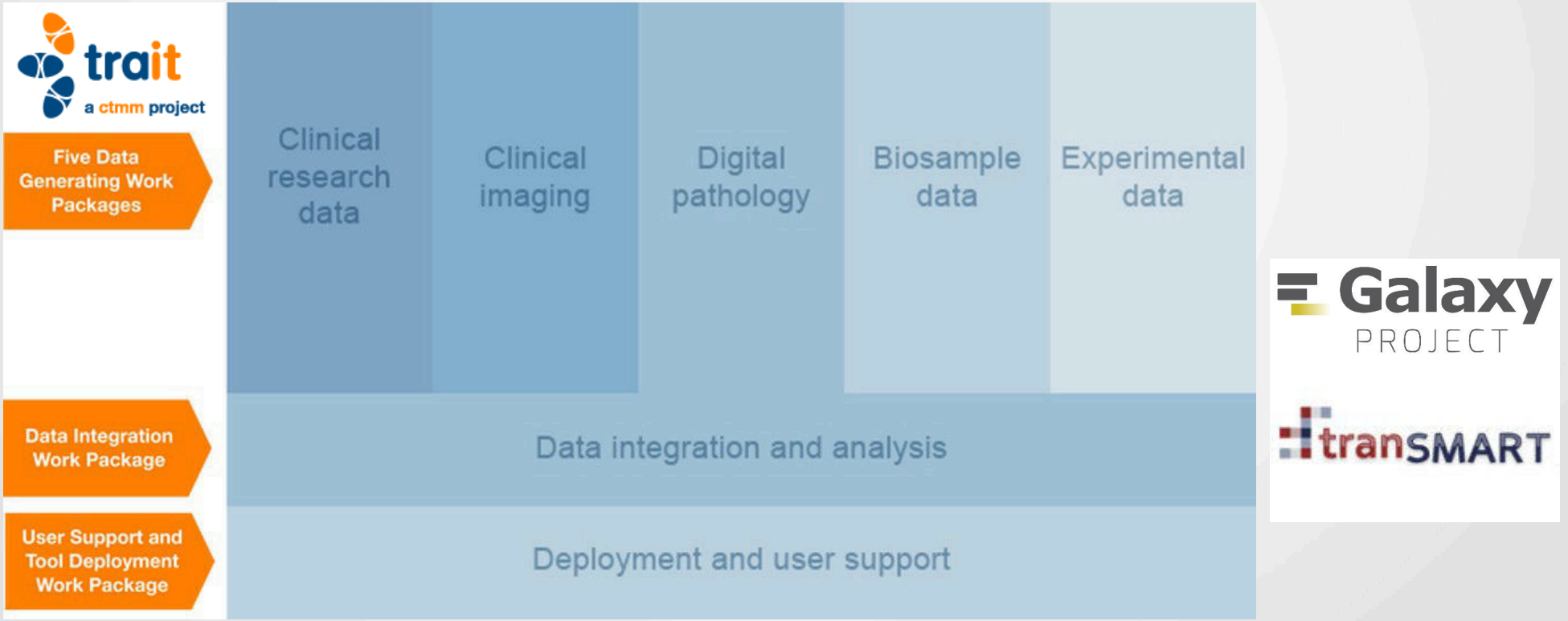
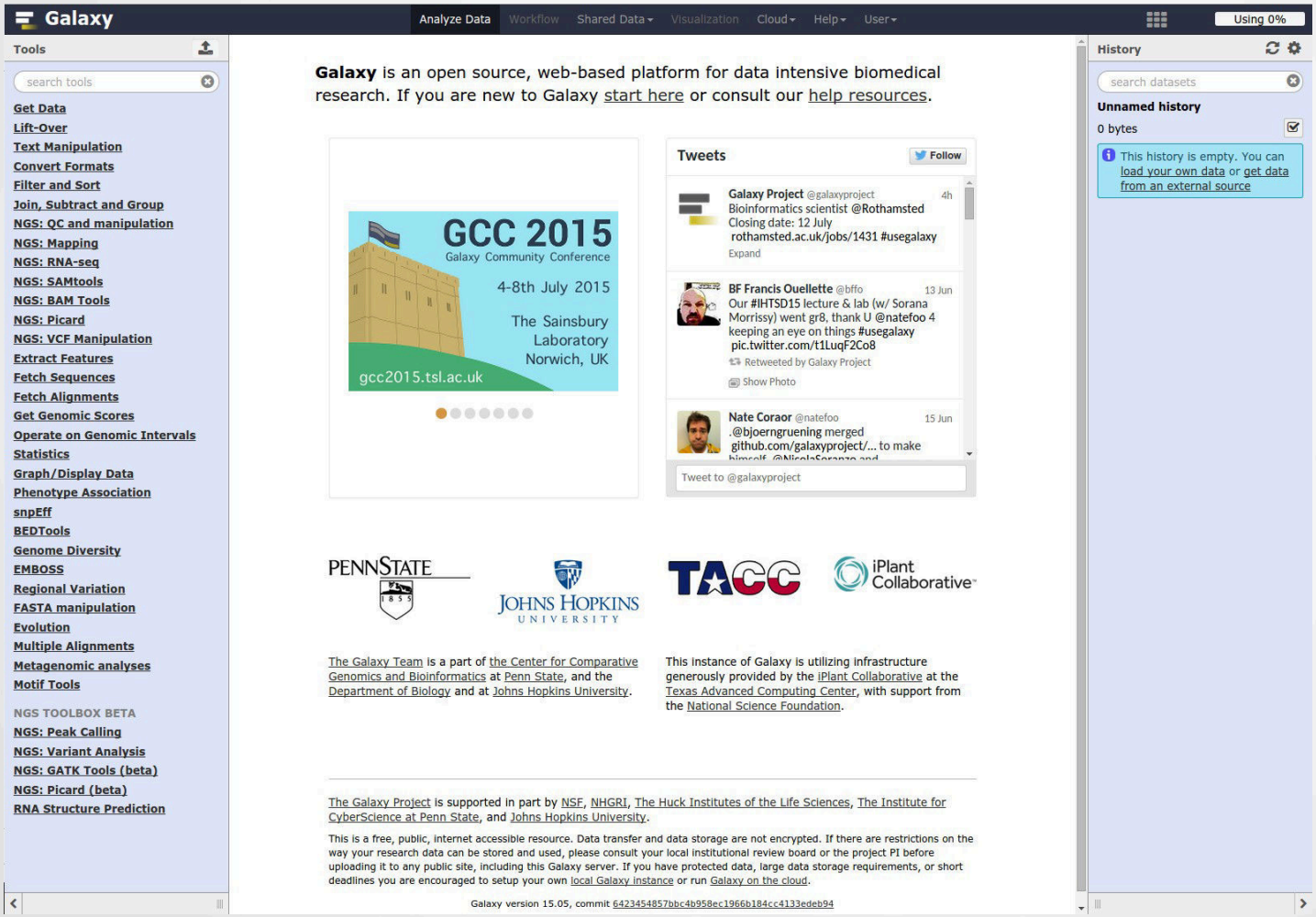
Galaxy as backend for TraIT genotype to phenotype studies

