratch_area_only=True  
)
```



```
<task id="example_objectscount_map" name="Object counter" config="no" checked="checked" flow="1to1">  
  <components>  
    <!-- Reads MULTIPLE "science" "application/fits-table" file of a partition element -->  
    <component id="example_objectscount_map" name="Object counter" multithreaded="False" />  
  </components>  
  <dependencies>  
    <dependency id="retriever">  
      <file class="science" mimetype="application/fits-table" optional="False" />  
    </dependency>  
  </dependencies>  
</task>
```

```
input_file_list = self.io.getFileListByMimeClass(  
    file_class="science",  
    file_mime="application/fits-table"  
)  
self.info("Input files: '%s'" % self.to_str(input_file_list))
```

# Estudo do pipeline example\_objectscount

- Declaração input/output de pipelines

```
<input>
  <product class="coadd_objects" optional="True" />
  <product class="bcc_objects" optional="True"/>
  <product class="mice_objects" optional="True"/>
</input>
```

```
<output>
  <product class="vac_generic" />
  <product class="vac_lss" />
  <product class="vac_ge" />
  <product class="vac_ga" />
  <product class="vac_cluster" />
  <product class="footprint_map" />
</output>
```

# Estudo do pipeline example\_objectscount

- Product Log

```
self.put_parameter(  
    section="Total count result",  
    name="Total objects",  
    value=total_objects  
)  
  
for element in partition_elements:  
    self.put_parameter(  
        section="Count results per file",  
        name=element["name"],  
        value=element["n_objects"]  
    )
```

## Total count result

Total objects	313380
---------------	--------

## Count results per file

6833_rows_1_to_128	128
6833_rows_129_to_256	128
6812_rows_1_to_103	103
6812_rows_104_to_206	103
6812_rows_207_to_309	103
6812_rows_310_to_412	103
6812_rows_413_to_515	103
6812_rows_516_to_618	103
6812_rows_619_to_721	103
6812_rows_722_to_824	103
6812_rows_825_to_927	103

# Estudo do pipeline example\_objectscount

- Depuração de código
  - /mnt/scratch/users/<user>/master\_des/<process\_id>
  - 1 pasta por job \*\_sandbox
  - python -c "import components.<component\_name> as a; a.run()"

# Versionamento de código

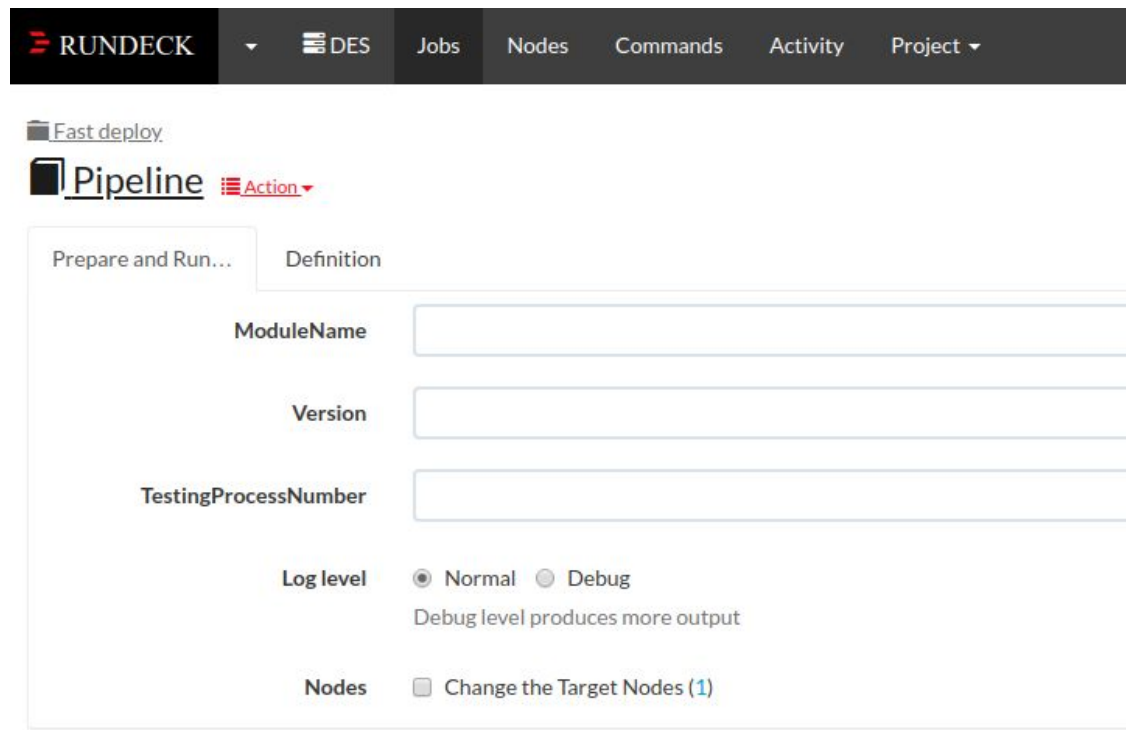
---

- fluxo de desenvolvimento local
  - `git pull`
  - `git checkout -b new_branch`
  - `git commit -a -m "new feature"`
  - `git checkout master`
  - `git merge new_branch`
- levar mudanças para repositório central
  - `git add ChangeLog VERSION`
  - `git commit -m "explanation"`
  - `git tag `cat VERSION``
  - `git push --tag`

# Versionamento de código

---

- testing - script to update
  - `sudo -H -u testing -i /home/testing/update_portal.sh`
- production - fast deploy



The screenshot shows the Rundeck web interface. At the top, there is a navigation bar with the following items: RUNDECK, a dropdown arrow, DES, Jobs, Nodes, Commands, Activity, and Project. Below the navigation bar, the breadcrumb path is 'Fast deploy' > 'Pipeline'. The 'Pipeline' page has two tabs: 'Prepare and Run...' and 'Definition'. The 'Definition' tab is active and contains the following configuration fields:

- ModuleName**: A text input field.
- Version**: A text input field.
- TestingProcessNumber**: A text input field.
- Log level**: Radio buttons for 'Normal' (selected) and 'Debug'. Below this, it says 'Debug level produces more output'.
- Nodes**: A checkbox labeled 'Change the Target Nodes (1)'. The number '1' is in blue.

---

---

# Questões

---

---

# Obrigado