Youri Hoogstrate
GCC2015
7 July 2015, Norwich

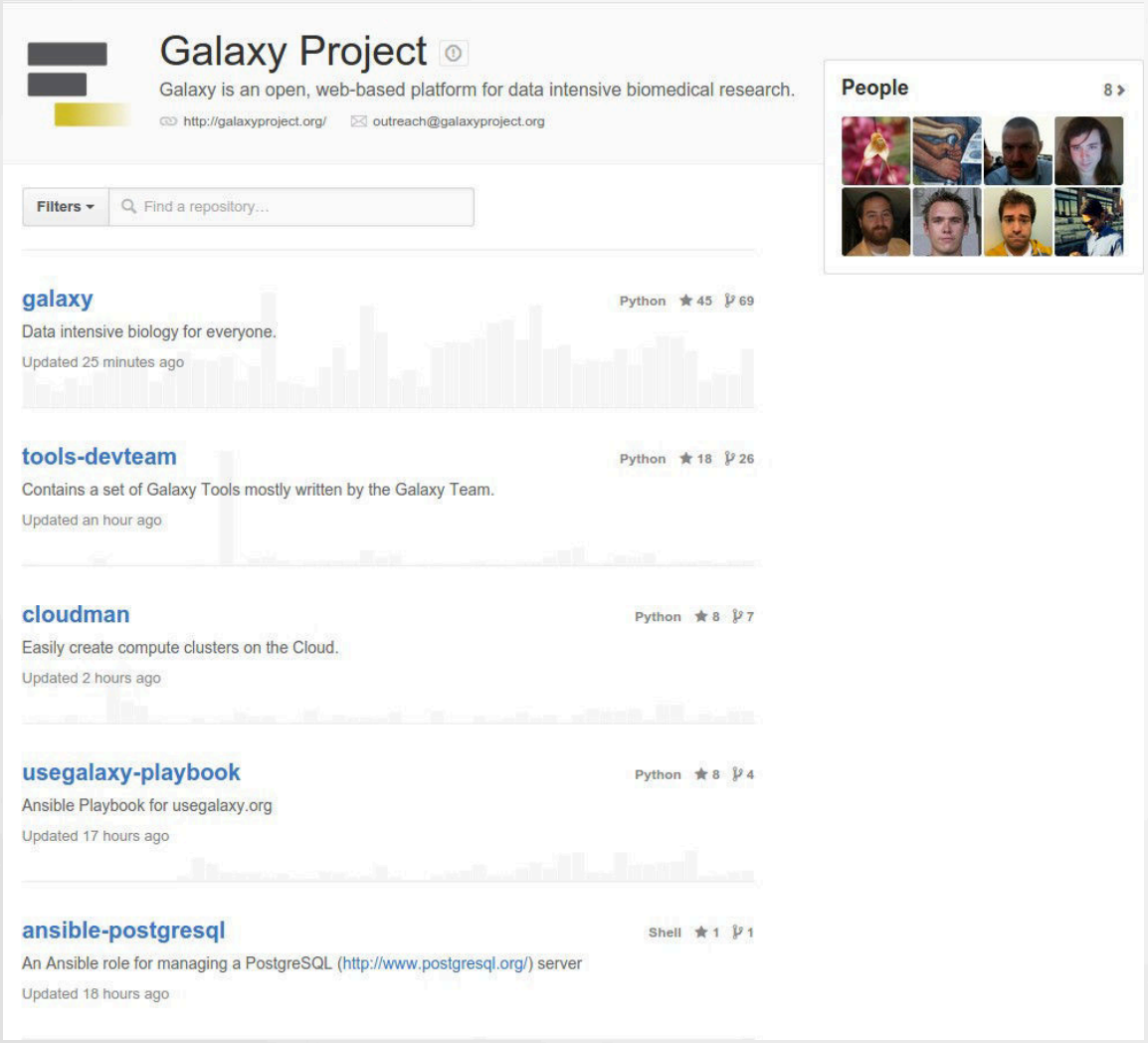
TraIT, TranSMART & Galaxy



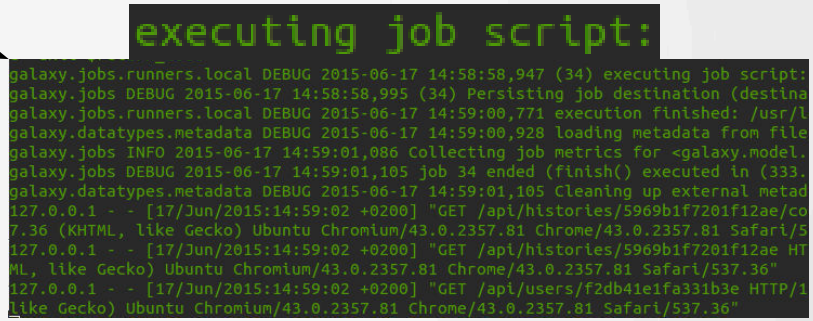
When you think of Galaxy



When you think of Galaxy



Purpose of Galaxy: using



Purpose of Galaxy: using

```
executing job script:
galaxy.jobs.runners.local DEBUG 2015-06-17 14:58:58,947 (34) executing job script:
galaxy.jobs DEBUG 2015-06-17 14:58:58,995 (34) Persisting job destination (destina
galaxy.jobs.runners.local DEBUG 2015-06-17 14:59:00,771 execution finished: /usr/l
galaxy.datatypes.metadata DEBUG 2015-06-17 14:59:00,928 loading metadata from file
galaxy.jobs INFO 2015-06-17 14:59:01,086 Collecting job metrics for <galaxy.model.
galaxy.jobs DEBUG 2015-06-17 14:59:01,105 job 34 ended (finish() executed in (333.
galaxy.datatypes.metadata DEBUG 2015-06-17 14:59:01,105 Cleaning up external metad
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/histories/5969b1f7201f12ae/co
7.36 (KHTML, like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/S
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/histories/5969b1f7201f12ae HT
ML, like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/537.36"
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/users/f2db41e1fa331b3e HTTP/1
like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/537.36"
```

Purpose of Galaxy: using

```
executing job script:
galaxy.jobs.runners.local DEBUG 2015-06-17 14:58:58,947 (34) executing job script:
galaxy.jobs DEBUG 2015-06-17 14:58:58,995 (34) Persisting job destination (destina
galaxy.jobs.runners.local DEBUG 2015-06-17 14:59:00,771 execution finished: /usr/l
galaxy.datatypes.metadata DEBUG 2015-06-17 14:59:00,928 loading metadata from file
galaxy.jobs INFO 2015-06-17 14:59:01,086 Collecting job metrics for <galaxy.model.
galaxy.jobs DEBUG 2015-06-17 14:59:01,105 job 34 ended (finish() executed in (333.
galaxy.datatypes.metadata DEBUG 2015-06-17 14:59:01,105 Cleaning up external metad
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/histories/5969b1f7201f12ae/co
7.36 (KHTML, like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/S
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/histories/5969b1f7201f12ae HT
ML, like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/537.36"
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/users/f2db41e1fa331b3e HTTP/1
like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/537.36"
```

```python
from bioblend.galaxy.objects import GalaxyInstance
gi = GalaxyInstance("URL", "API_KEY")
wf = gi.workflows.list()[0]
hist = gi.histories.list()[0]
inputs = hist.get_datasets()[:2]
input_map = dict(zip(wf.input_labels, inputs))
params = {"Paste1": {"delimiter": "U"}}
wf.run(input_map, "wf_output", params=params)
```

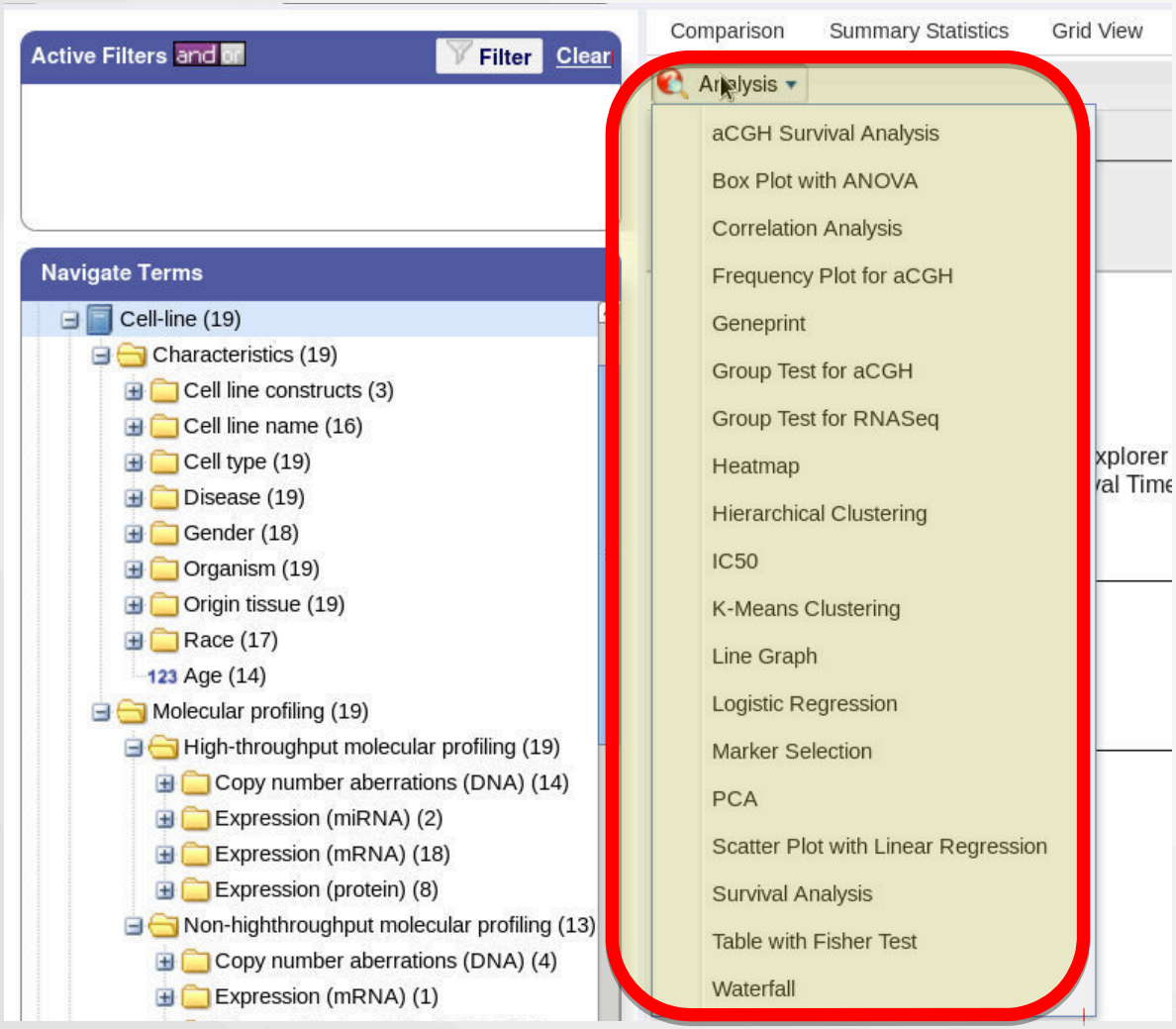
Purpose of Galaxy: using

```
executing job script:
galaxy.jobs.runners.local DEBUG 2015-06-17 14:58:58,947 (34) executing job script:
galaxy.jobs DEBUG 2015-06-17 14:58:58,995 (34) Persisting job destination (destina
galaxy.jobs.runners.local DEBUG 2015-06-17 14:59:00,771 execution finished: /usr/l
galaxy.datatypes.metadata DEBUG 2015-06-17 14:59:00,928 loading metadata from file
galaxy.jobs INFO 2015-06-17 14:59:01,086 Collecting job metrics for <galaxy.model.
galaxy.jobs DEBUG 2015-06-17 14:59:01,105 job 34 ended (finish() executed in (333.
galaxy.datatypes.metadata DEBUG 2015-06-17 14:59:01,105 Cleaning up external metad
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/histories/5969b1f7201f12ae/co
7.36 (KHTML, like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/S
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/histories/5969b1f7201f12ae HT
ML, like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/537.36"
127.0.0.1 - - [17/Jun/2015:14:59:02 +0200] "GET /api/users/f2db41e1fa331b3e HTTP/1
like Gecko) Ubuntu Chromium/43.0.2357.81 Chrome/43.0.2357.81 Safari/537.36"
```

```python
from bioblend.galaxy.objects import GalaxyInstance
gi = GalaxyInstance("URL", "API_KEY")
wf = gi.workflows.list()[0]
hist = gi.histories.list()[0]
inputs = hist.get_datasets()[:2]
input_map = dict(zip(wf.input_labels, inputs))
params = {"Paste1": {"delimiter": "U"}}
wf.run(input_map, "wf_output", params=params)
```

tranSMART

TranSMART



TranSMART
- Clinical data
- Interpreted data
  - VCF files
  - Expression tables
  - Copy number aberration

No intermediate files
- No BAM files
- No raw FASTQ files

The right man for the right job

+ Clinical data
  + Interactive cohort selection
+ Drag 'n Drop cohorts

– Analysis suite
– Large experimental data

+ API
+ Many bioinformatics apps (>3300)
+ Scaling resources for tools

– Link clinical- to experimental data
– Drag 'n Drop cohorts

The right man for the right job

+ Clinical data
  + Interactive cohort selection
+ Drag 'n Drop cohorts

– Analysis suite
– Large experimental data

+ API
+ Many bioinformatics apps
+ Scaling resources for tools

– Link clinical- to experimental data
– Drag 'n Drop cohorts

The right man for the right job

tranSMART

+ Clinical data
  + Interactive cohort selection
+ Drag 'n Drop cohorts

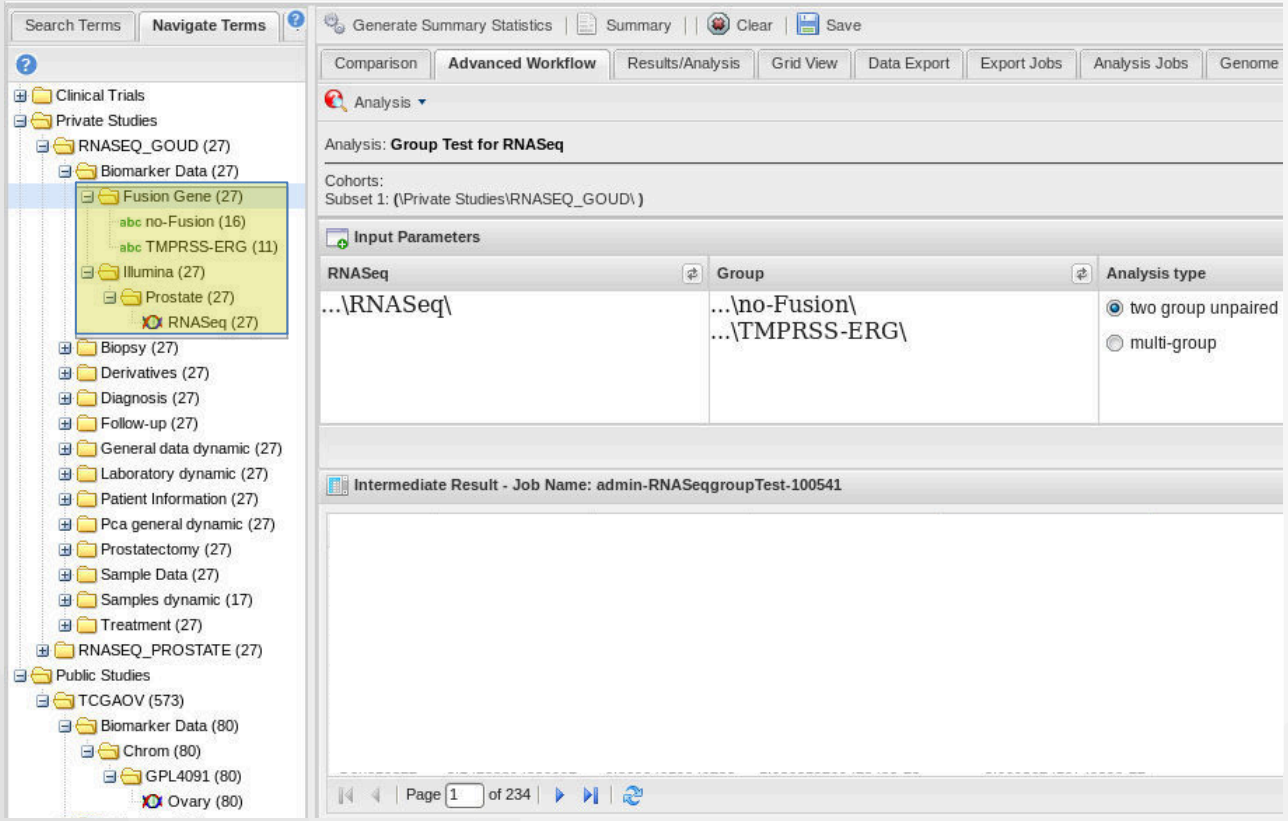
– Analysis suite
– Large experimental data

Galaxy

+ API
+ Many bioinformatics apps
+ Scaling resources for tools

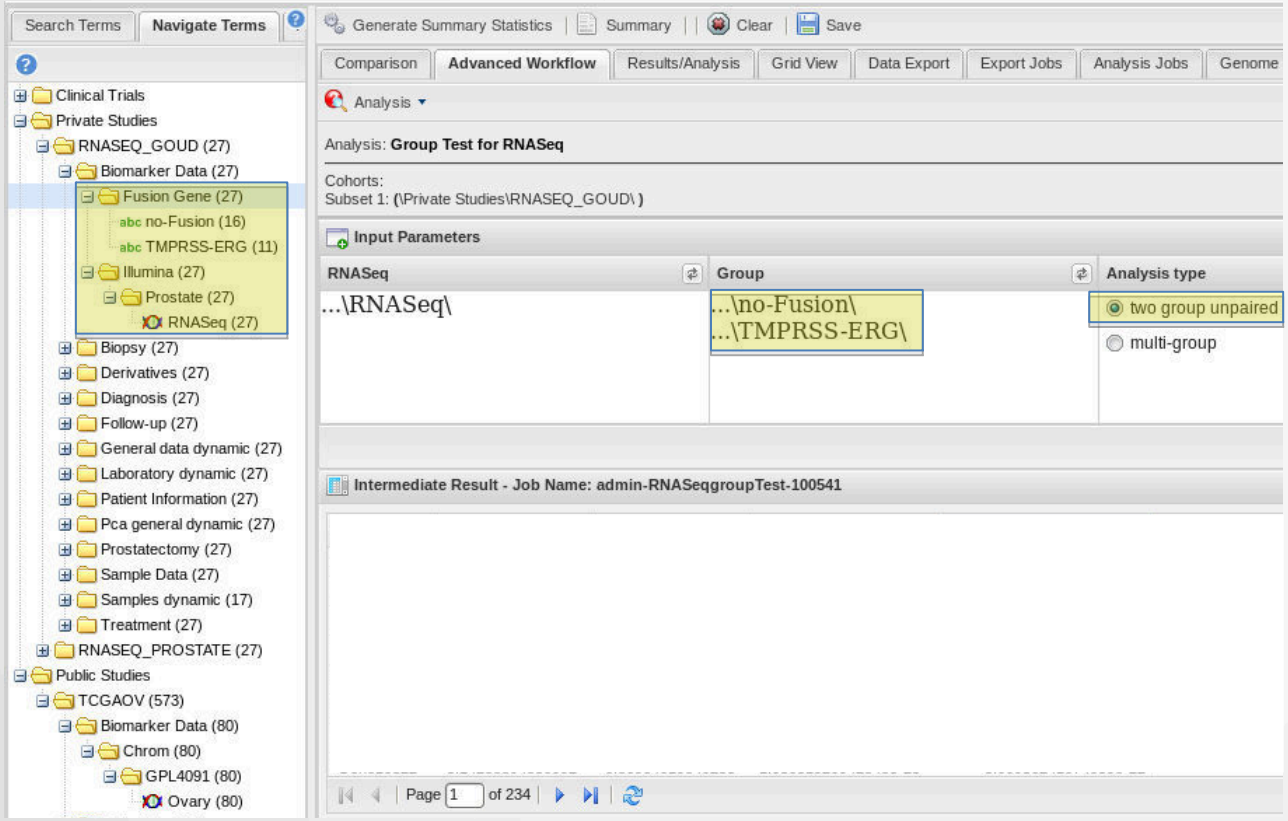
– Link clinical- to experimental data
– Drag 'n Drop cohorts

Example: PoC: RNA-seq



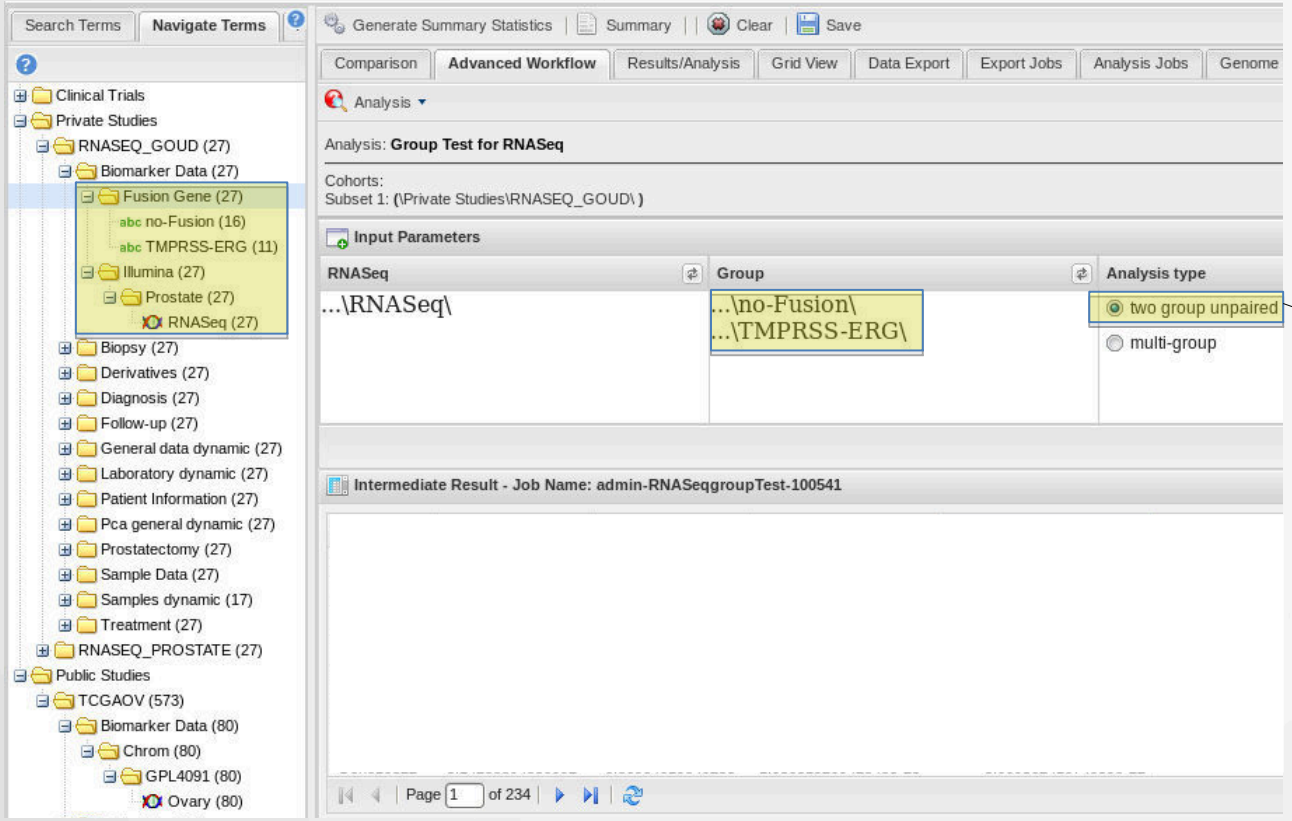
- Drag 'n Drop

Example: PoC: RNA-seq



- Drag 'n Drop
  - Creates expression- & design matrix

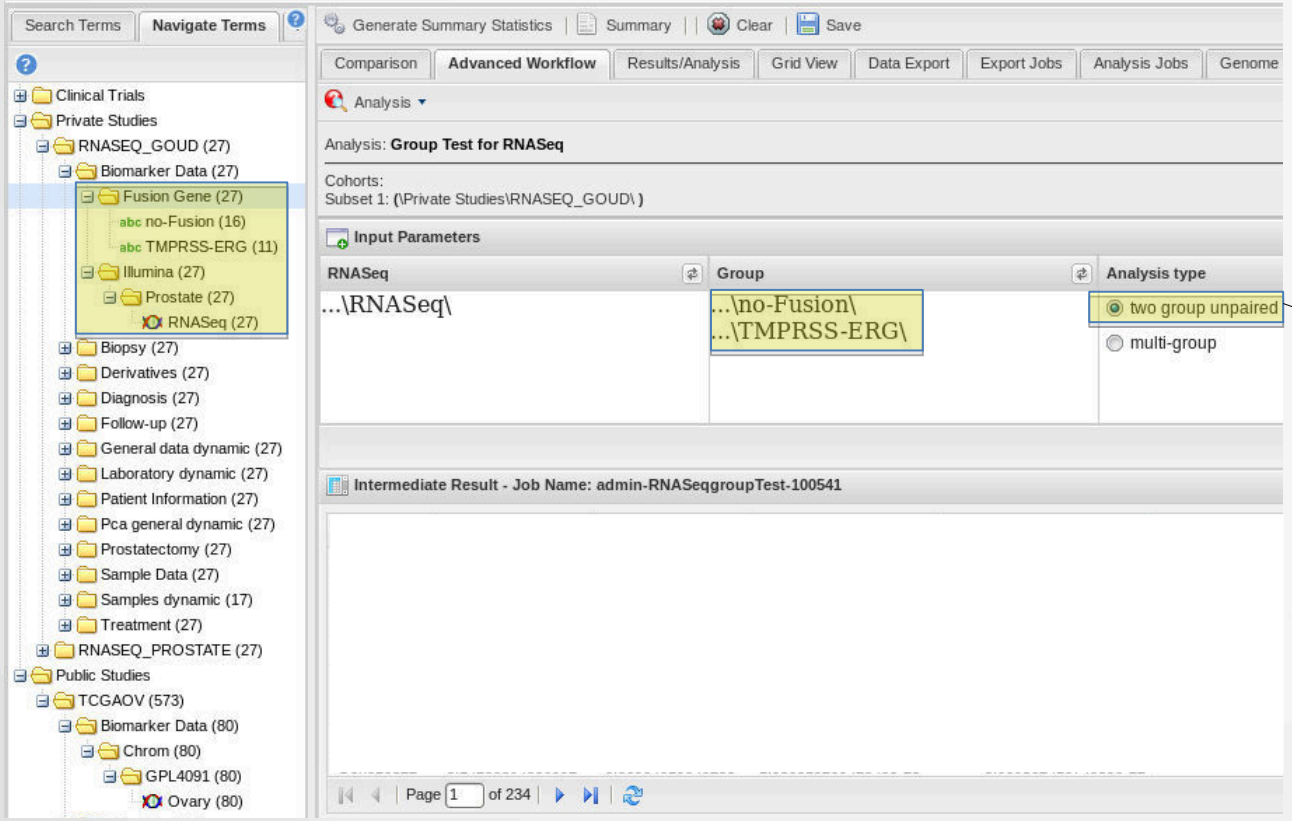
Example: PoC: RNA-seq



- Drag 'n Drop
  - Creates expression- & design matrix
  - Sends job to Galaxy

Galaxy PROJECT

Example: PoC: RNA-seq

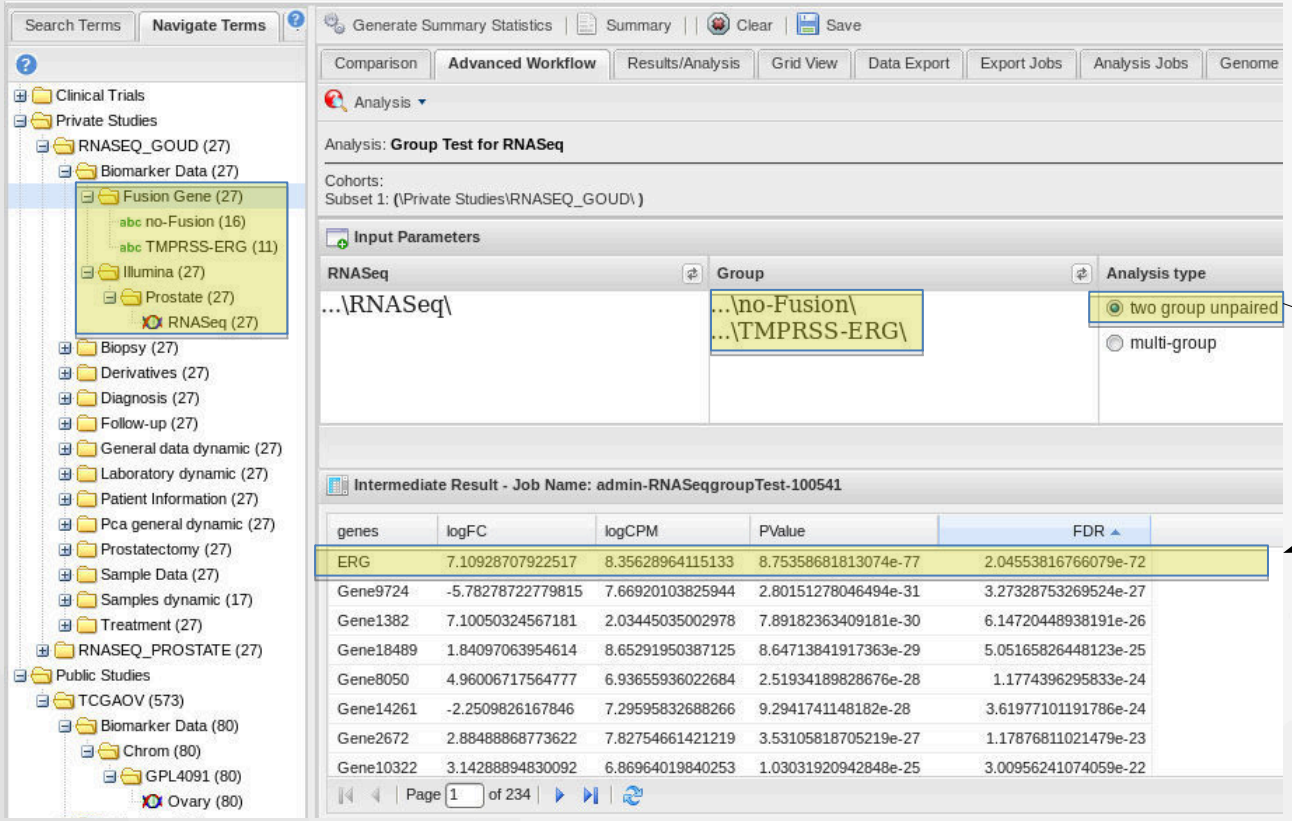


- Drag 'n Drop
  - Creates expression- & design matrix
  - Sends job to Galaxy

Galaxy PROJECT

- Runs edger_with_design_matrix

Example: PoC: RNA-seq

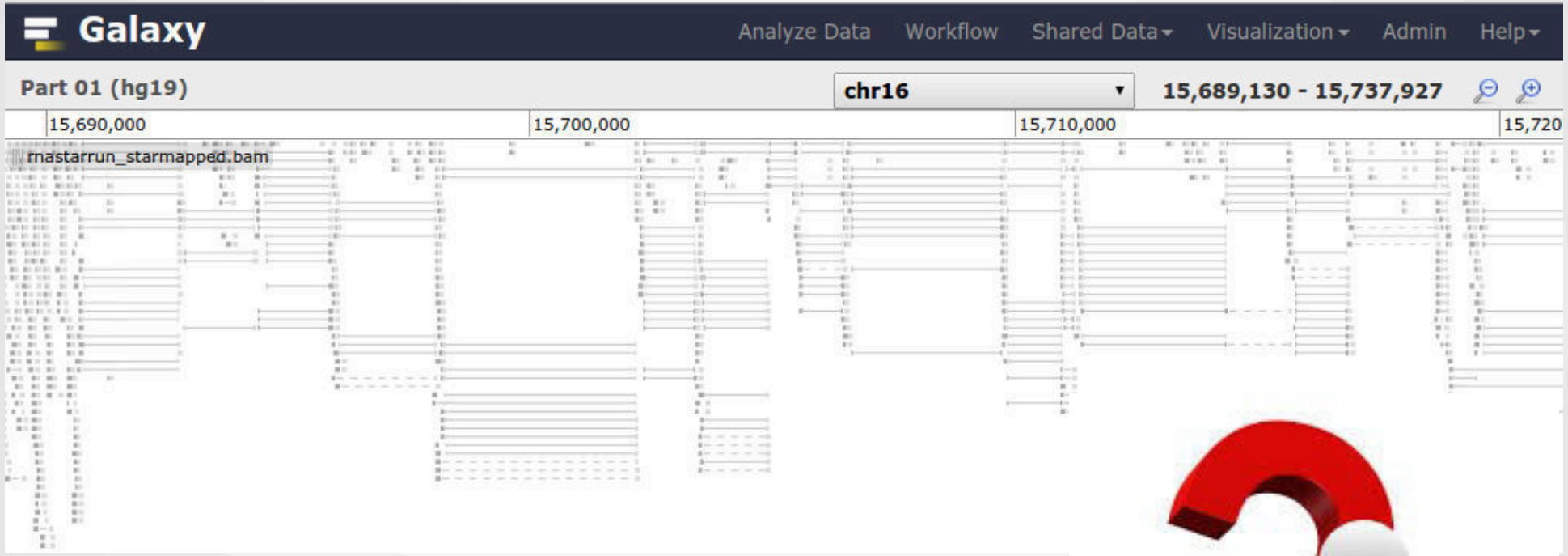


- Drag 'n Drop
  - Creates expression- & design matrix
  - Sends job to Galaxy

Galaxy PROJECT

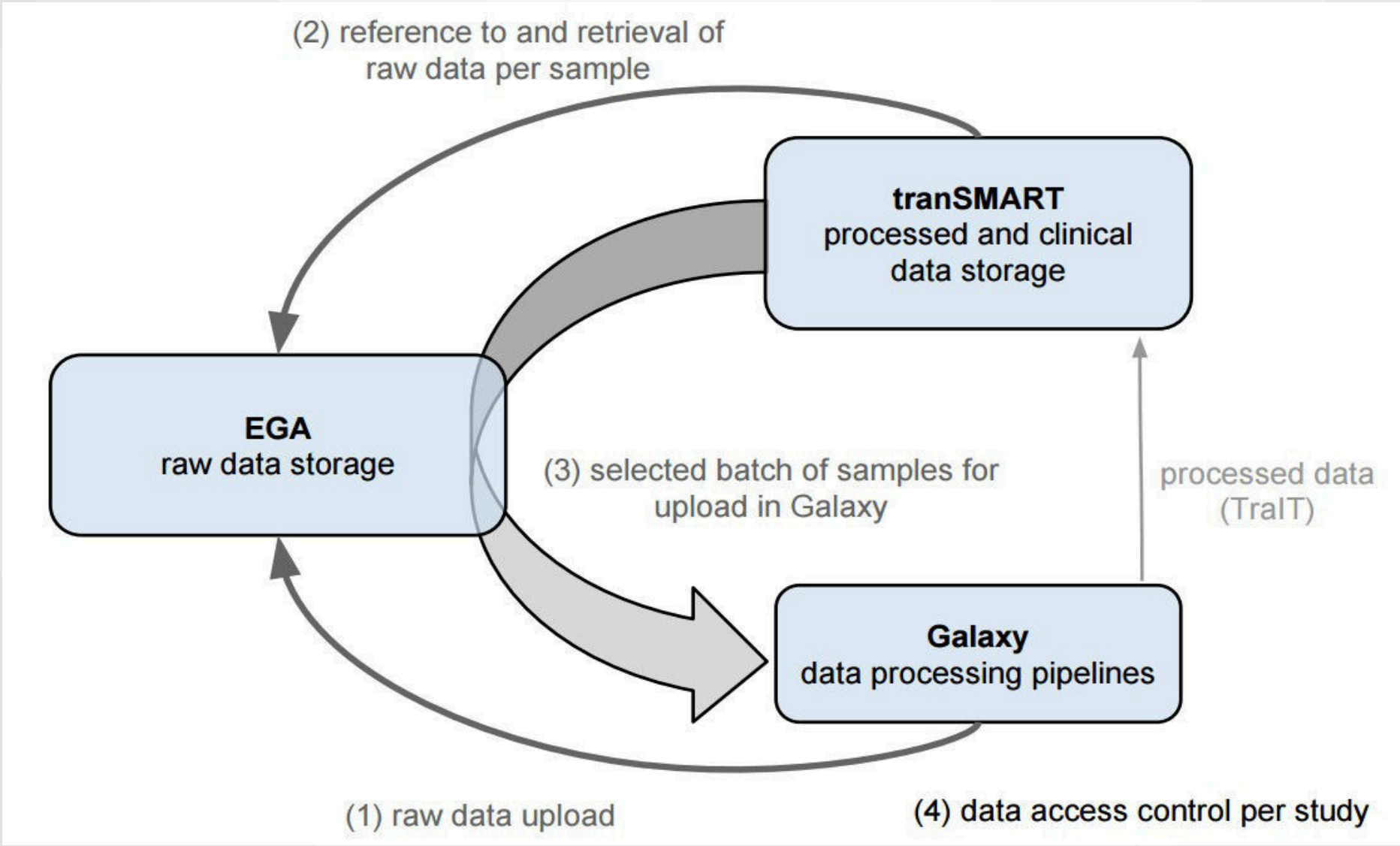
- Runs edger_with_design_matrix
- Returns statistics & plots

Example: PoC: RNA-seq



- How to get back to the reads?

Export: EGA


(2) reference to and retrieval of raw data per sample
tranSMART
processed and clinical data storage
EGA
raw data storage
(3) selected batch of samples for upload in Galaxy
processed data (TraIT)
Galaxy
data processing pipelines
(1) raw data upload
(4) data access control per study

Acknowledgements

- Andrew Stubbs
- Freek de Bruijn **<< TranSMART to Blend4j interface**
- Guido Jenster
- Jan-Willem Boiten
- Jochem Bijlard
- Remond Fijneman
- Ruslan Forostianov **<< Internal TranSMART modifications**
- Sanne Abeln
- Wim van der Linden